WELCOME TO THE NEW MAGAZINE FOR COMPUTING & DIGITAL MAKING EDUCATORS

(Hello World)

Issue 3 | Autumn Term 2017 helloworld.cc

10 TIPS TO SUPPORT INDEPENDENT LEARNERS

Keep your independent students engaged and motivated with our expert advice

BACK TO BASIC

Using BASIC and GOTO to teach the foundations of programming

CODE CLUB FOR TEENS

New resources for 9- to 13-year-olds

BUILD PONG IN SCRATCH

A lesson plan for 11-year-olds

APPROACHING ASSESSMENT

Our international panel of experts discusses innovative approaches and technologies for assessing programming skills

EMBRACING PLAY

How play could make you a better educator

OPEN-SOURCE RESOURCES

Computing has benefited from open-source. Can education too?

PLUS LEARNING THROUGH HACKING • WILL A COMPUTER TAKE YOUR JOB? • CS UNPLUGGED • MATHEMATICAL MUSINGS • YOUTUBE CLASSROOM • RASPBERRY PI & CODERDOJO MERGE • WHAT DOES A PROGRAMMER DO? GETTING GIRLS CODING • APPS FOR GOOD AWARDS 2017 • ASSIGNING VARIABLES • SCHOOL LEADERS MATTER

A BIG CAS Barefoot Thank You!

Supported by BT

As we draw to a close the 16/17 academic year, it is clear that this has been the Barefoot Computing Project's best so far.

In 2016 / 17 alone we have reached:

- **20,000 teachers**
- **1/2 million pupils**

Overall we've reached over:

- **1.2 million pupils**
- **160,000 downloads**
- **40,000 teachers**

None of Barefoot's achievements would have been possible without the people who continue to improve the resources, the volunteers who run free workshops and, of course, the teachers who use Barefoot resources to help pupils develop their tech literacy.

Barefoot highlights from 2017:

1 MILLION PUPILS REACHED

In February 2017, Barefoot pupil reach smashed through the one million mark! There are now well over one million primary school pupils across the UK benefitting from Barefoot's free resources.

Barefoot across the UK

Barefoot was officially launched in Scotland, Wales and Northern Ireland, complete with ministerial backing. To support this, all resources - and workshops - were also made available in Welsh and Gaelic. The workshops were also refreshed at the same time, bringing to life more resources in a fun and hands-on way for even more teachers.

Barefoot BYTES

Barefoot Bytes were launched just a few weeks ago and are already making a big impact. The Bytes are an exciting series of films that demonstrate Barefoot's resources in action. Hosted by the charismatic Dr Jon Chippindall, they are designed to help teachers get started on their computing journey. If you missed them, search 'Barefoot Computing Project' on YouTube.

Thanks again for all the support you have provided this year. If you're already using Barefoot, please keep up the good work by recommending Barefoot to other teachers and help us spread the word.

Curriculum aligned

Easy to teach

Free

Register and book your free workshop today: www.barefootcas.org.uk

(Hello World)

CELEBRATING
10,000
SUBSCRIBERS

SUBSCRIBE IN PRINT FOR FREE TURN TO PAGE 32

HELLO, WORLD!

Welcome to issue three of Raspberry Pi and BCS/Computing At School's magazine for digital making and CS educators worldwide.

For those of us in the northern hemisphere, we're at the start of a new school year, with much enthusiasm and excitement about new courses, classes and clubs. The attention of many English secondary teachers is likely to be on the new Non-Examination Assessment that now forms part of GCSE Computer Science – exa.foundation's Alan O'Donohoe provides an excellent 'bluffer's guide' here.

We're exploring more aspects of assessment in this issue's cover feature, with expert insights into how projects and questions can help students and teachers alike understand where they are and where they're heading. Elsewhere in this edition, Chris Roffey discusses the Bebras computational thinking challenge and Oliver Quinlan explores 'assessment for learning' in computing. As Nicholas Tollervey remarks in his feature on life as a developer, "Tests express how I expect my code to run, and well-tested code is better because writing tests makes you reflect on how your code should behave." Is something similar true for teaching?

Marc Scott provides some insights into the connections between open-source software and open educational resources. Hello World itself is an open educational resource – our content is all published under a Creative Commons licence, and we'd love to share your news, views and lesson plans! We're launching a letters page in this issue, and would be very glad to receive contributions from any readers – just drop us a line at contact@helloworld.cc.

Miles Berry
Contributing Editor

FEATURED THIS ISSUE

LINDA LIUKAS
PROGRAMMER, STORYTELLER AND ILLUSTRATOR

Linda is a programmer, storyteller and illustrator from Helsinki, Finland. She is the creator of the **Hello Ruby** books, bringing computational thinking activities to 4- to 10-year-olds.

ALAN O'DONOHOE
CAS MASTER TEACHER AND LEADER OF EXA.FOUNDATION

Alan taught Computing and Technology in schools for more than 20 years. He runs **Exa.Foundation**, a non-profit organisation that promotes and supports computing and digital making.

CAT LAMIN
RASPBERRY PI CERTIFIED EDUCATOR AND CAS MASTER TEACHER

Former primary school teacher, founder of **Coding Evening** and real-life geek girl, Cat is enthusiastic about getting teachers and children interested in coding and computing.

(HW)

EDITORIAL

Managing Editor
Russell Barnes
russell@helloworld.cc

Contributing Editor
Miles Berry
miles@helloworld.cc

Sub Editor
Rachel Churcher

DESIGN

Critical Media
criticalmedia.co.uk

Head of Design
Dougal Matthews

Designers
Lee Allen, Mike Kay, and
Daiva Bumelyte

Illustrator:
Sam Alder

Cover photography:
Brian O'Halloran/Raspberry Pi Foundation

CONTRIBUTORS

Lucy Hattersley, Phil King, John Stout, and Mark Wainwright

Sponsored by BT

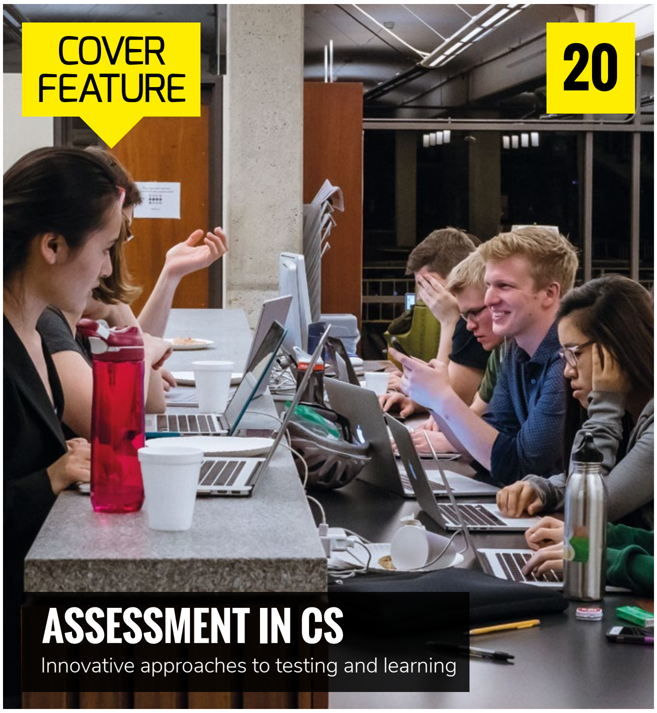
Hello World is a joint collaboration:

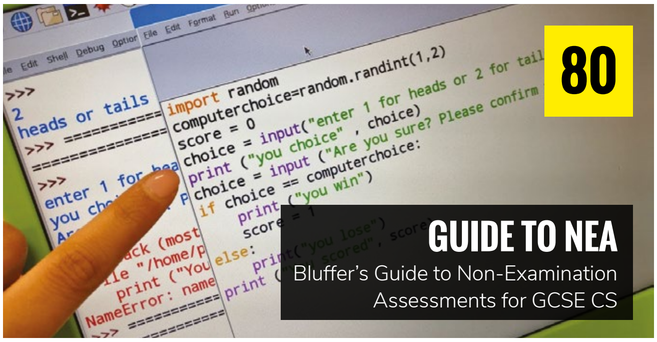
Raspberry Pi
bcs The Chartered Institute for IT Academy of Computing
COMPUTING AT SCHOOL EDUCATE · ENGAGE · ENCOURAGE Part of BCS, The Chartered Institute for IT

This magazine is printed on paper sourced from sustainable forests and the printer operates an environmental management system which has been assessed as conforming to ISO 14001.

Hello World is published by Raspberry Pi (Trading) Ltd., 30 Station Road, Cambridge, CB1 2JH. The publisher, editor, and contributors accept no responsibility in respect of any omissions or errors relating to skills, products or services referred to in the magazine.

(HW) CONTENTS





NEWS AND FEATURES

CONVERSATION

LEARNING

TUTORIALS & LESSON PLANS

The finalists and winners gather on stage to celebrate their achievements

APPS FOR GOOD AWARDS 2017

Students pitch their innovative technology ideas to experts at annual awards

The Apps for Good 2017 annual awards ceremony has taken place at the Barbican Centre in London. The event rewarded the hard work of young tech entrepreneurs around the world, and students took the opportunity to pitch their ideas to industry experts and partner tech companies.

Solving problems with technology

Finalists were selected from the 25,000 students who created a technology product tackling the problem or issue they care about. The prizes at stake, such as lucrative work experience opportunities and mentoring from industry leaders, mean that the weeks leading up to the awards can be tense for the selected finalists.

"It was both exciting and daunting," said Kirsty and Savannah from Booksy (**helloworld.cc/2fpmk7C**). After weeks of talking through, designing, and researching their app, they had the chance to present it to an audience of industry experts. Booksy, an app for simple and social note taking for GCSE revision books, won the prestigious Information Category award.

Now in its sixth year, Apps For Good celebrates today's young tech entrepreneurs. The event, at the end of the school year, is a culmination of all the ideas, learning and hard work that students and teachers have put into the completion of the Apps for Good course. Prizes recognising the achievements of the outstanding teachers who deliver the course, as well as the dedication of the schools which take part, are also presented on the day.

ENTER APPS FOR GOOD 2018

Apps for Good is an educational movement powering a generation to change their world with technology. Students aged 8-18 work together in teams to find real-life issues they care about, and learn how to solve them using technology. Since the launch in 2010, its courses have been delivered to over 100,000 students in more than 1,500 schools across the UK and internationally.

Give your students the opportunity to be award winners in 2018 by delivering the free Apps for Good course. For information on becoming an educator, go to **helloworld.cc/2vLaADm**.

Pitching to the experts

All the finalist teams spent the morning at accelerator sessions, perfecting their presentations with the support of partner tech companies. There was a buzz of excitement as the teams arrived at the Barbican ready to pitch to the panel of experts. The team behind the Home Help app (**helloworld.cc/2ulBSzQ**) summed up the sentiment among the students: “We were really happy to get shortlisted but a bit nervous. After working with our accelerator, we were well prepared for the presentation.”

As everyone prepared to showcase their ideas to the judges, the impressive prototypes created by the Internet of Things course finalists were a real talking point among students, teachers and Apps for Good team members. 2017 saw the IoT course piloted in ten schools, and the ideas on display at the awards were intriguing and eye-opening.

Team Micro:gate, Maliha, Talia, Eden, and Macey (**helloworld.cc/2ulqLCW**), from Thornleigh Salesian College, present their product, which aims to easily create GCSE science practical equipment

> “ THE MARKETPLACE IS AN IMPRESSIVE DEMONSTRATION OF ALL THE DIFFERENT ASPECTS OF THE APPS FOR GOOD COURSE

After presenting their ideas to the dragons and their fellow finalists (which many students found the most nerve-wracking part of the day!) it was time to head to the marketplace.

Ideas showcase

The buzz of the marketplace was the most fun part of the day and students say it’s what they look forward to most. They have the chance to showcase their ideas to industry leaders, experts and professionals. Everyone is excited, and students’ ideas are brought to life with illustrated displays. The marketplace is an impressive demonstration of all the different aspects of the Apps for Good course. The display of creativity and teamwork behind the ideas proves how much the teams achieve on top of the technical aspects of the course.

As well as inspiring the guests at the marketplace with their ideas, the students’ other aim was to win votes in the prestigious People’s Choice Award. Guests were faced with a very difficult choice, as the products offered solutions to a wide variety of problems in very innovative ways.

Everyone’s hard work culminated in the award ceremony, which was presented by Apps for Good Fellows (graduates) Jashvanth and Alex. The Fellows demonstrated the incredible opportunities available to graduates of the course. As the winners in each category were announced, everyone could see how their hard work and determination had brought them this far. Teachers, students and guests left the awards inspired and excited to continue working on their technology ideas. (HW)

The team from Holy Trinity CofE Primary School, David, Zamyrah, Hayden, Samiyah, Caleb and Kamira, bring their app Read:it (**helloworld.cc/2uGjYDm**), which helps children to read on their own, to the marketplace

CODE CLUB AGE RANGE EXPANDS WORLDWIDE

New resources and training available for Code Clubs for 9- to 13-year-olds

Code Club is expanding to cover older students, aged 11 to 13 years. From September, the popular coding clubs will be officially supported for 9- to 13-year-olds globally.

"There are currently almost 6000 Code Clubs in the UK, and already more than 600 of them are in secondary schools," says Maria Quevedo, Director of Code Club UK.

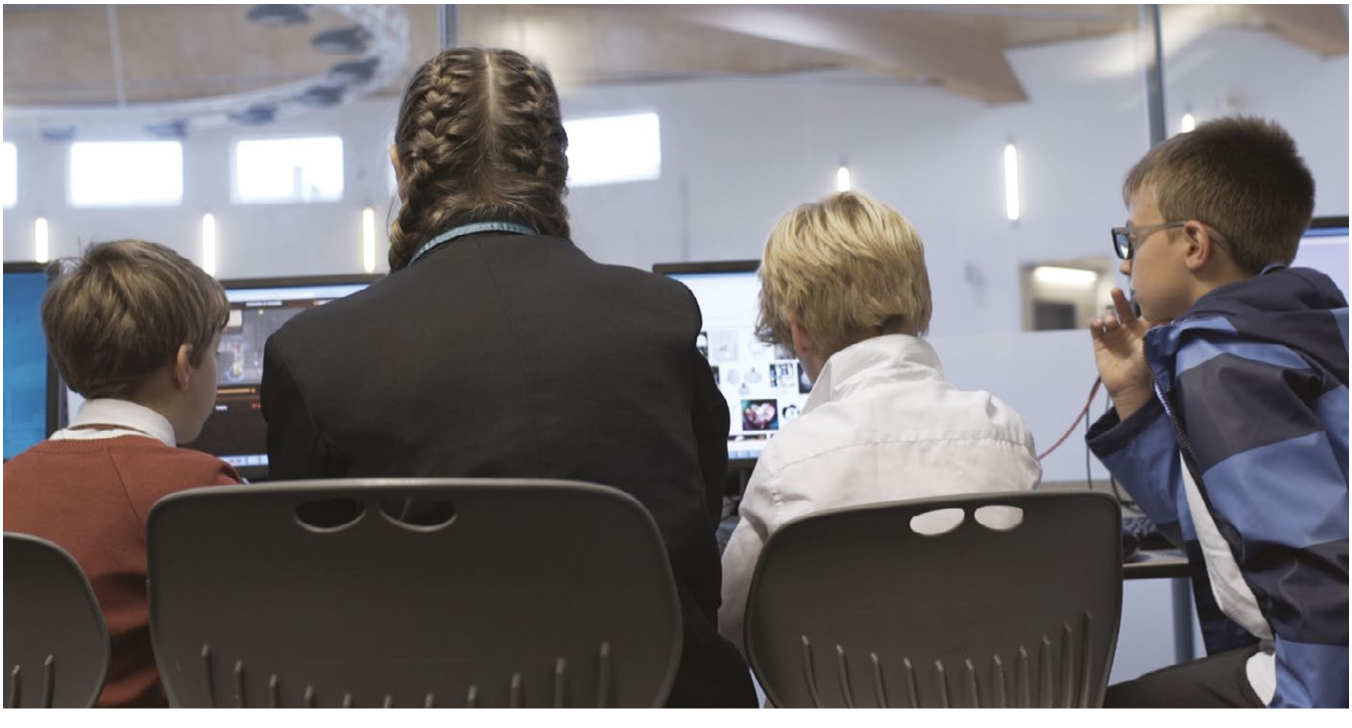
Student feedback

Code Club chose 50 of those schools to run a pilot programme during the summer term. The programme was used to understand how Code Clubs are currently working with the 11-13 age group, and to collect feedback on how Code Club's new projects would work with an older age range.

"We asked them to try out some projects, and they told us whether they were too hard or too easy," says Sarah Sheerman-Chase, Participation Manager for Code Club UK.

We spoke to a teacher who has been running a Code Club for 11- to 13-year-olds. Natalie Stewart, Subject Leader for Computing and ICT, and STEM Coordinator at Hadley Learning Community, told us: "The students were more than excited at the prospect of more challenging projects." Until now, they have been enjoying Code Club's existing projects. A particular favourite among students was 'Rock Band', a project where students compose songs from the current charts using new instruments. Games such as 'Beat The Goalie' also proved popular. "This became a competition amongst the students to see who could make their game most difficult," Stewart tells us.

Following feedback from the trial programme, Code Club has released new resources based on Sonic Pi, an open-source programming environment that can be used on any computer to make music using code. New resources include building drum loops, creating special effects, and a live DJ project.



Older Code Club children can mentor, and look after, younger recruits

START A CODE CLUB

Anybody can set up or volunteer at a Code Club. As a Code Club leader, you'll play an essential part in helping children gain useful skills: not just in coding, but in problem-solving, planning, design, and collaboration.

You don't need to be a programmer, and you can learn along with the students. All you need is a venue, computers with web browsers, and children aged 9-13. Visit the Code Club website for more advice on how to get started: **helloworld.cc/2vBdlXz**

Code Club's free resources can be thought-provoking for older students

"We think they will be very popular with teenagers," says Sheerman-Chase. Code Club is also planning to launch new projects in the Autumn term for more advanced students.

> THE STUDENTS WERE MORE THAN EXCITED AT THE PROSPECT OF MORE CHALLENGING PROJECTS

Existing resources include projects based on Python, HTML, CSS, and physical computing – also perfect for use with an older age range.

Mentoring younger students

The Code Club resources were "invaluable" said Stewart. Eventually, the students were running the Code Club and "supporting each other, especially new recruits." It is this change to spanning a wider age range of students that lends itself to mentoring. Code Club will be a superb "transition activity" for students moving from primary to secondary school, says Quevedo. "For example, Code Club could be a good thing to do on induction day, and we know of schools where the older students support the younger ones at Code Clubs."

Code Club resources are now officially available for 9- to 13-year-olds

Support and resources

Teachers of 9- to 13-year-olds can access all of the free resources available at Code Club.

"The idea behind the projects is that they're flexible," Quevedo tells us. You can use any computer with a web browser, and all the materials and training are free.

Club leaders do not need to be coding experts to run a club. They can come to one of Code Club's regular training sessions, or learn alongside the students. They can also complete any of the Raspberry Pi Foundation's courses on the FutureLearn platform (see page 42 for details). These courses can be a "fantastic" preparation for running a Code Club, says Quevedo.

All the Code Club resources are free to download (**helloworld.cc/2vJpM2Z**). Teachers and volunteers can find plenty of support for their Code Club by registering at the Code Club World website (**helloworld.cc/2vfNNhE**).

It is important that teachers register their Code Clubs with us, Quevedo reminds us. The resources are open to everybody, but there are "huge advantages to registering, as registered clubs have access to lots more resources and support." Teachers who register gain access to supporting materials, such as posters, certificates, and competition prizes. (HW)

A BIT OF CS4FN

Computer Science for Fun

Issue 1

Machines that learn

Make me invisible!

The robot painter

Spies like us

Queen Mary University of London

The pilot issue of A bit of cs4fn. Image credit: A bit of cs4nf

CS4FN MAGAZINE

A new free magazine for primary schools

The CAS London team behind Computer Science for Fun has created a new free magazine for primary-aged pupils called A bit of cs4fn. The pilot issue of the magazine combines articles with games and puzzles.

"A bit of cs4fn is designed to help girls and boys to become computational thinkers for fun," says Paul Curzon, Professor of Computer Science at Queen Mary University of London.

The magazine mixes computing with topics from other subjects, supporting numeracy and literacy while drawing on history, science, music, and art. The pilot issue explains how the Romans invented pixels, so they could have mosaic floors anywhere in the empire. Another feature explains how Victorian Countess and mathematician Ada Lovelace foresaw the key ideas behind the digital age.

"We also investigate whether an invisibility cloak could ever become a reality," says Curzon; and the magazine explores what "machine learning is all about" to explain how our brains work.

Sign up for free copies of A bit of cs4fn at **abitofcs4fn.org**. (HW)

NEW ZEALAND ANNOUNCES NEW DIGITAL TECHNOLOGIES CURRICULUM

Digital Technologies expanded to cover primary and secondary schools

New Zealand's Education Minister, Nikki Kaye, has announced a new curriculum for students from their first year of schooling.

Called Digital Technologies/Hangarau Matihiko, the new curriculum is backed up with $40 million of funding to support the change. The money will provide professional learning and development for teachers, the development of interactive resources, and an equity fund, plus scholarships, and competitions for students.

A draft of the curriculum has been published for consultation, which will run until mid-November. A final curriculum is to be released for use from the beginning of 2018, becoming mandatory in 2020.

The draft Digital Technologies material is equivalent to computing in the UK. It has two knowledge areas: Computational Thinking (CT) and Designing and Developing Digital Outcomes (DDDO). CT includes topics such as algorithms and programming, while the DDDO area includes 'understanding of the digital information technologies that people need in order to locate, analyse, evaluate, and present digital information efficiently, effectively, and ethically', as well as becoming 'more aware of how to build, install, maintain, and support computers, networks, and systems'.

The draft includes a Te Reo Māori (Māori language) version that covers the same material as the English medium curriculum, but also has extra material that reflects the unique aspects of Te Marautanga o Aotearoa (the national curriculum for Māori-medium teaching). The aim is to 'enable students to learn about digital technologies in various situations from a Māori world view, and demonstrate Māori values and principles to ensure that designers and users create a positive impact in their whānau, hapū, iwi and local and global environment.'

New Zealand's Education Minister, Nikki Kaye

The draft curriculum for consultation is available at **helloworld.cc/2ule2jQ**. The official announcements can be found at **helloworld.cc/2vicQz0** and **helloworld.cc/2ulTE67**. (HW)

ROYAL SOCIETY COMPUTING EDUCATION REPORT 2017

Teachers can download the 2017 report from October

The Royal Society is due to launch its new report into the provision of computing in UK schools.

The report is influenced by evidence gathered from individuals and organisations who are interested in computing. Teachers will be able to download the report in October from the Royal Society website, at **helloworld.cc/2hFn755**.

The new report follows the Royal Society's seminal 2012 report, *Shut down or restart?* (**helloworld.cc/2vibUdX**).

The old national curriculum in ICT "has been discontinued" said David Do Quy, Computing Project Leader at The Royal Society. In its place, we have a computing curriculum with three areas: IT, digital literacy, and computer science.

"A well-taught curriculum should cover these three strands", said Do Quy, so that young people are equipped with the skills to "thrive in our digital world".

Teachers need "more support and investment," said Do Quy.

The 2014 reform moved the curriculum away from an ICT approach, to one focused on computational thinking skills. While there is widespread enthusiasm for the new subject, it has not yet reached the same level of uptake at A level as the old ICT qualification. Do Quy notes that "more work needs to be done" to grow the subject, "especially for girls". (HW)

> A COMPUTING CURRICULUM NEEDS TO FOCUS ON THREE AREAS: IT, DIGITAL LITERACY, AND COMPUTER SCIENCE

Recruitment of computing teachers also needs to be addressed. Only 68 percent of the recruitment target is reported to have been met, which is lower than in any of the sciences. The report will outline actions to address the shortage of computing teachers.

Philip Colligan (left), CEO of The Raspberry Pi Foundation, and Giustina Mizzoni (right), Executive Director of the CoderDojo Foundation

RASPBERRY PI MERGES WITH CODERDOJO

Teaming up to deliver more for young makers and coders

A Dojo in action, with experts and mentors helping kids make and code

The Raspberry Pi Foundation and the CoderDojo Foundation have merged, with Raspberry Pi Foundation CEO Philip Colligan joining the CoderDojo board. The two Foundations will share back-office resources and expertise to deliver more quickly on their shared goals.

CoderDojo fans won't see any change at the Dojos, however. CoderDojo spokesperson Rosa Langhammer tells us that "CoderDojo is and will always remain 100 percent content, software and hardware neutral! The Raspberry Pi Foundation works to put the power of digital making into the hands of people all over the world," continues Rosa, "and the CoderDojo mission is to give every young person an opportunity to learn and create with technology. By combining our efforts we get closer to these shared goals."

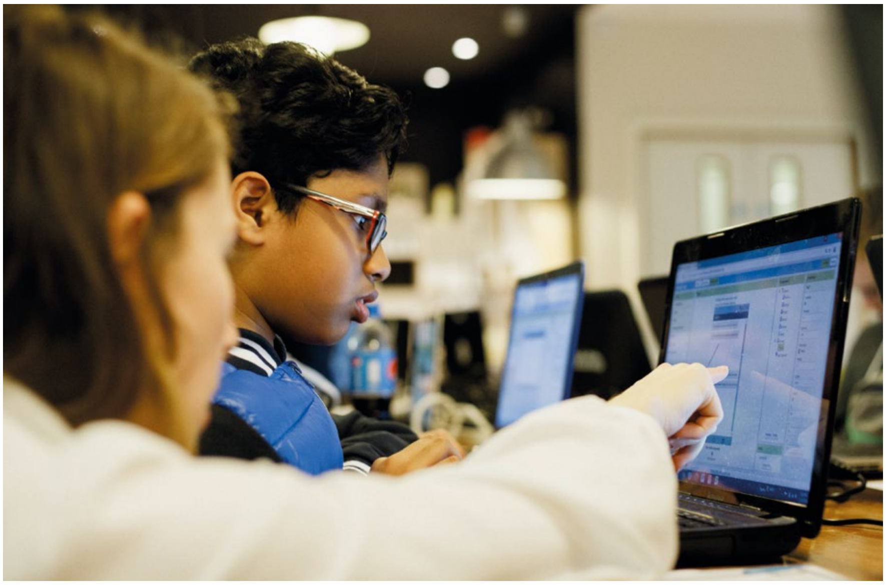
The merger will allow CoderDojo to reach even more young, enthusiastic makers and coders. We'll hear more about these plans soon, but Rosa did reveal that part of the strategy review includes "mapping the path to 5,000 Dojos [worldwide] by 2020."

But, "the first priority," Philip tells us, "is to make it as easy as possible to set up and run a CoderDojo. That means investing in the tools and resources that CoderDojo champions and mentors need."

Enabling more people and businesses to set up Dojos won't be easy. That's why Raspberry Pi will be "bringing all of the capabilities and expertise of The Raspberry Pi Foundation to support the growth of the CoderDojo movement around the world."

"The CoderDojo Foundation team is really excited," says Rosa, "and is looking forward to working with a wider team of people with similar goals, cooperating on new projects, and pooling all our experiences and talents to give more people the opportunity to learn and be creative with technology."

> WE DECIDED WE SHOULD JOIN FORCES TO ACCELERATE THE GROWTH OF THE CODERDOJO MOVEMENT



Any hardware and software is fine, as long as you're coding

WHAT IS CODERDOJO?

Aimed at young people between the ages of 7 and 17, CoderDojos are free, community-led programming clubs for anyone curious about coding. Whether that's building websites and apps, or just figuring out how technology works, Dojos have always offered freedom from hardware and software restrictions - any tool is fine, as long as you're coding.

"The CoderDojo mission is to give every young person an opportunity to learn and create with technology," as spokesperson Rosa Langhammer puts it.

Key to the movement are the volunteer 'Champions' who set up the Dojos. If you or your organisation would like to set up a Dojo, the CoderDojo Foundation will help you get started. Visit **helloworld.cc/2vBgAhE** for details. You don't need technical skills to start a verified Dojo, just the ability to bring people together, and a passion for the power of code to build a positive future.

To search for a Dojo near you, check the interactive map at **zen.coderdojo.com**.

Perfect timing

Why now? "In a word: demand," Philip explains. "We know that young people all over the world are crying out for places they can go to learn how to get creative with digital technologies. Education systems aren't responding quickly enough to that demand, so we decided that we should join forces to accelerate the growth of the CoderDojo movement."

Both Foundations were established at around the same time, and "we've been working together in lots of ways from the beginning," says Philip. "Lots of the people who are CoderDojo champions and volunteers are also members of the Raspberry Pi community." He also confirms that "we have no plans to change the approach to Code Club or CoderDojo; our priority is encouraging the growth of both initiatives all over the world." (HW)

CARRIE ANNE PHILBIN DIRECTOR OF EDUCATION AT THE RASPBERRY PI FOUNDATION

WATCH AND LEARN

How online video is changing the way we teach computer science

I remember writing my first computer program on a BBC Micro as if it was yesterday. It was of course during a maths lesson in which I had already completed the set work, so I was allowed to use the computer! Logo was my favourite – typing commands to make pretty geometric pictures on a monitor was exciting.

The BBC Micro has a lot to answer for. It inspired a generation of computer enthusiasts and professionals like me, and led to the development of the Raspberry Pi Computer and other small form factor programmable devices. But what was really revolutionary about the BBC Micro was its accompanying television series, called *The Computer Programme*, where you could learn to use it alongside the host, Chris Searle.

Before this TV programme, the most common way to learn about computers and how to program them was through magazines and books, or even the trusty computer manual. The availability of video tuition really helped me, and now 30 or so years later we have video-based platforms that provide content on every aspect of computing, computer science and technology. They can help us both develop our subject knowledge as computer science specialists, and support our students through all the stages of their learning.

Content for teachers

Continued professional development can be tricky to fit in with the everyday demands of being a teacher. Trying to find time to be released from the timetable to attend courses is always a cause of frustration. Thankfully, Computing At School (who spend every moment thinking about how they can support computer science teachers) created CAS TV (**helloworld.cc/2hktjzb**), a YouTube channel full to the brim with content for teachers at every stage of their computing journey. Videos include tips and tricks on how to teach particular concepts at different key stages, learning how to plan a scheme of work, and even how to engage under-represented groups.

Some CAS TV videos also explore creativity and innovation with leading developers. Genevieve Smith-Nunes looks at how dance can represent data (**youtu.be/ulj7DMCHdkA**), and Andrew Fitzgibbon introduces computer vision and machine learning (**youtu.be/AqEqTQeQ2DI**). Many of the videos are delivered by practising teachers in both primary and secondary schools, including Phil Bagge and Rebecca Franks, as well as leading teacher trainers like Miles Berry, Jane Waite and Alan O'Donohoe. It is quickly becoming the one-stop shop for a short burst of professional development.

Learning with online video has become so popular that whole courses have been developed for video delivery, for

■ Access bite-sized CPD from leading teacher trainers on the CAS TV YouTube channel

Practical lessons from Geek Gurl Diaries

example on the Open University's FutureLearn platform, or Harvard and MIT's edX. In the last few years, this MOOC (massive open online course) structure has moved towards developing educators as well as teaching students. The Teaching Computing course (**helloworld.cc/2f55FWX**) from The National STEM Learning Centre and the University of East Anglia is available free of charge, and can be worked on at your own pace. This year The Raspberry Pi Foundation launched a number of courses on FutureLearn (**helloworld.cc/2uye60k**), including *Teaching programming in primary schools*, *Teaching physical computing with Raspberry Pi and Python*, and *Object-oriented programming in Python* (more details on page 42). The MOOC model also includes plenty of social interaction, with the FutureLearn platform promoting lots of discussion points and opportunities for comments.

> “ Learning with online video has become so popular that whole courses have been developed for video delivery

Content for students

One of the best features of online video is that it can be played over and over, slowed down or paused. There were days when I wished I could replay my teachers' explanations, especially when I was revising! Online videos can also be used to help supplement the teaching of difficult concepts for students.

Sometimes understanding how the ALU works, and then being able to explain it to someone else, or even answer an exam question on it, can be difficult. Thankfully, the team behind the successful YouTube channel Crash Course has created an entire series on computer science (**helloworld.cc/2ugM0ox**), supported by PBS Digital Studios. The videos break down concepts into ten-minute chunks of explanation, with animations to show what is happening. Another great channel is Computerphile (**helloworld.cc/2hkMtVP**), which boasts videos 'all about computers and computer stuff', and is really fun to watch. Resources like this make great homework or revision exercises, and give students that extra support.

Skills-based videos are a great way to supplement student and teacher development, especially in programming. When I'm trying to work out how to write a function in an unfamiliar language, I'm often drawn to find the answer through online video. I learnt lots about Python thanks to Trevor Payne's video tutorials. They start off with the basics of computer programming and expand to cover broader and deeper topics (**helloworld.cc/2w2pceq**).

Embracing video

While teaching in a state secondary school in Dagenham, I noticed two things: most of the students who had chosen my subject were boys, and they would often search for supplementary material – not on Google, but on YouTube. I decided that I would start to create online video content that demonstrated my passion for computer science and digital making, as well as what I was learning or teaching at the time. My channel, Geek Gurl Diaries (**helloworld.cc/2ugl7Ad**), was a real labour of love outside the classroom. Recently, I've taken a break from producing content to work on *Crash Course Computer Science*, and learn from the professionals how to make great educational videos. However, I will continue to create content for my channel, and I encourage you as a computing education practitioner to do the same. You'll help your students even more than you already do, and without realising it you'll also help many more learners around the world.

When I think back to how I learned about computer science, I know that video played a key role. When I think about my continued learning in this field, I know that I turn to video first to find what I am looking for. When it comes to teaching, I make videos for others to learn from. Let's embrace online video – even though I'm sure we will never be able to compete with the original title: *The Computer Programme*. (HW)

Carrie Anne Philbin is Director of Education at The Raspberry Pi Foundation, a Computing At School board member, author and YouTuber.

Find free high-quality educational videos for everyone on the Crash Course YouTube channel

7-Eleven made the first legal U.S. drone delivery to a customer's home in July 2016 using Flirtey
The Raspberry Pi is a single-board computer intended to promote basic computer science that sells for $35
75
Carrie Anne is the Director of Education at the Raspberry Pi Foundation in the UK
Carie Anne hosts Geek Gurl Diaries, a series for teens who want learn about technology
YouTube
1. A robot dog must fetch the ball
2. A robot dog must bark at squirrels, unless this conflicts with the First Law
3. A robot dog must eat all the food, as long as such eating does not conflict with the First or Second laws

#INSIGHTS

ASSESSMENT FOR LEARNING

Diagnose your students' learning needs by asking the right questions

STORY BY Oliver Quinlan

Mention assessment in schools and most people think of tests. Memories of creaky exam halls and regulation stationery may come first, but there are several developments in assessment that can change how we think about discovering what students have learned.

Back in the late nineties, Black and Wiliam challenged educators' views on assessment with their seminal work *Inside the Black Box*, which popularised the idea of 'assessment for learning'. They suggested an approach which brought assessment into the learning activities themselves. When this is executed well, students are given feedback as they learn and have the opportunity to act on this feedback immediately. It's about the teacher not just delivering and then assessing later, but regularly checking for understanding and adapting their teaching. It's also about the learners getting regular insight into how they are learning and, crucially, having an opportunity to act on the feedback they get and ensure that they are making progress. The assessment is designed to serve the students' learning, and not to certify that they have achieved a set standard.

Assessment revolution

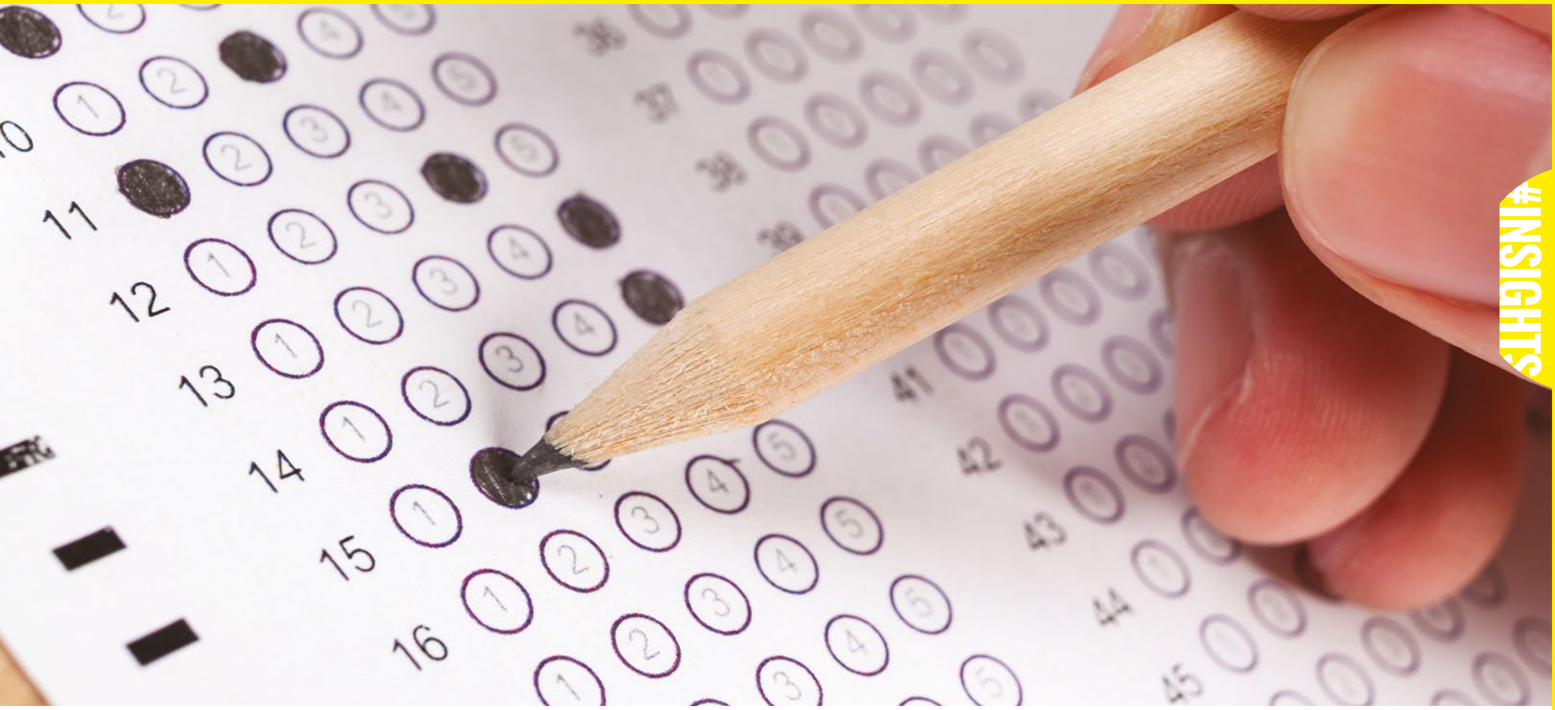
This work led to many developments in schools, with the government in the UK taking up 'assessment for learning', teachers being trained to regularly assess students' learning within lessons, and students being provided with feedback to act on as they learn. More recently this approach has been supported by Hattie's meta-analysis of influences on achievement in schools, which puts feedback at the very top in terms of the size of the measured effect.

Truly effective formative assessment is not just about finding out whether students have 'got it' yet. It's about understanding

Image: CoderDojo Dún Laoghaire

GETTING GIRLS CODING

The challenges around gender and computing are well documented, with women under-represented in technology jobs and in classes studying for computing qualifications. Lots of people are trying different approaches to addressing this problem. Dr Claire Quigley, from Glasgow Science Centre, worked with CoderDojo Scotland and Digital Scotland to look at some of these approaches and see how well they work.



how they are thinking about a topic, what misconceptions or naive understandings they have, and how your teaching or their activities can be adjusted to address this. Given the abstract nature of computing, the potential for misconceptions is very high.

Diagnostic questions

Ed tech company Diagnostic Questions is seeking to address this with their online assessment platform. Diagnostic Questions' assessments look familiar at first: multiple choice questions with four answers. Multiple choice questions have been given a bad reputation by some educators, but their quality all comes down to how the questions themselves are put together. If a question has a correct answer and three laughably implausible answers, then it won't be a useful tool. However, if each answer represents a different level of understanding, or a common misconception, then the answer the student gives is useful even if it is the wrong one. Imagine being told after an assessment not just who in a class had got the right answers, but why those who got it wrong did so, and potentially what misconception you need to address for each group of students.

For the last 18 months, Diagnostic Questions has been working with Computing At School and the Durham Centre for Evaluation and Monitoring to bring this approach to computer science. Project Quantum is a two-year project exploring the potential for crowdsourcing diagnostic questions for the computing curriculum, and using them for formative assessment. They have been encouraging teachers to add questions to the platform, and use the questions that have built up already, with their students to help them better understand their learning. This project brings the potential of a relatively new approach to assessment to computer science teachers, and a chance to better understand how students make sense of difficult topics.

Girls-only coding

Quigley looked at data from 36 CoderDojos in Scotland. These informal programming events had tested several initiatives to encourage more girls to get involved in their activities. Some tried girls-only introductory sessions as a way to create an environment that girls might find more attractive to get involved in.

The girls-only events did attract girls to attend and have a go at programming, providing an important introduction to the activity. However, this did not translate into more girls graduating from these introductory sessions to attend the mixed sessions. Quigley suggests that this may be because an introductory session is not enough to overcome the cultural trope that boys are more suited to technical subjects than girls. It could also reinforce the idea that girls do not usually code. There was some evidence from a Dojo that organised a sequence of girls-only sessions that this longer experience gives girls a chance to develop attitudes which move past these stereotypes. The dataset for this was small, but 40 percent of girls from this initiative went on to attend mixed CoderDojo activities.

Descriptive challenge

The work on descriptions showed that how an activity is presented may have a large effect on the proportion of girls. Descriptions emphasising creativity, familiarity, and with specific descriptions of what young people would do attracted more girls. Those emphasising competition, teamworking, the need to publicly showcase your work, or involving lots of jargon, were much less successful in attracting girls. The effect of descriptions that are attractive to girls was much stronger than the effect of the off-putting descriptions, so it's important to avoid things like jargon, but also to actively describe activities in a way that makes girls want to get involved.

These findings are closely linked to cultural contexts, so the effects in other parts of the world are likely to be different. However, they offer both an interesting set of recommendations for trying to attract more girls to informal computing activities, and a reminder that it is always worth collecting and looking at data to see whether the initiatives educators implement are working as intended.

You can read the full report here: **helloworld.cc/2uZ124z** (HW)

Comparative judgement

Creativity, problem solving and original approaches are key to computing, yet these things are very difficult to assess using traditional approaches. It's very common in education to use criteria to assess how well students are performing. We might set them a programming problem and then tick off whether they have used a Loop or an If statement, showing that they have understood those things. However, real-life programming is often more about the elegance of the design of a solution. What if the most accomplished student doesn't use the things you have on your checklist? This is a particular problem when using wide briefs for tasks, or even open-ended projects.

Comparative judgement is a field relatively new to education practice, which offers huge potential for solving this problem. It's based on well-established research showing that humans are relatively poor at making objective judgements about individual objects, but very good at making comparisons. Play a musical note to most people and ask them what it is and they will struggle. Play them two notes and ask them which is higher and they are likely to be successful. Repeat this several times, with a clever algorithm to keep track, present them with the right combinations, and you can come up with a scale. These rankings have been shown to be very reliable – even more so if you involve several people as judges.

This method has been shown to work well even for judging things for which we don't have clearly defined criteria, such as looking at working in maths and asking 'who is the better mathematician?' It can also be reliable, even when the judges are peers at a similar level of proficiency. This opens up some exciting new ground for the assessment of skills that involve approaching problems in an open-ended way, and assessing complex skills without resorting to trying to predefine what successful students must do. Assessment organisation No More Marking is exploring this approach for English and Maths in partnership with schools.

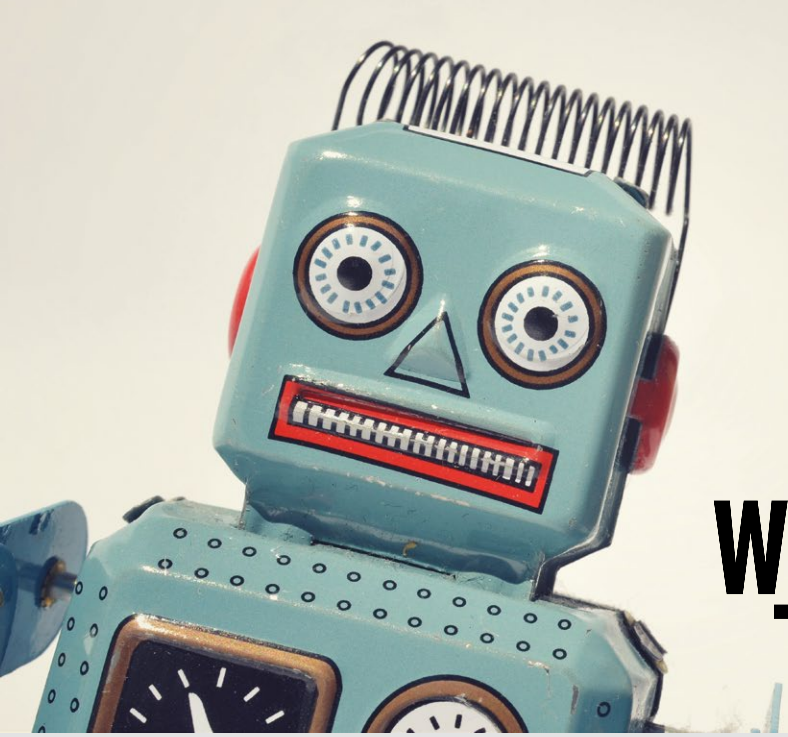
Assessment is all about students getting better at something, but it seems there are also some promising avenues for educators to get better at assessment. (HW)

> STUDENTS ARE GIVEN FEEDBACK AS THEY LEARN AND HAVE THE OPPORTUNITY TO ACT ON THIS FEEDBACK IMMEDIATELY

FURTHER INFORMATION

- Inside the Black Box: **helloworld.cc/2tZXFsy**
- Diagnostic Questions: **diagnosticquestions.com**
- Project Quantum: **helloworld.cc/2jAJXL3**
- No more marking: **www.nomoremarking.com**

Assessment for Learning is designed to serve the students' learning - not to certify that they have achieved a set standard.



WILL A COMPUTER TAKE YOUR JOB?

As self-driving cars and blockchain-based smart contracts become a reality, what will be the impact of artificial intelligence on our working lives?

Oxford academics Frey and Osborne set out to quantify this complex situation in their seminal paper *The Future of Employment*. They found that although there has often been concern from workers about the impact of technology, historically, across the labour market as a whole, people have adapted. In the nineteenth century, mechanisation led to deskilling as the tasks of artisans were moved to factories. However, increased productivity brought the development of new jobs, and workers adapted to the new skills required. In the twentieth century, many manufacturing jobs were automated by increasingly capable technology, but the boom in cognitive work involving information created new opportunities for workers. People have been very good at adapting, learning new skills, and finding new opportunities.

New technologies

The past alone can't predict the future, so Frey and Osborne studied the latest developments in artificial intelligence, machine learning, and mobile robotics to find out what these new technologies are suited to. They categorised jobs on a scale of manual (physical tasks) to cognitive (thinking tasks), and routine to non-routine. In the past, it was routine manual jobs that became automated, and new opportunities have been created in non-routine manual and cognitive tasks.

New technologies are moving in surprising directions, though. Increasingly clever sensors and nimble robotics, coupled with artificial intelligence, are moving into non-routine manual tasks in manufacturing. Routine cognitive tasks such as searching documents and assessing fraud are already being automated, with even surprisingly complex tasks such as the work of junior solicitors already being replaced by computation.

The researchers then used this understanding of technological capabilities and types of task to assess which of today's jobs are most susceptible to being replaced by automation.

Person or machine?

How many jobs are at risk from these new technologies? The answer, in the USA at least, is quite a shock. 47 percent of jobs were classified as being at risk, including most of the transport industry, office, and administration roles, and the majority of manufacturing and production roles.

The next question to answer is what won't be automated? The bottlenecks for technological development identified in the study are sensing, manipulation, creative intelligence, and social intelligence. In the UK, think tank Nesta has been making the case for the importance of creativity in education to prepare young people for the future this paper depicts.

It's not yet clear what the full implications of this work are for educators, but it is clear that the future holds huge challenges for the young people they work with. Sticking to the tried and tested is unlikely to be the safe bet it used to be, something made very clear by the challenges to junior positions in the traditionally high-status legal profession. As automation continues to expand into the non-routine tasks in the job market, we face a huge test of our creativity as individuals, and as a society, to make sure we can cope with the changes that are coming. (HW)

> “ WE FACE A HUGE TEST OF OUR CREATIVITY AS INDIVIDUALS, AND AS A SOCIETY

THE FUTURE OF EMPLOYMENT: HOW SUSCEPTIBLE ARE JOBS TO COMPUTERISATION?

BY Carl Benedikt Frey and Michael Osborne
URL helloworld.cc/2vtmNMG

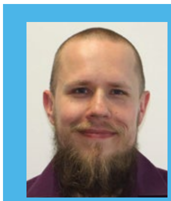
APPROACHES TO ASSESSMENT

Assessment is a thorny topic for many teachers. Here, an international panel of experts discusses innovative approaches and technologies for assessing programming skills

Capability in programming, as in other areas of academic study, requires knowledge and the skill to apply that knowledge. Assessing both requirements at the same time is asking a lot from teachers, but questions, problems and projects might all play a part here. In our cover feature, Cynthia Selby, Mark Guzdial, Barbara Ericson and Michael Ball discuss multiple choice questions, Parsons Problems and autograding Snap! code; Rob Leeman, David Malan, Doug Lloyd, Jesús Moreno-León and Gregorio Robles explore different approaches to assessing projects for GCSE Computer Science, for Harvard's CS50, and in Scratch; and Doug Belshaw and Pete O'Shea consider Open Badges, and how they're used by various organisations.

NEA FOR GCSE

What are exam boards looking for in the non-exam assessment (NEA)? Rob Leeman lifts the lid on the practical element of Computer Science GCSE, and how it is assessed



Rob Leeman
Rob is a product manager for ICT and Computer Science at CIE, and managed the redevelopment of Computer Science at A level and GCSE for OCR.

USEFUL LINKS

Ofqual AO document: **helloworld.cc/2uC7wWU**

Resource Bank blog: **helloworld.cc/2vlM6r7**

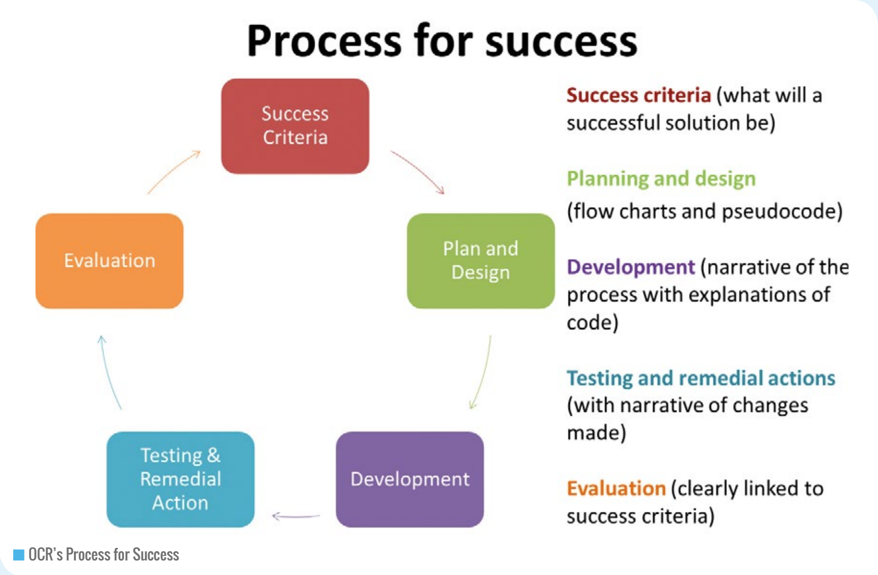
We can't explore how programming is assessed without first looking at the assessment objectives (AOs). The AOs for GCSE Computer Science were designed to allow both practical and theoretical assessment of the subject, and all exam questions and NEA tasks are mapped against these objectives. You can find the official Ofqual AOs on their website (see box). AO3 is concerned with candidates demonstrating their ability to analyse problems in computational terms, and (most importantly for the NEA) to design, program, evaluate and refine solutions.

What makes a good project?

Preparation is key. Candidates should be able confidently to apply the techniques in the specification to solve computational problems in their chosen language. The tasks are mapped against the techniques, so knowledge of their application is essential. This of course can be supported by using cheat sheets and the Resource Bank (see box). The project should be approached in an iterative development cycle, and OCR suggests the use of their Process for Success. This is a simple system development life cycle process, best applied to each component within the task as a whole. Not all candidates are expected to complete all component parts, so this is essentially differentiation by outcome, as stronger candidates will complete most or all of the component tasks. This allows all candidates to access the full range of marks.

One thing to impress upon candidates is the need for thorough planning. Many candidates rush the analysis and design phase, and dive straight into development, but there are lots of marks available in the analysis and design section. These should not be ignored, as a good design makes the implementation much easier. This analysis and design portion is where a candidate's ability to demonstrate their knowledge and understanding of the decomposition and abstraction is assessed. A good project will be well designed and will also consider robustness, functional design, data types, input sanitisation, and testing, and should include graceful degradation.

Even with the best plan in the world, a good project falls over if the implementation is not adequately evidenced. Candidates should be drilled in how to evidence their development work and regularly capture evidence of their code, fully explaining what is occurring and how it links to the designs. This element does not need to be verbose, but must demonstrate the candidates' understanding.



Process for success

Success criteria (what will a successful solution be)

Planning and design (flow charts and pseudocode)

Development (narrative of the process with explanations of code)

Testing and remedial actions (with narrative of changes made)

Evaluation (clearly linked to success criteria)

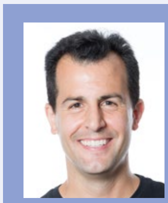
OCR's Process for Success

NEA or no NEA?

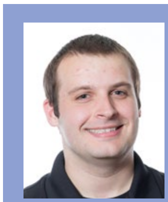
While the NEA seems to cause much anxiety among Computer Science teachers, it remains an important tool in assessing a candidate's ability to apply their knowledge in a practical fashion to a problem that could not be approached in a formal exam setting. This method of assessment, while fraught with regulatory problems and issues of malpractice, is still the closest we can get to how computer science is applied in the real world. It is a fantastic experience for candidates as well as preparing them for A level and further study. The NEA may be daunting, but the alternative is equally problematic. Hopefully after the first series we will see the true value of it, and appreciate it as part of GCSE assessment. (HW)

COMPREHENSIVE PROGRAMMING ASSESSMENT

Determining whether a program works needn't be the only goal of assessment



David J. Malan
David is Gordon McKay Professor of the Practice of Computer Science at the Harvard John A. Paulson School of Engineering and Applied Sciences. He is the instructor for CS50.



Doug Lloyd
Doug is Senior Preceptor in Computer Science at the Harvard University Division of Continuing Education. He is the course manager for CS50.

Determining whether a student's code is correct is an important consideration. Indeed, whether one uses an autograder, or benchmarks student code through a series of manual unit tests, it's fairly easy to determine whether a student has considered the solutions to your test cases in their solution. While correctness is important, there remain other questions to consider.
For example:

- Did the student at least attempt to solve the program at hand, even if they struggled along the way with syntax, such that their code might not compile? Particularly among those less comfortable, rewarding effort reminds students that it's normal and OK to struggle.
- Is it actually possible to read the student's code? Poor indentation (unless programming in Python!) and no commenting might not matter to the processor, but to an instructor it can cause quite the headache!
- How efficient is the student's code? If the program is correct, but takes 15 minutes to run, is it all that useful?

At CS50, we grade the answers to all those questions on the axes of scope, style, and design, respectively; in addition to correctness. It is design, however, that we'll discuss further here.

What to look for

If correctness asks the question 'Does it work?' then design asks the question 'How does it work?' Our goal in assessing design is to get students thinking critically about the decisions they are making in their code, and to put them into a mindset of continuous improvement. We rarely give out a 'perfect' score in design at any point through the school year, as there is almost always some area where improvement can be made. We ourselves are in a mindset of continuous improvement, and we feel that if we can find opportunities to better design our own staff solutions to our problem sets (and year after year, we almost always do), then our students should be able to as well.

CS50 staff grading student work at a grading party

Image credit: CS50, cs50.harvard.edu.

Students get help from CS50 staff members during office hours

Some questions we consider when looking at design:

- How frequently do the same lines of code repeat?
- Does a student's code have an over-reliance on loops or 'magic numbers'?
- Did the student choose the most efficient algorithm?
- Was the code broken into functions or subroutines where appropriate?
- How many lines of code did the student write? The length of a student's submission is often, though not always, a good proxy for how well-designed that code is: if fewer lines of code are needed, the student is probably taking advantage of an efficiency.

Trade-offs

Unlike the other three axes – correctness, which can be assessed with autograders; style, which can be assessed with linters; and scope, which can be assessed by a quick glance (see box, right) – grading design can take significant time and effort, for experienced and inexperienced teachers alike. Students in CS50 submit their code to course staff via GitHub, and teaching fellows can offer feedback via comments on GitHub's web interface. The ability to use 'saved replies' for issues that might recur across multiple students can save some time over writing the same comment out again, and grading design only becomes quicker and easier once one has seen multiple different solutions to the same problem, and is familiar with common errors.

That said, the qualitative feedback received via the design axis is, in our opinion, the most valuable. Indeed, comments on design are intended to get students thinking about what makes one solution better (or worse) than another, even if both have the same output. This reinforces the notion that there is no 'one right answer' in solutions to problems, and hopefully encourages students to be more cognisant of their programming decision-making. (HW)

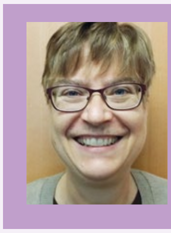
A MULTI-AXIS GRADING PHILOSOPHY

Assessing programming through different axes allows us to test several skills via a single assignment.

- Scope rewards student effort in solving problems, reiterating the importance of a 'culture of error', and normalising the idea that failure to completely solve a problem is not a total failure, but rather still a learning opportunity. (Did you try?)
- Correctness judges the performance of a student's code, either using a test harness or via manual execution of the code. (Does it work?)
- Design considers the efficiency, elegance, and clarity of a student's code from an organisational standpoint. (How does it work?)
- Style is the most human-focused axis, which considers how readable and well-commented the code is for others who might need to read it. (How does it look?)

MULTIPLE CHOICE

Can you really use multiple-choice questions to assess programming and computational thinking?



Cynthia Selby
Cynthia is Senior Lecturer and PGCE Computer Science Tutor, University of Southampton, and Member of the CAS Project Quantum Content Group.

Do teachers ask questions just for fun? Or do they ask questions to find out what their students know and do not know? Multiple-choice questions (MCQs) offer the advantage of being quick to mark, and have the potential for gamification. How can MCQs can be used to assess programming concepts and computational thinking?

Question design

All MCQs should be based on sensible design criteria. You can find lots of advice, based on very good research, about designing MCQs. They have been used for summative assessment in the context of medicine for many years. However, teachers tend to use them more formatively: to identify what needs to be taught or reviewed, and to help students improve. The box on the left lists some top tips for MCQ writing.

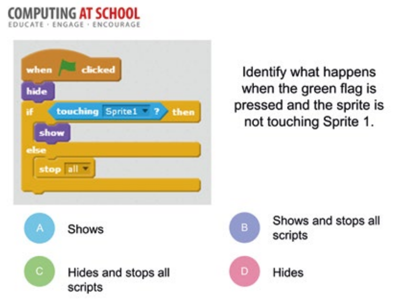
TOP TIPS FOR MCQ DESIGN

- Assess one, and only one, objective
- Provide four response options
- Provide one, and only one, clearly correct response
- Do not use 'none of the above' or 'all of the above'
- Use distractors based on misconceptions and common errors
- Make all distractors plausible
- Give necessary and clear context first, if required; then, separately, ask the question
- Use images as appropriate to support context, or as alternative responses
- Short sentences are easier to understand
- Make all response options grammatically parallel
- Make all response options of a similar length

Distractors and challenges

The importance of good distractors should not be underestimated. Trying to predict what a student does not know can be challenging. When choosing the alternative incorrect responses to an MCQ, consider the types of misconceptions you have identified among your students. For example, if a student uses a box analogy for a variable, they may believe it can hold two values at the same time. A distractor that expresses that understanding would help identify students who need additional help before moving on. You should also consider common errors in processes. A student with a misconception about indexing may always be out by one, so a response incorporating this misunderstanding would also make a good distractor.

It's not difficult to write an MCQ that assesses the students' ability to recall a fact. It's much more challenging, but not impossible, to write MCQs that assess application, analysis, or creation. With application, a question could be confined to a single step in a multi-step processes. Analysis can be assessed by asking 'what if' type questions where students predict what will happen next. An even more challenging assessment of analytical skills is to define a requirement, such as an output, and ask the student to select which program solution meets that requirement. Certainly, MCQs don't lend themselves to the creation of program solutions. However, they can be used in ordering exercises, where students must choose the correct sequence of instructions to meet a requirement. Slightly simpler would be asking a question that just required the identification of a single instruction to fill a gap.



COMPUTING AT SCHOOL
EDUCATE · ENGAGE · ENCOURAGE

```
when [flag] clicked
hide
if <touching [Sprite1 ▼] ?> then
    show
else
    stop [all ▼]
```

Identify what happens when the green flag is pressed and the sprite is not touching Sprite 1.

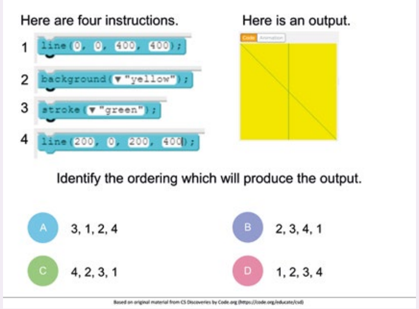
A Shows

B Shows and stops all scripts

C Hides and stops all scripts

D Hides

This question demonstrates assessment of tracing and predicting



Here are four instructions.

```
1 line(0, 0, 400, 400);
2 background(▼ "yellow");
3 stroke(▼ "green");
4 line(200, 0, 200, 400);
```

Here is an output.

Identify the ordering which will produce the output.

A 3, 1, 2, 4

B 2, 3, 4, 1

C 4, 2, 3, 1

D 1, 2, 3, 4

Based on original material from CS Discoveries by Code.org (https://code.org/educate/csd)

This question shows how ordering can be used to assess the skill of sequencing

Getting started

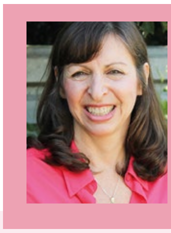
Writing questions with a colleague is an excellent way to get started with writing MCQs. Receiving feedback on questions is the best way to improve them. As time goes on, remember to go back and review your questions. If one distractor is never being selected, you're missing an opportunity to test a true misconception.

One of the easiest and simplest ways to get started with MCQs is to join a group and share your work. Project Quantum is a joint project to crowdsource MCQ computing questions. You can find more information by visiting **helloworld.cc/2eN44Vr**. (HW)

PARSONS PROBLEMS FOR ASSESSMENT AND LEARNING

Parsons Problems are programming puzzles that improve student learning and provide an effective assessment technique

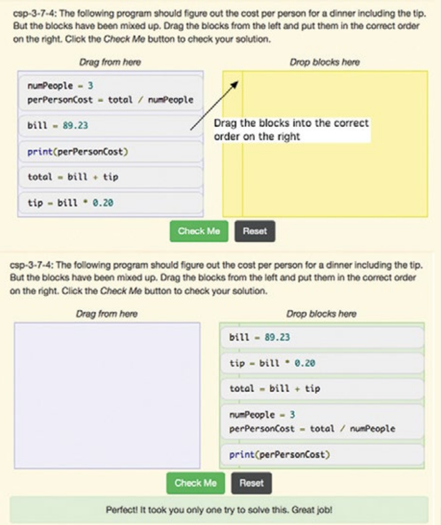
Mark Guzdial
Mark is a professor in the School of Interactive Computing at Georgia Institute of Technology, USA.



Barbara Ericson
Barbara is the Director of CS Outreach for the College of Computing at Georgia Institute of Technology.

Most of us teach programming by asking students to write lots of programs from scratch. It makes sense as a kind of 'learning by doing'. The problem is that programming is hard. A mistake, like forgetting to add a colon, can lead to a student spending hours tracking down the error.

Parsons Problems (invented by Dale Parsons at Otago Polytechnic, New Zealand) give students a programming problem, and give them all the lines of program code to correctly solve the problem, but the lines of code are broken into code blocks and mixed up. Imagine your code presented as refrigerator magnets: the students must place the blocks in the right order, and sometimes have to provide the correct indentation as well.



csp-3-7-4: The following program should figure out the cost per person for a dinner including the tip. But the blocks have been mixed up. Drag the blocks from the left and put them in the correct order on the right. Click the *Check Me* button to check your solution.

```
numPeople = 3
perPersonCost = total / numPeople

bill = 89.23

print(perPersonCost)

total = bill + tip

tip = bill * 0.20
```

Drag the blocks into the correct order on the right

csp-3-7-4: The following program should figure out the cost per person for a dinner including the tip. But the blocks have been mixed up. Drag the blocks from the left and put them in the correct order on the right. Click the *Check Me* button to check your solution.

```
bill = 89.23

tip = bill * 0.20

total = bill + tip

numPeople = 3
perPersonCost = total / numPeople

print(perPersonCost)
```

Perfect! It took you only one try to solve this. Great job!

A Parson's Problem with the correct code broken into code blocks and mixed up (top). A solved Parson's Problem (bottom).

Advantages

Completing Parsons Problems is easier for students than writing the same code from scratch. They never get a syntax error if they only have to order the correct blocks. Our research evidence suggests that Parsons Problems are still challenging for students; students still learn from completing them; and most students enjoy them. We have some evidence that they might be learning as much from solving Parsons problems as if they had written the code from scratch.

A big advantage of Parsons Problems is that they take less time. Students can solve several Parsons Problems in the time it might take them to write one piece of code from scratch. That means that we may be able to give students more of these puzzle-like activities, and get through more learning in less time, than with one code-writing activity.

Assessment

In our research group, we have also explored the use of Parsons Problems as an assessment technique. Imagine that you have two students, one of whom has learned more than the other. If you give both students a programming task on an assessment, you might find that neither student finishes the task. Or one might finish, and the other might not make much progress at all. If we give the students Parsons Problems, it is likely that they will both finish some of the problems. We find that Parsons Problems are a more sensitive assessment of programming knowledge than traditional code-writing problems.

RESOURCES

Try Parsons Problems at **helloworld.cc/2vJfttg** (our free interactive e-book for students) or **helloworld.cc/2uZcFKH** (a companion e-book for teachers). Both books teach introductory programming concepts using Python.

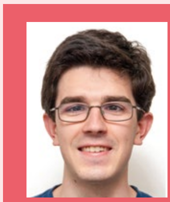
Author your own Parsons Problems or create your own interactive e-book using Runestone Interactive: **helloworld.cc/2uZO3Sd**

You can also create your own Parsons Problems with paper, make magnets out of each block, and ask students to order that code.

Barbara's research is exploring variations of Parsons Problems. In some of her problems, she makes the problem slightly harder by adding blocks which should not be part of the solution. These are called distractors. She is exploring versions of the problems where an undesirable block (the distractor) is visually paired with a block that should be included, which draws the students' attention to exactly the issues that the teacher most wants them to learn. For example, the distractor block could contain a common syntax error. (HW)

AUTOGRADING FOR SNAP!

Mark your students' projects with Lambda, an autograder for Snap! Michael Ball explains how it works



Michael Ball
Michael is a Software Engineer at Gradescope, an online grading platform, as well as a researcher and developer working at UC Berkeley with The Beauty and Joy of Computing. Recently, he earned an MSc in Computer Science Education from UC Berkeley, where he developed the autograder for Snap!

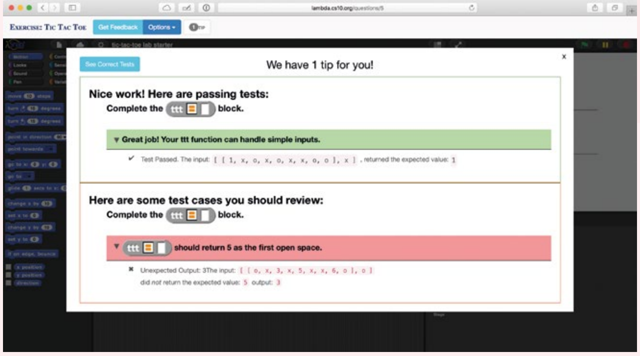
Teaching computer science has long relied on autograders, programs that can execute students' programs and provide feedback and scores, all with minimal work from the instructor. However, block-based languages such as Snap! and Scratch have been largely left out from the development of these programs – something we discovered when, two years ago, we endeavoured to bring our course, *The Beauty and Joy of Computing*, to the online world via **edx.org**. To scale for thousands of students, we experimented with Lambda, an autograder for Snap! Lambda consists of two components: an augmented Snap! environment that runs the autograding programs, and a web server with a database of questions.

How an autograder works

The autograder augments Snap! with scripts (written in JavaScript) that allow authors to execute students' code in the background and run a series of tests to generate feedback. The most common are 'input/output' tests that call a reporter block with some inputs and compare the results to an expected value. More complex tests can input the properties of a sprite, such as its pen colour or position, or the structure of the script itself. For example, when working on a 'draw square' program, we can assert that a student uses a 'repeat' block with an input of '4', to guide them towards a specific solution.

As with all traditional autograders, it is worth mentioning that our approach is not perfect. It is impossible to account for all solution types that students may create, and we must be careful not to stifle each student's own progress. We have tried to develop our autograders to check whether students have attempted a solution, rather than whether they have a perfectly correct solution. For exercises such as sorting a list or generating Fibonacci numbers, it is clear when a solution is correct, and this is where the autograder is most useful.

The second component of Lambda is a website that contains exercises that teachers can assign to students. The website allows students to access and retrieve their submission history for a particular exercise, enabling students to experiment freely.



lambda.cs10.org/questions/5

Exercise: Tic Tac Toe

We have 1 tip for you!

Nice work! Here are passing tests:
Complete the ttt block.

▾ Great job! Your ttt function can handle simple inputs.

✔ Test Passed. The input: [[1, x, o, x, o, x, x, o, o], x] , returned the expected value: 1

Here are some test cases you should review:
Complete the ttt block.

▾ ttt should return 5 as the first open space.

✖ Unexpected Output: 3The input: [[o, x, 3, x, 5, x, x, 6, o], o] did not return the expected value: 5 output: 3

The test results for a simple noughts and crosses game: this shows one passing and one incorrect test

We plan to use the collective submission history to improve the feedback given in future exercises. Teachers can choose to integrate the autograders into their own courses, so that students can automatically receive grades in their school's normal grade book.

USING AUTOGRADERS

Q: Should autograders replace manual review?

A: Probably not, especially with new programmers. You can often use them together to get fast (but generic) feedback, along with more personalised feedback that might take a little longer to return to students.

Q: Are there any projects where the autograder doesn't work well?

A: The more free-form the assignment, the more challenging developing an autograder will be. In particular, assessing creativity is incredibly challenging. Sometimes it's better to leave out the autograder, because fast but inaccurate feedback can be confusing to students.

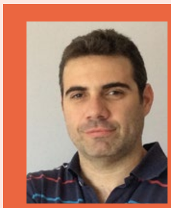
Writing autograders

Today, Lambda represents a beta version of autograding, but teachers who are interested can get started at **lambda.cs10.org**, where you can play with example exercises. However, if you are interested in connecting autograders to your own course or writing your own test cases, please get in touch by writing to **contact@bjc.berkeley.edu**.

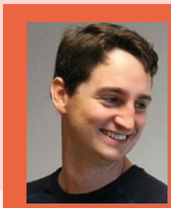
Finally, through this autograding work, we have been developing some Snap! projects that will allow anyone to experiment with writing tests for Snap! blocks directly within Snap! This project can be a starting point for writing more complicated autograders or practising test-driven development. Anyone is free to use the project at **bjc.link/testing-in-snap**. (HW)

ASSESSING SCRATCH

Analysing your students' Scratch projects can be difficult and time-consuming. Why not automate the evaluation of computational thinking skills?



Jesús Moreno-León
Jesús is a former secondary school teacher. He leads Programamos, a Spanish non-profit organisation, while working on his doctoral thesis at Universidad Rey Juan Carlos, Madrid.



Gregorio Robles
Gregorio is an Associate Professor at Universidad Rey Juan Carlos, Madrid. He is active in the field of technology-enhanced learning, and the study and evaluation of computational thinking.

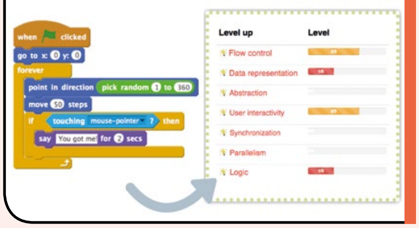
Just as professional software developers use automated tools to help them with their daily tasks, students and teachers who use programming for the development of computational thinking should be able to take advantage of such solutions in the classroom. While this is an area that still requires research, there are already some tools that educators can easily incorporate into their lessons to support their evaluation tasks.

Evidence of computational thinking

The blocks used in a Scratch project, as well as the way these blocks are combined, not only establish the behaviour of the sprites in the project, but can also be used to provide evidence of computational thinking, such as logical reasoning, abstraction, or data representation.

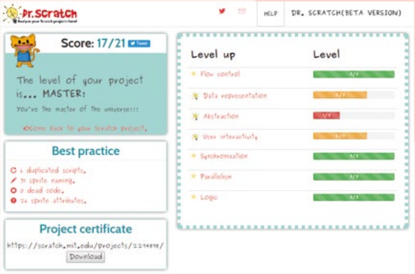
DR. SCRATCH ASSESSMENT SAMPLE

Dr. Scratch computational thinking score: two points for flow control, because it includes a Forever loop; two points for user interactivity, as players interact with the sprite by using the mouse; one point for logical thinking, because of the If statement; and one point for data representation, since the orientation and position properties of the sprite are modified. The other dimensions received no points.



```
when [flag] clicked
go to x: 0 y: 0
forever
    point in direction (pick random 1 to 360)
    move 50 steps
    if <touching mouse-pointer ?> then
        say You got me! for 2 secs
```

Level up	Level
Flow control	2/3
Data representation	1/3
Abstraction	
User interactivity	2/3
Synchronization	
Parallelism	
Logic	1/3



Dr.Scratch

HELP DR. SCRATCH(BETA VERSION)

Score: 17/21

The level of your project is... MASTER!

You're the master of the universe!!!

Come back to your Scratch project.

Best practice

- 1 duplicated scripts.
- 31 sprite naming.
- 0 dead code.
- 24 sprite attributes.

Project certificate

https://scratch.mit.edu/projects/[illegible]/

Download

Level up	Level
Flow control	3/3
Data representation	2/3
Abstraction	1/3
User interactivity	2/3
Synchronization	3/3
Parallelism	3/3
Logic	3/3

Figure 1 The Dr. Scratch feedback report offers an assessment of computational thinking skills, and provides ideas to improve the analysed Scratch project

For instance, let's imagine a project in which every time a decision must be made based on the value of a sensor or variable, the programmer has exclusively used `If` blocks. If this project is compared with another project where the programmer has also used `If else` blocks when required, we would all agree that the second project shows higher levels of logical thinking. In the same way, a project making use of lists would demonstrate a better understanding of data representation than a project using only variables; and a project that includes the use of clones would show higher levels of abstraction than a project in which the programmer has created multiple copies of a sprite using identical scripts.

However, manually inspecting each of our students' Scratch projects can be difficult and time-consuming. Wouldn't it be nice if we could use tools to automate part of the process?

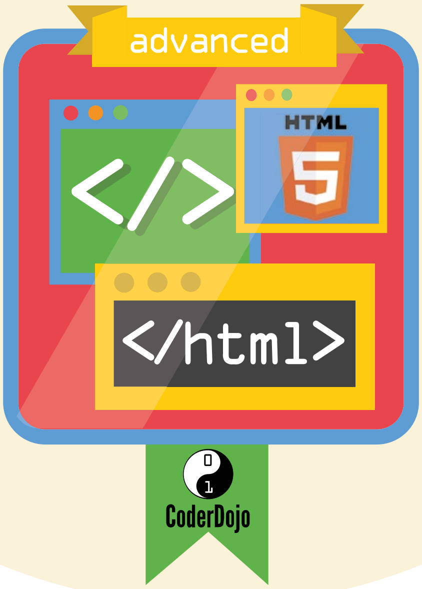
Automating assessment

Dr. Scratch is a free web tool that analyses Scratch projects to assess the development of computational thinking skills. The assessment is based on the degree of development of seven dimensions of this competence: abstraction and problem decomposition, logical thinking, synchronisation, parallelism, algorithmic notions of control flow, user interactivity, and data representation. These dimensions are statically evaluated by inspecting the source code of the analysed project, and given a score from 0 to 3, resulting in a total mastery score that ranges from 0 to 21 when all seven dimensions are aggregated. With this information, the tool generates a feedback report, as shown in Figure 1.

How can teachers use Dr. Scratch in the classroom? The tool could be used to detect students who never use certain instructions in their projects, such as logic operations (`And`, `Or`, `Not`) or flow control structures (different types of loops). Specific tasks could be prepared for those students to help them to learn these concepts.

Limitations and opportunities

Several fundamental aspects of programming, such as debugging or design skills, are not assessed by the tool. Other crucial aspects of the projects, such as originality or creativity, are not taken into account either. Consequently, Dr. Scratch should not be understood as a replacement for evaluators or mentors, but as a supporting tool that automates tedious work, and lets the human instructors focus on the added-value parts. (HW)



ONLINE CREDENTIALING FOR DIGITAL MAKING

Dr Doug Belshaw
Doug is a former teacher and school senior leader who has worked in universities and with the non-profit Mozilla Foundation. He is now a consultant working with a range of organisations around the world.

Learning new programming techniques and approaches can be tough. Open Badges provide a way to scaffold learners' attention, and provide a tangible record of success

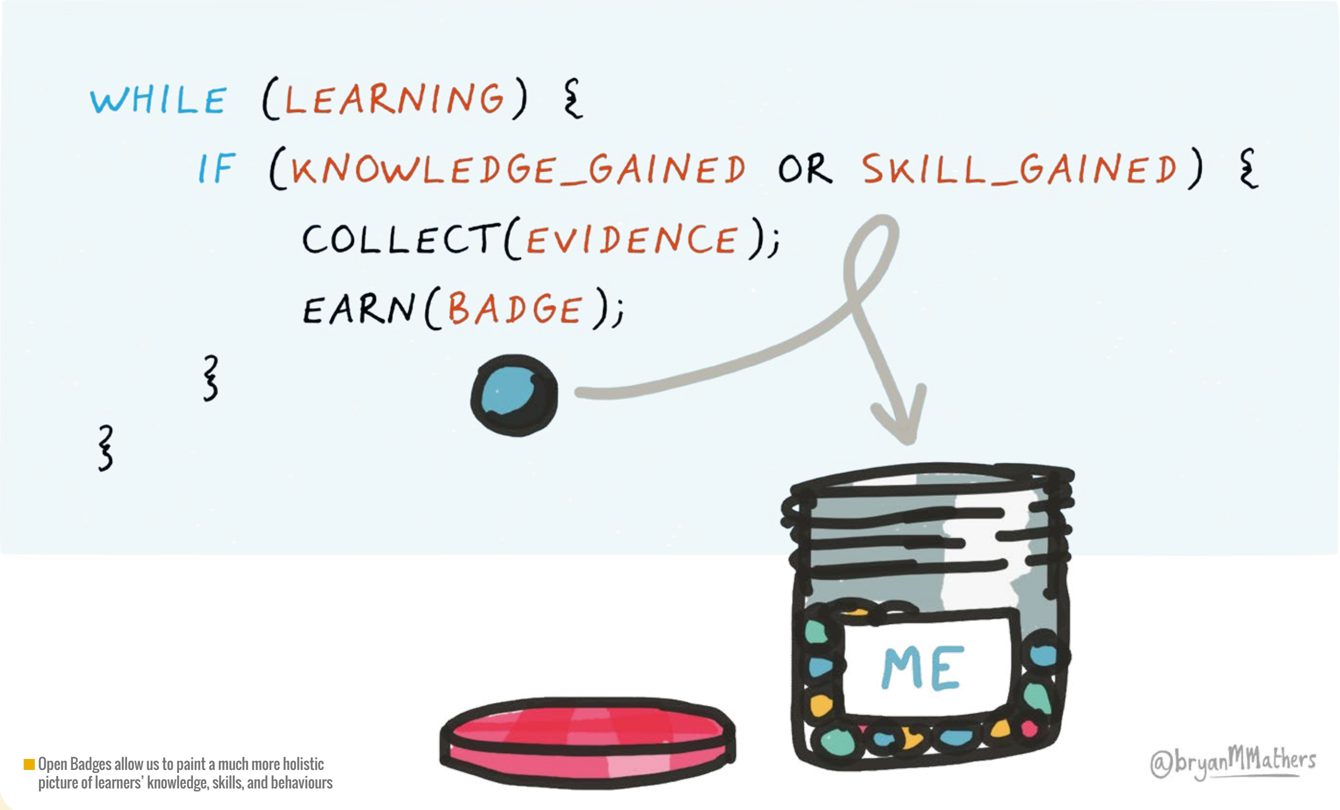
What do Naace, the BBC, and CoderDojo have in common? Other than being supporters of young people getting to grips with coding, they are also issuing new digital credentials called Open Badges. These badges can be issued by anyone for anything - in these examples, they're issued for teacher CPD, learning how to program the micro:bit, and levelling up via increasingly-complex projects, respectively.

Open Badges aren't mere stickers, though. They're digital images full of metadata: information such as the criteria by which they were issued, and links to evidence justifying the award. Organisations as large as City & Guilds, Microsoft, and Salesforce are getting on board with Open Badges, while they're also being issued by volunteers running after-school clubs. Millions of badges have been issued in the last few years to hundreds of thousands of individuals around the world.

Why are badges so useful? Well, employers and educators are finding it increasingly frustrating that credentials gained via high-stakes examinations can do a poor job of preparing people for the diversity and complexity of the modern world. We need ways of capturing students' skills and behaviours, as well as scaffolding the valuable knowledge that grounds true learning.

'Chunky' credentials such as GCSEs, A levels, and degrees make it difficult to peer inside and figure out what an individual knows and can do in practice. In addition, these credentials are often analogue artefacts in an increasingly digital world. We need a solution that works as fast as our digital world changes.

Thanks to Code Kingdoms (**helloworld.cc/2uhp4sO**) and CoderDojo (**helloworld.cc/2tNlanT**) for permission to print their digital badge images



```
WHILE (LEARNING) {
    IF (KNOWLEDGE_GAINED OR SKILL_GAINED) {
        COLLECT(EVIDENCE);
        EARN(BADGE);
    }
}
```

Open Badges allow us to paint a much more holistic picture of learners' knowledge, skills, and behaviours

Image courtesy of Bryan Mathers. Used under a Creative Commons Attribution-NonCommercial licence.

Badge benefits

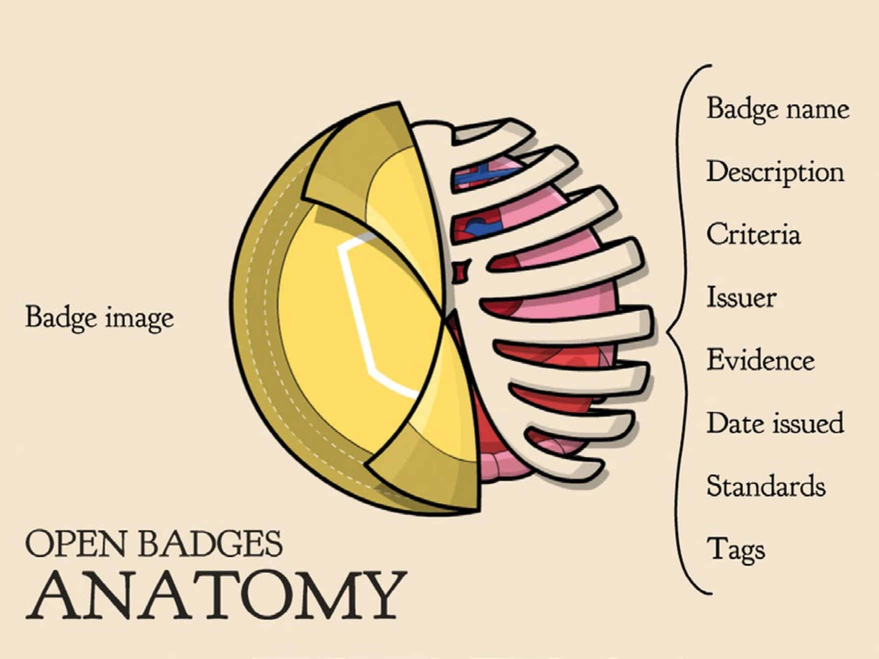
One of the biggest benefits of badges is that they can be evidence-based. Not only can individuals display a credential claiming mastery of a particular area, but they can prove it by making the badge point to relevant evidence. This can be anything that works on the web: a screencast, a CodePen demo, or a working app.

Another advantage is that Open Badges can be issued for anything. You can award a badge for participation, for exceptional effort, for joining a club – whatever you and the recipient deem valuable. For more introverted students, it's a great way to help them tell their stories of success.

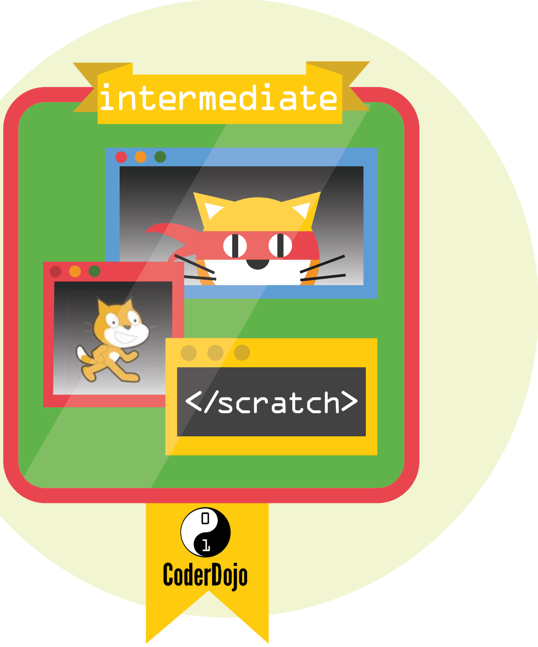
Among the many positives for students using badges is the idea that they can be linked together to form a 'currency' that lasts from their very earliest years, right the way through to adulthood. Although you may choose to call them digital credentials, medallions, stars, or whatever seems appropriate at the time, the Open Badges specification underpins all these schemes, ensuring that they are interoperable and standards-based for the years to come.

OPEN BADGES

The Open Badges specification was developed by Mozilla in 2011 with support from the MacArthur Foundation. Since then, millions of badges have been issued to hundreds of thousands of people around the world, in both formal and informal contexts. The specification is now stewarded by IMS Global Learning Consortium. To find out more visit openbadges.org.



Badge Anatomy by Class Hack. Creative a license CC BY-SA.



> TAKE A LINEAR, SEQUENTIAL APPROACH, OR MIX THINGS UP TO MAKE LEARNING MORE FUN AND ECLECTIC!

LINKS

Open Badge Academy:	helloworld.cc/2vqztkl
Code Kingdoms:	helloworld.cc/2uhp4s0
CoderDojo:	helloworld.cc/2tNlanT
Digital badge FAQ:	computingfaqs.net
Moodle:	helloworld.cc/2vr9zx0

How do I get started?

There are three main ways:

- Build your own solution
- Install a plug-in for a popular open-source platform
- Use an established badge provider

Most people choose the second or third options. There are plug-ins for all the major systems you may use, including WordPress, Moodle, and Drupal. Meanwhile, there is a list of off-the-shelf issuing platforms on the **openbadges.org** website.

Teachers of secondary computing may be interested to know that Computing At School's Progression Pathways have associated badges based on learning statements that are compatible with 'life after levels'. It's up to you whether you want to take a linear, sequential approach, or mix things up to make learning more fun and eclectic! The important thing is that the metadata contained in each badge evidences the progress made towards each learning statement.

There are plenty of ways that teachers, parents, and other interested parties can get involved in capturing learning wherever it happens. Why not get started and issue your first badge today? (HW)

CASE STUDY

CODERDOJO

CoderDojos are informal after-school club environments centred around encouraging and inspiring self-directed learning and peer-to-peer learning in young people, by giving them the tools and supports they need to learn and collaborate. Rewarding and acknowledging young people for their participation and learning at CoderDojo is a core part of our ethos.

Digital recognition

We noticed that young people were learning a whole host of new skills at their Dojos, and in many cases these skills were not being recognised or rewarded. To address this we developed our own Open Badges to acknowledge and reward young people for the new skills (both digital and soft) that they are learning at Dojos around the world.

> STUDENTS WHO PERCEIVE THEMSELVES AS BEING IN CONTROL OF THEIR LEARNING OFTEN DEVELOP GREATER CONFIDENCE

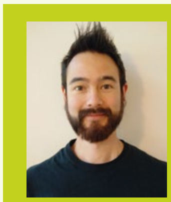
Open Badges are like Scout badges, except digital! Young people participate and learn at their local Dojo. They are awarded a specific Open Badge once they have achieved a set list of tasks that accurately represents a specific skill.

Students who perceive themselves as being in control of their learning often develop greater confidence and increased self-esteem. Open Badges can be a useful motivational tool to support this process, by encouraging young people towards self-directed learning, starting new projects and learning new languages in order to earn more badges.

Part of a community

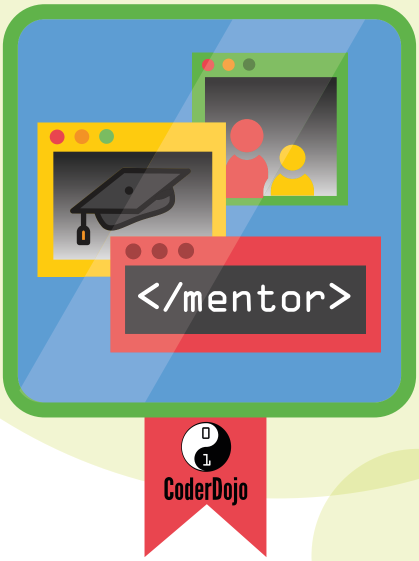
Statistics from our 2016 Annual Survey have shown that 47 percent of Dojos globally are using a method of reward and recognition. Dojos are currently using methods such as end of term certificates, presenting coloured USB belts for certain levels of achievement, or the use of either digital or physical badges to assist with recognising the achievements of young people, while keeping them motivated and engaged.

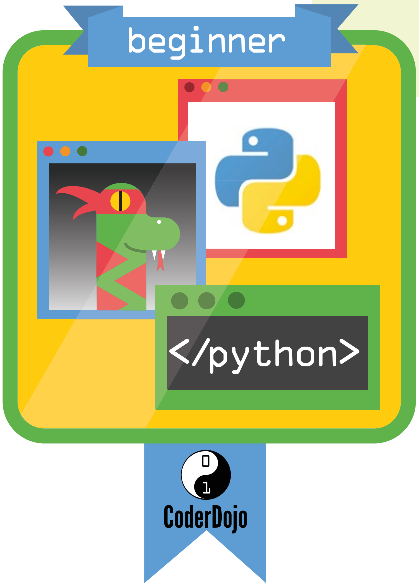
Awarding ninjas (youth attendees) badges for their achievements in a Dojo allows them to build a digital portfolio, showcasing not only their abilities, but the new skills that they have learnt at their local Dojo. The end goal is for these portfolios to assist them in the future: to gain work experience, in job applications, and for college and university applications. In addition, badging gives community members a sense of connection and membership with other members of their Dojo, and in the wider CoderDojo movement, through cultivating a culture of reward and recognition. (HW)

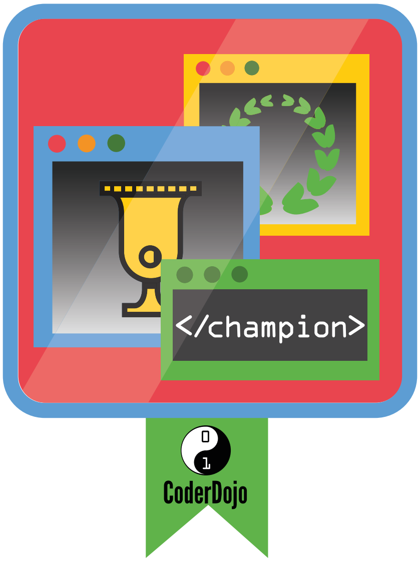


Pete O'Shea

Pete is in charge of Community EMEA and events for the CoderDojo Foundation. He explains the benefits of digital badges for the CoderDojo community







SUBSCRIBE

Sign up today for a year - prices start at FREE!

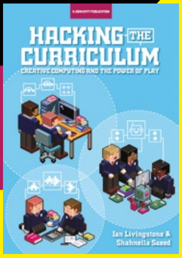
WELCOME TO THE NEW MAGAZINE FOR COMPUTING & DIGITAL MAKING EDUCATORS
(Hello World)
Issue 3
Autumn Term 2017
helloworld.cc
10 TIPS TO SUPPORT INDEPENDENT LEARNERS
Keep your independent students engaged and motivated with our expert advice
BACK TO BASIC
Using BASIC and GOTO to teach the foundations of programming
BUILD PONG IN SCRATCH
A lesson plan for 11-year-olds
APPROACHING ASSES
Our international panel of e and technologies for asses
EMBRACING
How play could make you a b
CODE CLUB FOR TEENS
New resources for 9- to 13-year-olds
OPEN-SOURCE RESOURCES
computing has benefited from open-source. Can education too?
PLUGGED • MATHEMATICAL DOES A PROGRAMMER DO? • SCHOOL LEADERS MATTER
PLUS
LEARNING THROUGH MUSINGS • YOUTUBE CLA GETTING GIRLS CODING • AP

FREE IN PRINT

For UK-based educators!

FREE BOOK

WORTH **£15**


HACKING THE CURRICULUM
CREATIVE COMPUTING AND THE POWER OF PLAY
Ian Livingstone & Shahneila Saeed

Sent to all UK-based subscribers

Subscribe Today

- Get three term-time issues
- Have them delivered directly to your door
- Hello World is not available in stores!

(Hello World)

HOW TO SUBSCRIBE

- **Visit us online:**
  helloworld.cc/sub1year
- **Call our subscriptions hotline:**
  +44(0)1202 586848

Not a UK-based educator?

- Buy any issue for £6

Visit: helloworld.cc/buyissue

- Subscribe from £15 for 3 issues

Visit: helloworld.cc/sub1year

SUBSCRIPTION FORM

YES, I'd like to subscribe to **Hello World magazine**!

This subscription is: □ For me □ A gift for someone

HW#3

YOUR DETAILS □ Mr □ Mrs □ Miss □ Ms

First name .... Surname ....

Address ....

....

Postcode .... Email ....

Daytime phone .... Mobile ....

Are you a UK-based educator? □ Yes □ No

GIFT RECIPIENT'S DETAILS ONLY □ Mr □ Mrs □ Miss □ Ms

First name .... Surname ....

Address ....

....

....

Postcode .... Email ....

Is the recipient a UK-based educator? □ Yes □ No

If you're giving Hello World as a gift, please complete both your own details (left) and the recipient's (right).

Your Hello World subscription will start from issue #3, unless you specify the issue number here: ....

RETURN THIS FORM TO: Hello World Subscriptions, Select Publisher Services Ltd, PO Box 6337, Bournemouth BH1 9EH. *We'll be in touch!*

□ Please tick this box if you DO NOT want to subscribe to the Raspberry Pi Foundation Education newsletter (no spam).

PROGRAMMING AND PLAY

How embracing play made me a better educator

STORY BY Linda Liukas

Inspiration for becoming a better technology educator can be found in unlikely places. What if Roald Dahl had been a programmer? And what does a tiny Italian town have to do with programming pedagogy?

When I first started writing storybooks about programming, I knew almost nothing about pedagogy. I enjoyed programming, but I mixed Piaget with Papert and didn't differentiate between computational thinking and constructivism. I just had a strong sense of the kind of world I'd like to create.

For me, computing was magical, charming and imaginative - but the materials teaching it were often dull and uninspiring.

Programming as make-believe

Most of my childhood was spent in a very immersive world of make-believe. My siblings and I built small villages in the woods for Moomins, and created galactic maps around Star Wars heroes. On the asphalt in our front yard we sailed in a self-made raft and imagined a mysterious sea around us.

This is the way I relate to programming, even today. Being able to build ever more complicated worlds and structures without the need for physical components like LEGO bricks is fascinating, especially for a child. Most children at least once in their lives feel very powerless. Someone else comes up with the rules – but not in programming.

How is programming related to play?

When I decided to learn programming back in 2009, using narrative as a learning tool was a natural fit. I was learning a programming language called Ruby, and every time I ran into a word or concept I didn't understand (like 'what is object-oriented programming' or 'garbage collection') I would try to explain the concept as a six-year-old girl called Ruby would explain it. This project eventually turned into a series of books explaining and celebrating computing from the tiniest Booleans to immense algorithms.

Luckily, on my journey to writing about computing in early childhood, I quickly stumbled upon the work of Seymour Papert and Alan Kay. Even Alan Turing wrote a whimsical note in an early artificial intelligence paper on how we should 'teach' computers like children, teaching them to learn to learn. I realised creativity was something that was built into computing education, but that somehow we had lost it.

Whenever I asked educators about play and programming, they would direct me to apps that gamified learning. Programming education was experienced through challenges, collecting points, or winning competitions. But the type of programming I really enjoyed was full of other types of play: finding and giving support, exploration, and the joy of finding a new way of solving a problem. To rediscover playfulness in programming I needed to visit a small town in Italy.

CREATIVITY IN PROGRAMMING

- **Decomposition and logical thinking**
  As Ruby says, "Even the biggest problems in the world are just tiny problems stuck together". Every programmer starts by breaking down the problem at hand.

- **Creativity and collaboration**
  Even though the instructions a programmer gives to a computer need to be exact, in the right sequence, and carefully named, programming is also highly creative. Try with a friend to instruct each other on how to brush your teeth, and see how many different ways there are to give the commands!

- **Debugging and persistence**
  Learning to program is all about learning to overcome mistakes. Even the best programmers forget a semicolon from time to time, and need to go back and find the mistake.

> "THE COMPUTER IS LIKE A FOREIGNER, AND IF YOU WANT TO TALK TO IT, YOU HAVE TO SPEAK ITS LANGUAGE."
>
> "YES, BUT THE COMPUTER HAS TO UNDERSTAND HOW WE TALK, TOO, AND IT HAS TO DO WHAT WE WANT IT TO DO."
>
> THE HUNDRED LANGUAGES OF CHILDREN: THE REGGIO EMILIA EXPERIENCE IN TRANSFORMATION

Building a computer out of paper and getting to know the components inside it. Download this free exercise at **helloruby.com/play**

The hundred languages of children

In an Italian town called Reggio Emilia I finally found the framework of thinking I needed to create Ruby's world. Reggio Emilia is an educational approach for preschool and primary education, named after the town where it began. From the outside it has very little to do with computing: the approach highlights respect, responsibility and community through artistic exploration and discovery.

The first thing I learned to love from Reggio was the idea of a hundred languages. The core idea of Reggio is that a child has hundreds of ways of expressing themselves: with clay, gestures, paint, and rubber stamps. However, in schools we often limit children to writing and reading. Reggio educators treat the computer as just one more material to learn alongside paper, ruler, pens and movement. One of the hundred languages.

The second thing I fell in love with in Reggio Emilia was the open-ended nature of projects that can take all sorts of twists and turns. Many of my own favourite exercises start with kids posing questions that interest them like "What kind of a computer would a dolphin doctor need?", "What is the world's most dangerous animal?", or "What if my paper computer could print candy?" Throughout the process of exploring and experimenting they learn about abstraction, collaboration, and media literacy, and they develop a plethora of powerful ideas I would never have anticipated. That's why most of the exercises I create for kids include discussion points, and very few of them have right or wrong answers. I think it is important to give kids permission to trust themselves and allow for many right answers to a question.

> I WISH TO SEE PROGRAMMING BECOME ONE TOOL IN A BIG BOX OF SELF-EXPRESSION

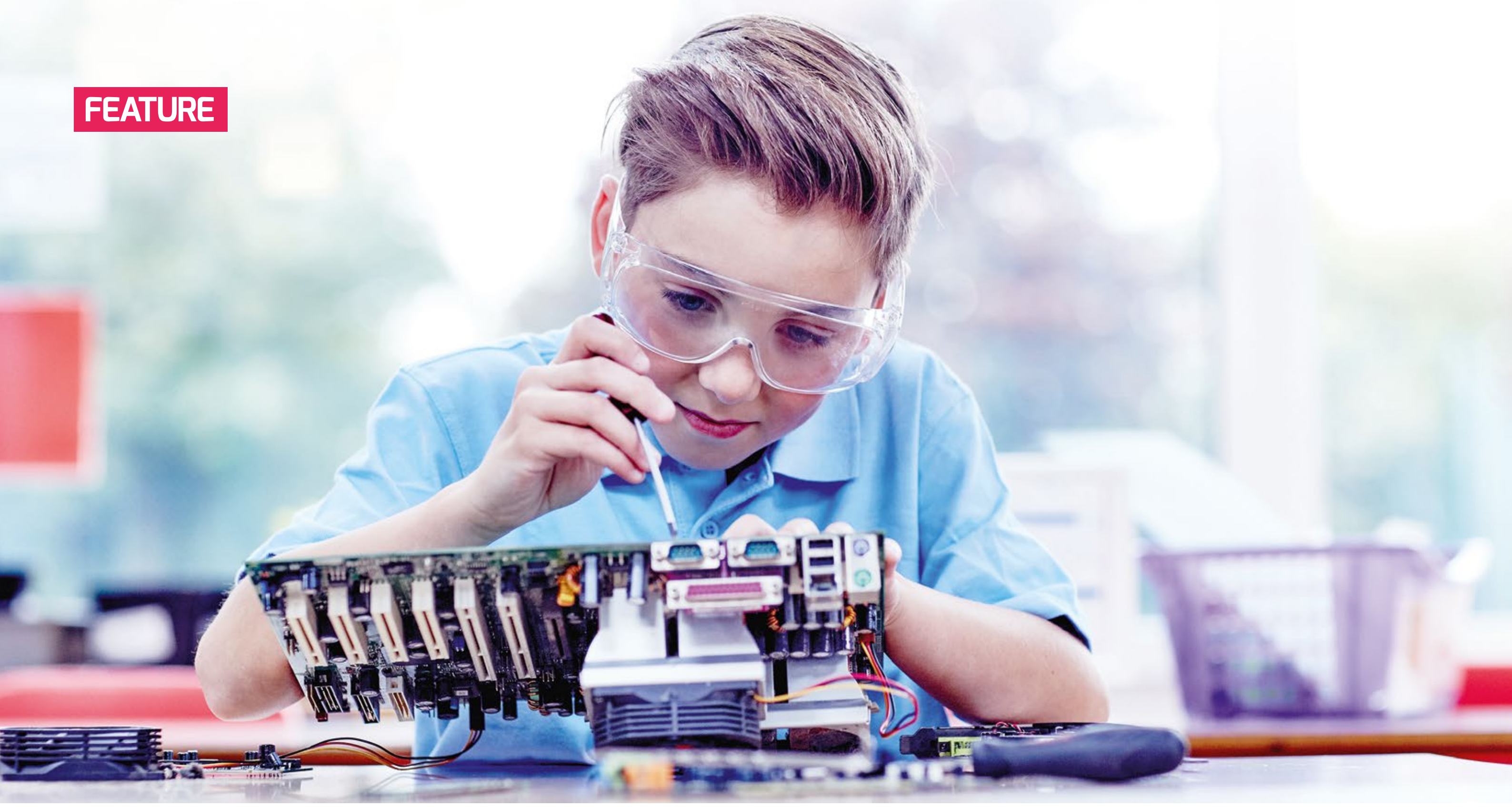
The third thing that really resonated with me is the idea of observing children at work, and responding to their unique needs. I've learned to simplify my writing, creating exercises and materials that have only a single concept to teach. Reggio also suggests that we shouldn't use words as shortcuts to knowledge. Computer science is riddled with abstract words like functions, Booleans and decomposition. But what does a loop feel like? And can we find conditionals in the everyday surroundings of kids, such as the way they choose clothes for a rainy or sunny day?

Computational thinking concepts are more fascinating when we notice their presence all around us. Inspired by Reggio, I've practiced making computer science concrete, specific, and understandable to the child. A computer can take a thousand forms.

I wish to see programming become one tool in a big box of self-expression – along with crayons, blocks of wood, prisms, and pipettes. This can help us to present a more colourful, exciting computing culture. Why does Reggio keep inspiring me after 70 years of existence? I think the answer lies in wonder. These pedagogical movements have helped me to rediscover my own wonder around technology. It is this wonder that allows me to invent new teaching practices that offer unusual and beautiful pathways to computing. (HW)

FURTHER READING

- Hello Ruby: **www.helloruby.com**
- The Hundred Languages of Children: **helloworld.cc/2tiQVXk**
- Understanding the Reggio Approach: Early Years Education in Practice: **helloworld.cc/2syy79u**



BAREFOOT CLASSROOMS

Five ways Barefoot is helping educators

Last issue we reported that the Barefoot Computing Project had reached one million primary school children in the UK. We asked four leading education figures what's driving this success.

It's easy for teachers to follow

"Barefoot is very intuitive and attractive," offers Katy Potts, Islington Council's computing and e-safety lead for children's services. "So far, Barefoot's resources have been really well received throughout all our schools. It's an easy resource to implement – we haven't had any difficulty at all in our schools."

Joy Christie, acting Deputy Head Teacher at Inverbrothock Primary School in Arbroath, Scotland, agrees that Barefoot is easy to use. "The instructions and all the lesson plans go step by step. Anybody could work with it and feel that they were beginning to deliver something."

Barefoot offers free training and resources to primary school teachers in the UK

You don't need any technology to use it

Barefoot provides resources that make it easy for teachers to get started on a journey to teach computational thinking. The resources have been designed specifically to be accessible to teachers who are otherwise unfamiliar with computing and technology.

Irene Bell is Head of STEM at Stranmillis University College in Northern Ireland. The fact that she has a huge amount of experience in computing might suggest that she would lean towards some of Barefoot's more complex resources, but she advocates the project's hugely accessible unplugged activities, which don't require the use of any technology.

"If you're not confident in any of the coding languages, then this is an activity that will be directly relevant to what you're teaching in the classroom." Irene, who trains new teachers, adds: "You won't have to step out of your comfort zone."

To demonstrate her thinking, Irene picks out Barefoot's *Tut, Clap or Jive* activity, which encourages pupils to simply use their bodies to begin their computational thinking journeys.

Barefoot is easy to understand

The theme of accessibility continued with another interviewee. "One of the most popular resources is *Crazy Characters*," offers Dr Jonathan Chippindall, a primary teacher in north Manchester. "It demystifies some computing terminology, such as the word 'algorithm', in a way that's really easy to engage with," he adds.

Barefoot's resources and workshops are all completely free of charge, which helps to explain why large numbers of primary schools are getting behind them. But none of those we spoke with – teachers and education academics – even mentioned the fact that schools were not dipping into their sparse budgets to access Barefoot's resources.

> THE INSTRUCTIONS ARE REALLY EASY TO UNDERSTAND, NO MATTER HOW LITTLE YOU KNOW ABOUT COMPUTER SCIENCE

One of the most popular Barefoot resources is the Crazy Characters activity

Barefoot addresses a global challenge

"The changes to the national curriculum, from ICT to computing, are incredibly important, because the jobs of tomorrow are going to rely more heavily on understanding computer science," adds Jonathan Chippindall. "You only have to look at things like driverless cars, delivery drones, and jobs such as working in supermarkets on checkouts. These roles are being replaced by computers, and the skills of tomorrow are going to be coding. The Barefoot resources enable teachers to deliver that part of the curriculum to pupils."

Barefoot resources introduce computational thinking skills to primary school pupils

Great cross-curricular benefits

"Barefoot stands out as being very accessible to teachers," adds Joy Christie. "The instructions are really easy to understand, no matter how little you know about computer science. It also gives a range of lesson plans that don't just cover computer science. For example, in social subjects, you can use the *World Map Logic* activity. There's so much that teachers can see that they can apply to different areas of the curriculum."

Accessibility and flexibility

One overarching theme has emerged: Barefoot's accessibility. Without fail, everyone we talked to indicated that it is Barefoot's ease of use that makes all the difference.

Barefoot's training and resources may well be free and, of course, convenient. Undoubtedly this acts as an incentive for cash- and time-strapped schools. However, the evidence suggests that it is overwhelmingly the resources' accessibility and flexibility that has really appealed to the 40,000 primary school teachers now confident in teaching computational thinking at a primary level. (HW)

FURTHER INFORMATION

To register for free resources and book a free teacher workshop, visit **barefootcas.org.uk**

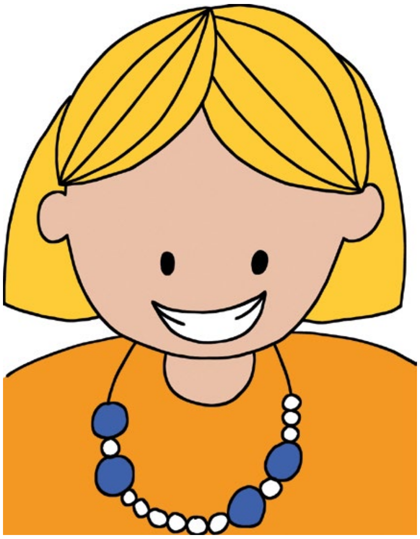
CS UNPLUGGED

The CS Unplugged project is releasing completely revamped resources aimed at teachers. Tim Bell introduces the new format

STORY BY Tim Bell

The CS Unplugged project (**csunplugged.org**) has become widely used in schools for teaching computer science concepts without having to use a computer. We have found that it is popular because it uses familiar materials, and removes the barrier of having to learn programming before students can engage with deep concepts – ranging from binary representation of data to parallel algorithms.

The material was originally developed 25 years ago to support outreach to schools, but now that computer science has become a mainstream subject in many countries, there has been a demand to provide it in a format more suitable for classroom use. The new material is now being released, and the changes include:



One of the integrated activities asks students to create a bead necklace, bracelet, or bag-tag that represents the student's name in binary. Image from csunplugged.org

- Presenting material as lesson plans for teachers
- Explicitly linking the activities to computational thinking concepts
- Providing 'Plugging it in' follow-up programming challenges to enable students to connect the ideas to digital devices
- A more flexible way to print customised worksheets (for example, choosing the level of difficulty, such as the range of numbers used)
- Curriculum integration suggestions, with learning outcomes and follow-up projects touching on areas as diverse as art, music, literacy, and biology

Plugging it in

The lessons have been adjusted based on feedback from classroom teachers, and feature new activities based on recent curriculum changes, as well as classic unplugged activities. The 'Plugging it in' challenges for students range from very simple programs to advanced challenges, so teachers can choose a level to suit their students. All challenges include a working solution, currently primarily in Scratch, but with Python and other languages to be added. The addition of a programming follow-up is based on research that shows the value of integrating unplugged and computer-based learning.

The material is all available under a Creative Commons Attribution-ShareAlike licence, which means that teachers are free to copy and edit it. The source for the new version is available on GitHub. It is built on new software that supports presenting the material online and in print, and enables others to contribute.

The project is supported by funding from Microsoft Philanthropies and Google Inc, and some of the content has been supported by a grant from the New Zealand Ministry of Education.

CS Unplugged can be found at **csunplugged.org**. You'll find links to both the new material (which is being gradually added over the next few months) and the 'classic' version of the site. (HW)

NICHOLAS PROVENZANO TEACHER

GETTING PHYSICAL

The Nerdy Teacher, Nicholas Provenzano, explains how getting physical with computing can inject excitement into the classroom

My introduction to the world of digital making involved working on Python, Scratch, and Minecraft on a Raspberry Pi. These were also the tools I used with students when I introduced them to the Raspberry Pi. After a while, my students were starting to get restless working on screen-based projects. The digital side of computing was not enough for them, and it was not enough for me. I needed to find ways to expand what everyone was doing. It was time to dive into physical computing.

Physical learning

The exciting thing about Raspberry Pis, Arduinos, and micro:bits is that they allow users to explore digital computing alongside physical computing. I noticed that introducing physical computing into the classroom allowed students to explore different aspects of both worlds in ways that were meaningful to them. They were fully engaged because they had a project they had chosen, and they wanted to make it work for everyone to see. Since the students already had the basic digital computing skills, they welcomed the opportunity to jump into more complex physical computing and bring their digital skills to life. This can be true for all learners new to digital and physical computing.

> “ Combining the digital and physical sides of computing takes learning in an exciting new direction

Physical reaction

I helped to facilitate a Picademy event in Ann Arbor, USA. After a day of exploring digital computing and some aspects of physical computing, the educators were given three hours to make a project using their new skills. There were many great projects: using a camera to take pictures and then tweet them; using a light sensor to activate a red LED to let people know that a bathroom was occupied; and a mood-ring-esque device that played music based on the temperature of the person who held the sensor.

The excitement infecting these educators, as they took their digital computing skills and transferred them to making physical projects, was fun to watch. The projects were so varied, and they took risks to see how far they could stretch their knowledge to make their projects work. By doing so, they expanded their skills in a way that was meaningful to them. This is what students need in the classroom.

Physical challenge

Combining the digital and physical sides of computing takes learning in an exciting new direction for students and their teachers. It is important to keep challenging and encouraging students to explore the unknown. Taking their digital making skills and offering them the chance to build and iterate is a great way to allow students take ownership of their learning and direct it where they want to explore – exactly what every teacher hopes to impart to their students. (HW)

Nicholas Provenzano is a high school English teacher, author, speaker, and consultant. He has been featured on CNN.com, Education Week, The New York Times, and other media outlets.

SCHOOL LEADERS MATTER

Support from the school leadership team is crucial for successful computer science teaching. How do you get them on board?

STORY BY Niel McLean

Schools have made significant strides in developing computer science as a subject. The number of students taking GCSE computer science has grown, and primary teachers are reporting significant increases in their confidence to teach computing. However, introducing a new subject is challenging. We have some way to go before computer science is established as a subject in the way that, for example, history or geography are.

The feedback from computer science teachers getting to grips with the curriculum is that a supportive school leadership team can make a significant difference. Where school leaders provide support, the subject flourishes; where they do not, computing teachers face an uphill struggle. It was for this reason that CAS, with support from a Microsoft YouthSpark grant, developed a toolkit for school leaders.

Slow progress

Curriculum development works in different ways in primary and secondary schools. In primary, step-by-step development is possible. Teachers can try a few things, building up the subject as their confidence grows. Secondary schools are different. The need for schools to timetable teachers, students and rooms within a highly structured school curriculum makes a gradual development impossible – the school either offers GCSE Computer Science, or it doesn't.

I was talking about computing with an old colleague, who is now the head teacher of a large London secondary school. "Putting it bluntly," he said, "with all the accountability there is now, why should I take the risk of running a new GCSE with staff who aren't confident they can get good results?" Feedback from CAS Regional Centres also suggests that a 'vicious circle' is developing in some secondary schools – the senior leadership decides not to prioritise computing; computing teachers are not released for training; GCSE is offered only to high achievers as there are not enough confident teachers; the senior leadership think they were right not to prioritise it in the first place; and Computer Science remains a minority option.

Perhaps this is a caricature of the reality in secondary schools, but there is enough feedback from teachers to raise concerns, and we know that support from the school's senior leadership is essential if a subject is to flourish. So how do we convince school leaders that computing is important, and worth the necessary investment of time and scarce resources?

What school leaders think

While researching the CAS Leadership Toolkit, I spoke to a number of secondary school leaders and their representatives about the challenges they face and how they have addressed them. What struck me most was that the head teachers who were most supportive were those who 'got it' – they understood the value of computing for the life chances of the young people in their schools. They stressed the importance of a clear vision

The Leadership Toolkit contains a series of leaflets designed to help teachers and school leaders to provide excellent computer science education

bcs Academy of Computing

EXCELLENT COMPUTING IN EVERY SCHOOL:
A TOOLKIT FOR SCHOOL LEADERS

Produced with support from Microsoft Youthspark

Microsoft | COMPUTING AT SCHOOL | EDUCATE · ENGAGE · ENCOURAGE | Part of BCS, The Chartered Institute for IT

bcs Academy of Computing

LEADING AND MANAGING COMPUTING ACROSS THE SCHOOL

Who this is for:
SLT and computing subject leads

"In these challenging times for school leaders, computing will only succeed if school leadership teams are able to manage its implementation strategically."

Duncan Baldwin
Deputy Director of Policy
Association of School and College Leaders

Microsoft | COMPUTING AT SCHOOL | EDUCATE · ENGAGE · ENCOURAGE | Part of BCS, The Chartered Institute for IT

bcs Academy of Computing

COMPUTING AND THE SCHOOL'S CURRICULUM VISION

Who this is for:
SLT members and governors responsible for the curriculum

"As a computer scientist, I'm aware of the shortage of computer science and IT skills in the economy, and this is something schools should take very seriously."

Dr Saima Rana
Principal
Westminster Academy

Microsoft | COMPUTING AT SCHOOL | EDUCATE · ENGAGE · ENCOURAGE | Part of BCS, The Chartered Institute for IT

bcs Academy of Computing

MONITORING AND EVALUATION

Who this is for:
SLT and computing subject leads

"The introduction of computing is a significant change with implications for staffing, CPD and resourcing. In their strategic role, school leadership teams need to ensure that the change is properly implemented, monitored and evaluated."

Dr Richard Marshall
Principal
Royal Greenwich UTC

Microsoft | COMPUTING AT SCHOOL | EDUCATE · ENGAGE · ENCOURAGE | Part of BCS, The Chartered Institute for IT for computing and how it fits into the school's wider aspirations for all its students. Derek Peaple, Headteacher at Park House School, was clear that "placing and protecting computing at the heart of the curriculum is essential to developing the next generation of creative, innovative, and digitally confident young people."

This suggests that, notwithstanding the pressures, most school leaders are driven by a desire to do what's best for the students in their schools. They are proud to go beyond the merely regulatory requirements. We shouldn't be surprised at this: no head teacher wants an epitaph that reads 'She did everything that the DfE and Ofsted required'. Articulating a compelling vision and purpose for computing is a necessary first step for gathering senior leadership support, and here we have one serious disadvantage – computing is new. While virtually all secondary heads could explain why well-established subjects such as geography are on the curriculum, in specific terms such as 'understanding how places shape people and people in turn shape places', few could do the same for computing.

We also have one serious advantage – it's generally recognised that computing matters to the future of the UK. Dr Saima Rana, Principal of Westminster Academy, agreed with this, bringing in the wider economic context. As a computer scientist herself she is "aware of the shortage of computer science and IT skills in the economy", and recognises the opportunities this creates.

> “ MOST SCHOOL LEADERS ARE DRIVEN BY A DESIRE TO DO WHAT'S BEST FOR THE STUDENTS IN THEIR SCHOOLS

Where school leaders see the importance of computing for their students, they have taken practical steps to support their teachers, often adopting innovative approaches. As Sion Humphreys, a professional adviser to the National Association of Head Teachers, has said: "Effective school leaders are creative people. They thrive on solving problems, and will find creative solutions to the challenges."

While setting a vision and providing visible support are clearly important, the introduction of computing represents a significant change with implications for staffing, CPD and resourcing. These implications need to be managed, and are linked to the school leader's strategic role. As Dr Richard Marshall, Principal of Royal Greenwich UTC, has said: "In their strategic role, school leadership teams need to ensure that the change is properly implemented, monitored and evaluated."

Using the Leadership Toolkit

The Leadership Toolkit is deliberately short – all the school leaders we spoke to stressed how busy they were. It has been sent to all secondary schools in hard copy. We've designed the toolkit as a set of leaflets – again, this is deliberate. We know that there's always the risk that busy school leaders will simply pass it on to an already hard-pressed head of computer science. It includes a short booklet aimed directly at the school leadership team.

If the leadership team has passed the whole toolkit to you, why not take the short booklet (it's called Excellent Computing in Every School) to the senior leader with curriculum responsibility. When they have read it (it takes only a few minutes to read), talk to them about how they could use the other leaflets to help introduce computing in your school. Suggest that a small team is set up that reports directly to the leadership team; offer to raise the awareness of school governors; and stress that this is about improving your students' life chances.

Of course, a few slim leaflets can't change everything, but hopefully they can help support a conversation between you and your school's leadership team that helps computing to flourish in your school. Good luck. (HW)

Academy of Computing

CURRICULUM
PLANNING

Who this is for:
SLT SLT members with curriculum and assessment responsibilities and computing subject leads

"Placing and protecting computing at the heart of the curriculum is essential to developing the next generation of creative, innovative and digitally confident young people. It is essential that computing staff are represented on SLT, and therefore able to directly influence strategic decisions about learning if we are to secure such a vision for the present – and future – of learning."

Academy of Computing

TEACHING, LEARNING AND
ASSESSMENT

Who this is for:
SLT members with responsibility for teaching and learning as well as computing subject leads

"Computing is new and the pedagogy is developing all the time. In their strategic role school leaders should be supporting the development of those new pedagogies and evaluating their impact."

Nevita Pandya
Deputy Head Teacher
Townley Grammar School

Academy of Computing

STAFF
DEVELOPMENT

Who this is for:
SLT members with responsibility for teacher Continuous Professional Development (CPD) and computing subject leads

"Where head teachers are providing computing teachers with time to learn new skills and new subject knowledge the impact of the new curriculum is considerably greater."

Simon Humphreys
National Coordinator
Computing At School

Academy of Computing

INFRASTRUCTURE AND
RESOURCES

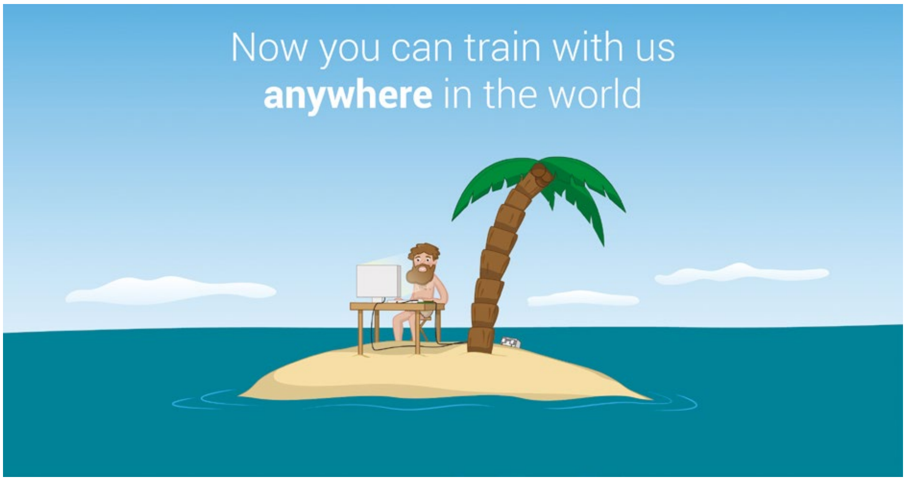
Who this is for:
SLT, network managers, ICT coordinators and computing subject leads

"Effective school leaders are creative people, they thrive on solving problems and will find creative solutions to the challenges."

Sion Humphreys
Professional Adviser
National association of Head Teachers

Microsoft

COMPUTING AT SCHOOL
EDUCATE · ENGAGE · ENCOURAGE
Part of BCS, The Chartered Institute for IT



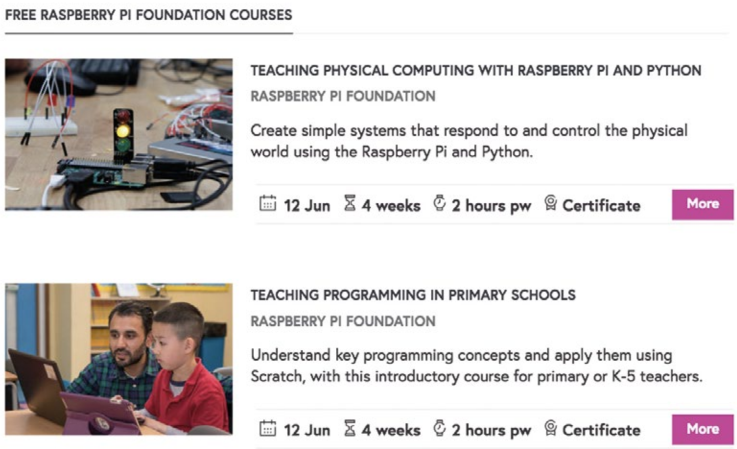
ONLINE TRAINING: THE STORY SO FAR

Six months ago, The Raspberry Pi Foundation launched its online training courses. Lauren Hyams looks at what they've learnt from the experience

STORY BY Lauren Hyams

The Raspberry Pi Foundation has been running Picademy, a successful face-to-face training programme for educators, since 2014. So far, more than 1500 educators around the world have taken part in the programme.

This free face-to-face training has proved hugely popular. On average, we receive three eligible applications for each available place. However, this model of delivery is not without its limitations. After seeing our Picademy attendees getting excited on Twitter, we often get questions like "Why haven't you run a Picademy near me yet? When are you coming to train us?". It has been frustrating, having to tell eager educators that we are not able to deliver Picademy everywhere. We decided to find a way to reach educators around the world, so now you can access our free courses online – wherever you are, and whenever you want.



FREE RASPBERRY PI FOUNDATION COURSES

TEACHING PHYSICAL COMPUTING WITH RASPBERRY PI AND PYTHON
RASPBERRY PI FOUNDATION
Create simple systems that respond to and control the physical world using the Raspberry Pi and Python.
12 Jun 4 weeks 2 hours pw Certificate More

TEACHING PROGRAMMING IN PRIMARY SCHOOLS
RASPBERRY PI FOUNDATION
Understand key programming concepts and apply them using Scratch, with this introductory course for primary or K-5 teachers.
12 Jun 4 weeks 2 hours pw Certificate More

COMING SOON

Following the success of the first two courses, the Raspberry Pi Foundation is delighted to announce the launch of two new courses in the autumn term.

PREPARE TO RUN A CODE CLUB

This course will give you the confidence and skills to launch and run a Code Club. You'll get practical advice and tips from volunteers, teachers, and Raspberry Pi Foundation staff, and explore the resources that Code Club provides.

OBJECT-ORIENTED PROGRAMMING IN PYTHON: CREATE YOUR OWN ADVENTURE GAME

Learn object-oriented programming principles by creating your own text-based adventure game in Python. The course is aimed at educators who have a bit of programming experience, but have never programmed in the object-oriented style. The course is hands-on, giving you a chance to apply your new knowledge by completing a range of programming tasks.

Image: Melissa Huch / Digital Harbor Foundation

More than 1500 educators around the world have taken part in the Picademy programme

Learning together online

In February 2017, we were delighted to launch two free online CPD training courses on the FutureLearn platform, available anywhere in the world. Since the launch, more than 11000 educators have joined our *Teaching Programming in Primary Schools* course, and more than 18000 have joined *Teaching Physical Computing with Raspberry Pi and Python*. Thousands of educators have been building their skills, completing tasks such as writing a program in Python to make an LED blink, or building a voting app in Scratch.

The two courses are scaffolded to build skills each week. Learners are supported by videos, screencasts and articles, and they have the chance to apply what they have learned in as many different practical projects as possible. Visit **helloworld.cc/2uye60k** to join us.

> THE TWO COURSES ARE SCAFFOLDED TO BUILD SKILLS EACH WEEK

Classrooms all over the world

The highlight of the courses for me has been watching the rich discussions between learners from more than 120 countries. More than 800 learners on the Physical Computing course had a great discussion around the concept of learning through failure. This is an idea we have been talking about a lot in our team, and it was amazing to expand the conversation to include hundreds of educators all over the globe. Hearing educators' thoughts on how it felt to make something and what they learnt from it has been amazing – from beekeeping and building flat-pack furniture, to various triumphs and disasters in the kitchen!

One educator, Jordan Renault, made a thought-provoking observation. He pointed out that when we were infants learning to walk, our attempts were celebrated, and the falls were seen as a necessary product of learning. We should try to not lose this outlook as we grow older.

Learners on the courses have been very positive about the experience. They have appreciated the way the content is structured, building the difficulty each week as skills and confidence grow. It has been wonderful to see learners using the courses with their students in class. One learner commented, "The opportunity to learn from and discuss ideas and thoughts with my peers was invaluable – I have so many ideas to implement Scratch in my classroom next academic year, thanks to the ideas submitted".

EDUCATOR IN FOCUS

MARTIN CADDY

Headteacher,
Stokeinteignhead Primary School

As the Headteacher of a small village primary school, I cover the time when the class teachers have planning sessions away from the classroom. Since the introduction of the 2014 curriculum, I have taken the opportunity to teach computer science and coding, particularly at KS2.

I have participated in the physical computing and primary programming online training courses. The physical computing course was great for improving my subject knowledge, and also for consolidating the learning I had recently enjoyed at Picademy. It was really useful to gain this subject knowledge, which will be invaluable as I develop my use of physical computing in the classroom.

The primary programming course contained a lot of subject knowledge I already had. This course was really great for developing my pedagogy, as it introduced and encouraged concepts such as 'tinkering' and testing, as well as developing planning and evaluation processes with learners. Many of the ideas from this course are quickly transferable to the classroom.

I found that both courses were full of practical, hands-on activities, which really suits my style of learning. My favourite part of the courses was making the practical light-flashing activities on the physical computing course. Being fairly new to Python and physical computing it was great to have some success! The videos and screencasts were really useful, especially to take things step by step.

Participating in the courses has had a positive impact on me, in terms of subject knowledge, introducing more structure to my teaching, and taking ideas and transferring them readily into the classroom.

Your ideas

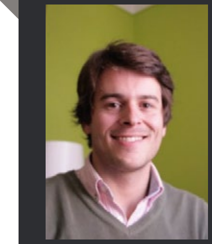
There are lots of other courses in development. Look out for courses on creating a makerspace in your school, and moving from block- to text-based programming. We are always open to ideas about what to develop next for our online courses. If there's something you'd really like to learn about with us, email us at **onlinelearning@raspberrypi.org**. (HW)

TEACHING PROGRAMMING TO EAL STUDENTS

How can we support computer science students who don't speak English as their first language?

STORY BY Cat Lamin

SPOTLIGHT ON FRANCISCO GOUVEIA



Job description: Computer Club Coordinator and Teacher on the island of Principe

First language: Portuguese

Other languages spoken: Many!

Language spoken by pupils: Portuguese

Programming languages taught to pupils: Scratch

Problems encountered: While it's easy to switch languages in Scratch, so that pupils can learn about block-based coding in their own language, the libraries of images are still in English and this already causes problems for my pupils. They can find it difficult to find a specific image because you're effectively spelling out letter sounds instead of just saying the words. I've made a start with translating the libraries for my pupils, but it's hard work for them and for me, and as soon as a new version of Scratch is released, I have to start again. I'd considered teaching my students Python as a next step, but for them it's that much harder because Python is written in English and they only speak Portuguese.

Next Steps: To tell you the truth, using Python wasn't in my immediate plans, although I was exploring the idea. But reading through Carrie Anne Philbin's **Adventures in Raspberry Pi**, and the transition she proposes - of using the same types of commands but in two different languages, and in so doing gradually 'graduating' from Scratch to Python - the problem kind of popped up...

We often think of the learning process for text-based programming languages as being similar to learning a foreign language, but how much harder is that process if your students are already learning English as an additional language? Most popular programming languages, including Python, contain syntax which is fundamentally English. This adds a whole extra layer of learning for EAL students, and we must be aware of this if we are to effectively teach computer science to these pupils.

I recently met Francisco Coutinho Gouveia, who teaches Portuguese-speaking students on the island of Principe. He mentioned the difficulties he's been facing with teaching students to code when English isn't their first language. It got me thinking about the same problem in UK schools for EAL pupils.

One thing we know about CS and coding is that it's incredibly empowering for students across a range of abilities. Even students who find academic subjects difficult can find success in coding, and I've seen an incredible improvement academically in pupils who have attended Code Clubs and Raspberry Jams. Their passion for computer science has improved their computational thinking skills, which in turn have improved their academic performance.

Can we provide the same experience for EAL pupils in an English-speaking classroom?

Teaching Python

Let's take a look at Python, the most popular language taught in schools in the UK. Its popularity is due to the balance it strikes between ease of use and the ability to solve complex challenges.

While it was created by a Dutch programmer, Guido Van Rossum, Python is a programming language written in English. The commands are all English, and it is common practice in the global Python community to write code almost exclusively in English, including the naming conventions for variables, functions etc. – even if the programmer does not speak English as a first language, or even fluently. There can be problems with Python interpreting non-ASCII characters, and further problems can arise when libraries are written in both British English and American English.

Programmers around the world seem to agree that learning English is a core element of becoming a developer, as it establishes a common ground for discussion. Indeed, one commenter on stackoverflow.com explains that he learnt English because of his desire to learn to code, but it was only after he had learnt the core commands that he understood their literal English translations. This raises the question: do we need to understand the commands we are typing in order to understand and recall the core commands of Python?

A deeper understanding

I would argue that, while it is possible to simply learn the commands by rote, it is significantly easier to apply them if you can appreciate why you are using these words and phrases. For example, it helps to understand that 'while True' means that as long as the code is running, the statement is true and will keep looping forever. Without understanding English, this phrase is just a random collection of letters. If we attach meaning, and a local translation, to this phrase, there is a greater chance of the

It is important that we allow children to become excited about coding, no matter what their first language happens to be

Image credit: Connor Ballard-Pateman

student being able to retain and use the phrase effectively in future code.

It's also useful to remember that libraries, APIs and supporting documentation are all usually written in English. It is therefore valuable for users to gain a basic understanding of the English language.

The solution

Based on my experience, I would propose one of two strategies for teaching Python to a non-English speaker.

- Assume the user will accept the commands at face value with no explanation of the specific vocabulary. Instead, explain their context within the code e.g. 'while True' is a forever loop. It doesn't matter if you don't understand the words 'while' or 'true' in your native language – what is important is your understanding of what the code is doing. This would require clear descriptions of the purpose of various commands throughout the written resources used by the teacher.

- Allow users the opportunity to translate and understand commands, core words and phrases. Spend time identifying key words and phrases that are common in Python, and use them to create a worksheet or paper resource which explains these words in the user's native language and then tests their understanding.

The first method assumes that the user is able to abstract relevant information without the need to understand the individual words. It is possible to understand and write programs in Python without completely understanding the translation of the words. However, this is a limited method for teaching and would only be useful at the most basic level.

> “ PUPILS ARE TAUGHT TO UNDERSTAND HOW DIFFERENT METHODS WORK AND WHY THEY ARE EFFECTIVE

The second method provides a more robust understanding of the English language, and of the logical reasoning behind the code. My experience is that learning by rote is not the most effective way of ensuring that pupils retain their knowledge and apply their understanding to unfamiliar situations. Retaining and applying knowledge is more likely if pupils are taught to understand how different methods work and why they are effective. In classrooms today, pupils are often taught not just to memorise their multiplication tables, but to understand multiplication in various different ways. They might use arrays to better understand multiplication as the repetition of groups, which in turn allows a better understanding of division, as pupils recognise those groups as part of the inverse operation.

My gut instinct is that the latter method would be a more effective way to support the learning of Python for a non-native English speaker. However, it is important to be aware that some students may be able to effectively use abstraction to understand the code without learning or understanding the language. They might prefer to learn the code and then retrospectively learn the English language.

Modern Foreign Languages

Modern Foreign Language (MFL) teachers would argue that in language learning, understanding through context can be a key aspect of learning. Being able to translate core vocabulary is also an essential aspect of learning any language, whether that is a spoken language or a computer language. Indeed, when we teach Python to English speakers, it is essential to explain the vocabulary to the learners when it is unclear – for example 'def' is short for 'define', which allows us to define a function.

It's also worth considering the English language and its significance as a global language. English is currently the language of the internet, the language of trade, and the language of commerce. However, we need to be aware of the often quoted 'arrogance of the English-speaking world': the assumption by native English speakers that 'everyone speaks English'. Let's not offend communities in the non-English-speaking world by assuming that they are willing to learn and understand English phrases. We cannot assume that they already understand words and phrases like 'true or false' or 'if'. I would suggest that time is taken to ensure that the resources we provide are not patronising, and do not assume a dominance of the English language. It is essential that we are nurturing and supportive of learners so that we achieve our goal of making computer science accessible to all. (HW)

OER Global Logo by Jonathas Mello is licensed under a Creative Commons Attribution Unported 3.0 Licence

OPEN EDUCATION RESOURCES

Software development has benefited greatly from the free and open-source software movement. Can the world of education benefit as well?

STORY BY Marc Scott, Head of Curriculum Development at The Raspberry Pi Foundation

In the 1980s Richard Stallman, a young hacker at MIT, grew frustrated with the way that software was being produced and licensed.

OPEN EDUCATION LINKS AND RESOURCES

- Creative Commons licences: **creativecommons.org**
- Open University OpenLearn Create: **helloworld.cc/2ucgjNj**
- OER Commons: **www.oercommons.org**
- Computing At School resources site: **helloworld.cc/2tX2wih**
- Raspberry Pi Resources: **helloworld.cc/2vlJbYC**

At the time, many companies were producing closed software that was impossible to decode, and was often licensed so that altering it in any way would be illegal. Stallman founded the Free Software Foundation, and began working on his own operating system and collection of software tools that he called the GNU operating system. His idea was that the code for GNU software would be available for people to read, share and improve. He even created his own license, which software developers could use, called the GNU General Public Licence (GPL), and in doing so began the open-source software revolution.

Today, companies such as Google, Microsoft, Apple, and many more are all creating or contributing to open-source software and hardware projects. Additionally, thousands of programmers, designers, and engineers volunteer their time and talents to develop open-source projects. The Linux operating system, for example, is the predominant operating system on mobile devices, web servers, supercomputers, and embedded computers. It powers Google's Chromebook, the Raspberry Pi, and many people's desktop computers as well. Linux now contains around 15 million lines of code, with less than two percent of that written by the original creator, Linus Torvalds. That means that around 14.7 million lines of code have been written by volunteers working on Linux.

Why open-source?

Why do organisations, businesses, and volunteers spend so much time and effort developing open-source software, if they're not going to sell it on? The truth is that open-source software is usually a vast improvement on anything a single individual or company could produce in isolation. Open-source software has the advantage that large numbers of people can study the code, find bugs or potential improvements, and then alter the software themselves. As a result, this software is often more robust, more powerful, and easier to use.

Open-sourcing software and educational resources can make them more robust and easier to use

The open education resources movement

If this model works so well in the field of software development, couldn't it also work in the field of education? Educators the world over produce lesson plans, slide shows, worksheets, websites, video tutorials, diagrams, and even whole courses to help their students learn. Too often the fruits of their labour lie dormant in their desk drawers, or stored away safely on USB flash drives, waiting for the next academic cycle when they'll be used with the next cohort of thirty students. Wouldn't it be better if, as educators, we shared the resources we produce, allowed others to correct them and improve on them, and let our learners access them, for free, at any time of the year?

Open educational resources (OER) are teaching and learning materials that are freely available online for everyone to use: educators, students, or independent learners. The key aspect of OER is the licence under which they are shared, meaning that they can be republished, edited, and used free of charge – all without having to ask for the author's permission.

The OER movement started at the turn of the millennium, as first MIT, and then other educational institutions around the world, began to place their courses online, for free. As the number of institutions offering free or open courseware increased, UNESCO organised the first Global OER Forum in 2002, where the term 'OER' was adopted.

> OPEN-SOURCE SOFTWARE IS USUALLY A VAST IMPROVEMENT ON ANYTHING A SINGLE INDIVIDUAL OR COMPANY COULD PRODUCE IN ISOLATION

The movement has grown rapidly since then, with a huge number of universities, colleges, and schools offering their educational material for free. But anyone, including individuals, can assist in helping the OER movement to achieve its goals. Any educator can open-source their work. It's as simple as copying and pasting a Creative Commons licence to the bottom of a worksheet or scheme of work, and then publishing it online.

Get involved

There are many ways to distribute your work to learners and educators. You could simply upload your work to any cloud storage provider and share the links on social media. You could convert your slide show to a video, add a voice over, and upload it to YouTube. There are also some sites that cater specifically for OER. For instance, the Open University operates OpenLearn Create, and ISKME operates a platform called OER Commons.

Then there are more specialised sites, such as Computing At School, that allow hosting of OER. The Raspberry Pi Foundation publishes all its resources under a Creative Commons Licence, and we love it when people help improve our resources. You can see all our educational content on GitHub, copy the resources if you like, tailor them for your students, and then submit improvements.

Whether you share and publish your resources or just help improve someone else's, becoming part of the OER movement will help educators and learners all over the world, and together we can try to replicate the success of open-source software in the world of education. (HW)

AGE RANGE

11 years

LESSON TYPE

Programming

REQUIREMENTS

- Scratch

SCRATCH PONG

Laura Sach's creative task assesses how well your new Year 7 students understand basic programming concepts as they join you from junior school

At the start of the school year, students arrive from junior schools far and wide to join secondary school computing classes. Some will have covered the primary computing curriculum extensively whilst others will have barely scratched the surface. Use this project as a basis for assessing your students' current skill levels, and to work out the gaps in their knowledge.

Planning ahead

Begin by thinking as a class about which sprites will be needed for the game of Pong. To do this, you could show a video of the original game and ask students to identify the different objects they can see.

Students will then implement a basic Pong game in Scratch. This will act as a refresher for those students who have covered the concepts previously, but will allow those who have not used Scratch a chance to familiarise themselves with the environment and the programming concepts.

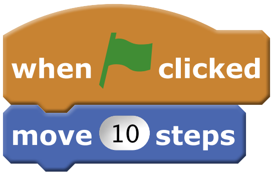
The following instructions can be provided to students, who can use the coloured text to help them figure out which blocks they need. The additional images could be provided to those students who are unfamiliar with Scratch.

1. Creating the Ball

Delete the cat sprite and add a sprite that looks like a ball. You can use one of the existing sprites or you can paint your own ball.

Pull out blocks into the Scripts area to do the following:

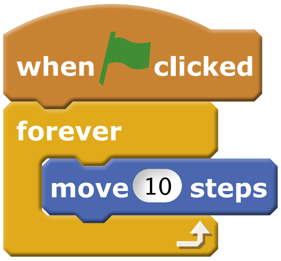
- When the flag is clicked
- Move the ball 10 steps (**figure 1**)



```
when 🏴 clicked
move (10) steps
```

Figure 1

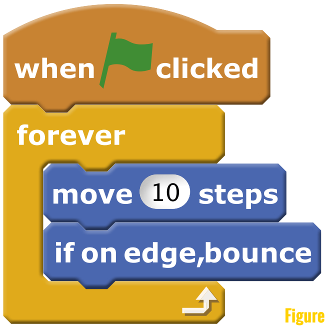
What happens to the ball? Can you add something to make the ball keep moving (i.e. move forever)? (**figure 2**)



```
when 🏴 clicked
forever
    move (10) steps
```

Figure 2

What happens when the ball hits the edge? Can you add something to make the ball bounce off when it hits an edge? (**figure 3**)

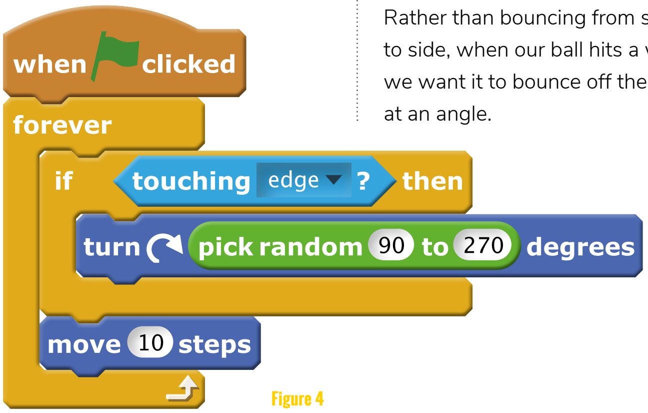


```
when 🏴 clicked
forever
    move (10) steps
    if on edge, bounce
```

Figure 3

2. Bouncing ball

Rather than bouncing from side to side, when our ball hits a wall we want it to bounce off the wall at an angle.



```
when 🏴 clicked
forever
    if <touching [edge ▾] ?> then
        turn ↻ (pick random (90) to (270)) degrees
    move (10) steps
```

Figure 4

ALTERNATIVE ACTIVITY IDEAS

Revise and create

Adapt this project theme for other age groups:

14-18 years - Programming

Use the same concept to assess the starting level of your GCSE or A level Computer Science students. Set a tutorial task followed by an opportunity for students to be more creative and show off their skills.

FURTHER READING

The original Pong game: helloworld.cc/2vMTamE

Video of the original gameplay: helloworld.cc/2vaQPVH

ASSESSMENT

You can allow students to try out each other's games, and also incorporate a peer assessment task, as long as you are careful to model the kinds of responses you wish to see. A page full of 'I loved it' responses isn't much use to anyone!

- What do you think <name> could improve about their program?
- Was there anything in <name>'s program you could have used in yours?
- How do you think <name> programmed their <x> sprite to do <y>?

THE CHALLENGE

- Create a Pong game in Scratch using movement, variables, and logic
- Apply these concepts to create an original game
- Peer assessment

Remove the if on edge bounce block. In its place, we need some new code. You need to figure out how many degrees to allow the ball to turn.

- If
- The ball is touching the edge
- Turn random degrees
- Move 10 steps (**figure 4**)

Experiment with the range of random degrees the ball can turn - some numbers will produce much better results than others.

Challenge: Is there a better way of making your ball bounce off the walls?

3. Creating the paddles

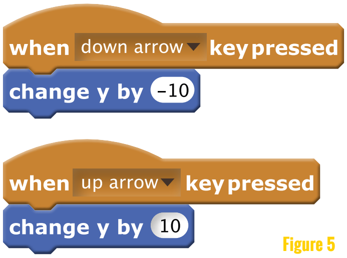
Draw a new sprite to represent the paddle.

Drag blocks into the work area to make the paddle work:

- Set the start position of the paddle
- When the up key is pressed, move up the screen
- When the down key is pressed, move down the screen

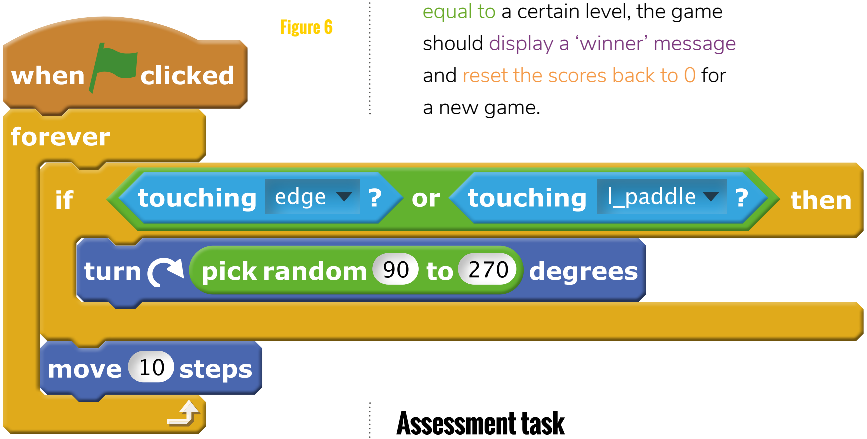
Hint: To get the up and down movement, think about which axis on a graph represents up/down (**figure 5**).

Start the game and check that your paddle moves. You might notice that the ball doesn't bounce off the paddle. Can you add some more code to the ball to make it bounce if it is touching the paddle? The code for bouncing is exactly the same as the code you used to bounce off the wall in step 2 (**figure 6**).



```
when down arrow key pressed
change y by -10

when up arrow key pressed
change y by 10
```

Figure 5



```
when [flag] clicked
forever
    if touching edge ? or touching l_paddle ? then
        turn ↻ pick random 90 to 270 degrees
    move 10 steps
```

Figure 6

4. Two player mode

- Duplicate your paddle sprite.
- Change the keys you use to move the sprite, otherwise one player will control both paddles!
- Change the starting position of the second paddle so that it starts on the opposite side of the screen.
- Now alter the ball script so that it bounces off the second paddle.

5. Extra challenge

If we now have two players, we need to have some way of keeping score.

- Can you use the variables in Scratch to create a score box for each player?
- How will a player score a point? Perhaps each end of the game board could have a different coloured line so we know when the ball has reached it.
- When one player scores a point, their score should increase by one.
- If one player's score becomes equal to a certain level, the game should display a 'winner' message and reset the scores back to 0 for a new game.

Assessment task

During the following lesson, set the assessment task for the students. They must use all their skills to design and build any game of their choice in Scratch. Students who have 'done Scratch' before should attempt to show off their skills, and those who are new to programming or to the Scratch environment will not be left behind, thanks to the introductory tutorial. (HW)

AGE RANGE

7-10 years

LESSON TYPE

Visual / block-based coding

REQUIREMENTS

- Computer running Scratch 2.0 for each group
- Mini whiteboards and pens for unplugged elements

CLOCKWORK CODING

Explore repetition and variables by coding your own stopwatch and clock in Scratch

STORY BY Jon Chippindall

One of the joys of learning to code is realising that we can become creators of the technology we might normally reach for in an app – such as a stopwatch or clock. In this lesson students program their own timing tools, which they can go on to use in other subjects such as science and maths.

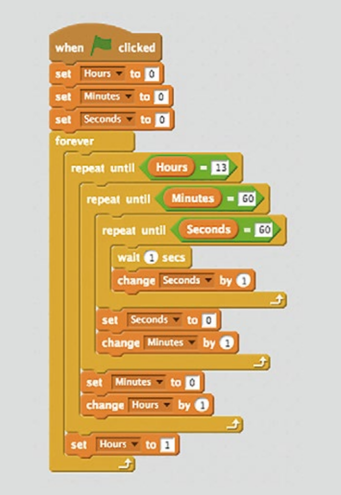
The activity is in three stages. First, students create a human model of a stopwatch, giving them time to decompose the counting process, consider where repetition plays a role, and write the algorithm to help with their coding. Pupils then code a stopwatch in Scratch before finally adapting this code to drive the hands of a clock face. To extend the challenge, pupils can create further clock faces and enable the display to be toggled between them.

THE CHALLENGE

- Can students write the algorithm for a stopwatch, and identify how repetition plays a role?
- Can students translate their algorithm into code using repetition commands and variables?
- Can students adapt their code to drive the hands on a clock face?

OPTIONAL

- Can pupils switch between different clock faces, including a digital clock?



```
when ⚑ clicked
set Hours ▾ to 0
set Minutes ▾ to 0
set Seconds ▾ to 0
forever
    repeat until Hours = 13
        repeat until Minutes = 60
            repeat until Seconds = 60
                wait 1 secs
                change Seconds ▾ by 1
            set Seconds ▾ to 0
            change Minutes ▾ by 1
        set Minutes ▾ to 0
        change Hours ▾ by 1
    set Hours ▾ to 1
```

Figure 1: How pupils' stopwatch code might look - notice the variables needed were seconds, minutes and hours

FURTHER READING

This idea comes from a Barefoot Computing resource.

Example program: helloworld.cc/2vaA0tR

This lesson focuses on algorithms, repetition and variables. The following Barefoot Computing resources may be helpful (sign-up required):

Barefoot Computing on Algorithms: helloworld.cc/2uBwuHt

Barefoot Computing on Repetition: helloworld.cc/2w37t5H

Barefoot Computing on Variables: helloworld.cc/2v0IcfC

Human stopwatch

Whenever possible, encouraging students to first act out the system they are trying to create can be incredibly powerful in helping them to break down the steps in the process, and start thinking about how they might translate this into code. Start by putting students in groups of three (seconds, minutes, hours), with a mini whiteboard and pen each, and get them to become a human stopwatch (perhaps not the most accurate one!). Lead a discussion as a class to determine the algorithm for the stopwatch. Where is repetition present? When do the seconds reset to zero? What happens to the minutes at this point?

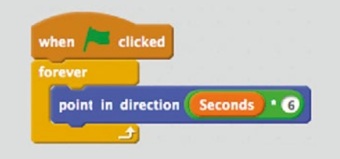
Coding a stopwatch

Challenge students to code a simple stopwatch in Scratch. Which three variables will they need? Why? Students will need to know how to use Scratch's `Repeat until` block with an operator to ensure that the stopwatch counts correctly – and we also need it to start at zero each time.

Driving a clock face (or three!)

With a bit of tinkering we can now repurpose our code to drive a clock face (or three). Ask pupils to create a clock face background in Scratch and draw three clock hands as separate sprites (**Top tip**: ensure the hands are drawn horizontally and pointing to the right with the sprite centre at the base - this will make the geometry far easier!)

Using the variable values to drive our clock hands is now simply a matter of geometry. **Figure 2** shows the code for the second hand. Why are we multiplying by six? Can pupils work out the code for the remaining hands?



```
when ⚑ clicked
forever
    point in direction (Seconds * 6)
```

Figure 2: Why are we multiplying 'seconds' by six to give the direction of the second hand?

The clock is now complete, and can be set using the sliders accessible by double clicking the variables windows.

As an extension: Why not get pupils to create a second clock face with Roman numerals, and add code to switch between the two? Or, even more challenging (See suggestion for 11- to 13-year-olds), create a digital clock, which uses the variable values to switch costumes for a digital style display. (HW)

ALTERNATIVE ACTIVITY IDEAS

This lesson can be adapted for different age groups:

5-6 years - Using ScratchJr

- Students could create a program in a simple language, such as ScratchJr, with links to time and repetition. For example, a program in which their sprite says something or changes appearance every ten seconds.

11-13 years - More challenging programming

- Can students create a digital clock? One way this could be achieved is by using sprites for each digit of the clock face, with multiple costumes. The variable values can then be used to switch these costumes (see the example program in the Further Reading box).

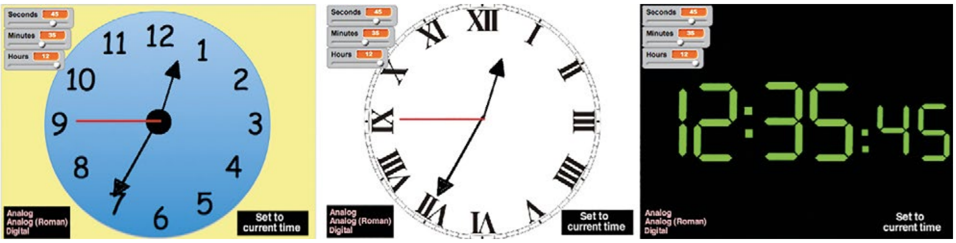
Physical Computing

- Can students program a micro:bit or CodeBug to show a stopwatch on their displays?
- Can students try making a physical clock driven by motors or servos connected to a Crumble controller? Challenge the students to move the hands accurately - bearing in mind that most servos can only move through 180 degrees.

ASSESSMENT

- **What is your algorithm for the stopwatch? Why?**
- **How have you used repetition and variables in your code? Why?**
- **How does your code drive the clock hands or digital numbers?**

Create your own clock in Scratch



AGE RANGE

7-10 years

LESSON TYPE

Block-based programming

REQUIREMENTS

- Scratch 2.0
- Google Docs

INTRODUCING GAME DESIGN

Inspire students as young as eight to design and program their own games with collaborative and resilient working

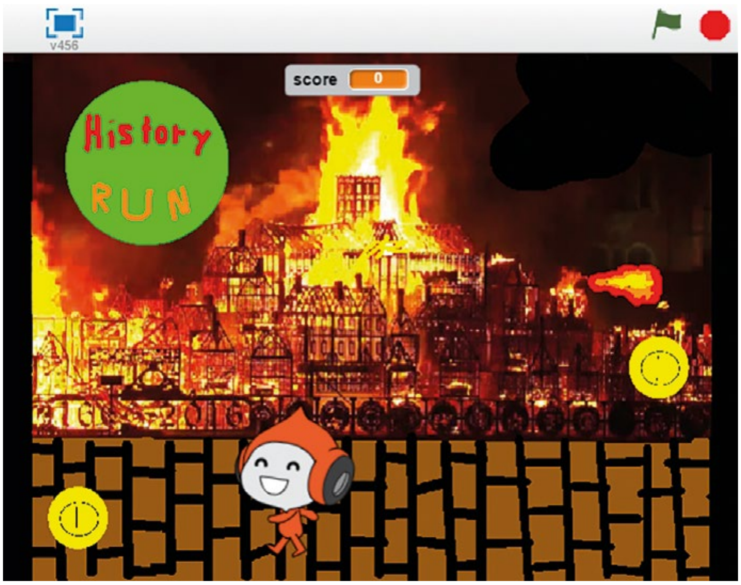
STORY BY Nikki Cossey

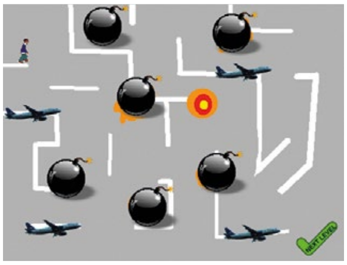
I designed this scheme of work to introduce game design to my Year 4 students in a fun, collaborative way. They role-play working in the games industry, and design a game to meet specific requirements. It is easy to make links with other subjects: this unit is cross-curricular with History as they explore the history of London during the spring term. Heterogeneous groups no greater than four work best, and each group selects a leader.

The group leader has the usual responsibilities of keeping track of time and ensuring everyone is on task. He or she also tells me when they are ready to go on to the next step of the process, or seeks assistance from me, and is the one from whom I ascertain the group's progress.

I allocate more than a term and a half (40-minute lessons, once a week) to this unit to enable them to finish their games. They learn key skills, including the game design process, writing pseudocode algorithms, user interface design, collaboration in Google Docs, game creation in Scratch and, of course, programming. Encourage them to start small by creating a simple, working game, and not worry about levels or choosing a character.

A game based on the Great Fire of London





A game based on the Blitz

The game design process

An engaging way to begin a game design unit is to discuss students' favourite video games. This will be a lively conversation – students love playing games and talking about their favourites! Delve deeper into this discussion by asking students to explain why they enjoy those games. This should elicit responses such as that they are fun, colourful, and challenging; they have a theme (car racing, football); and so on. Project their responses so they can instantly see the requirements for a successful game.

I believe in teaching a game design process. I want students to appreciate that there is a creative design process that needs to be followed in order to create products, from the original concept or client brief to testing and evaluating. There are various processes for game design, and I combine ideas from a few models to make it accessible for my Year

THE CHALLENGE

Follow the game design process:

- What are the requirements?
- Come up with an idea and do some research
- Design the interface
- Write the pseudocode algorithms for your game
- Create the game in Scratch - first artwork, then code
- Test, test, test
- Repeat steps 4, 5 and 6 until it's perfect!

The game requirements are that their game should:

- Have clear instructions on how to play
- Be about a historical event that took place in London
- Not be too easy!
- Have a colourful, appealing interface
- End appropriately, by winning or losing
- Have sounds

OPTIONAL REQUIREMENTS:

- Have another level
- Keep track of score
- Have a timer

4 students. The creative industries are usually project-based and collaborative, and I aim to replicate this with my game design unit.

The first step in the game design process is the requirements, which are essentially the success criteria and extension tasks. I pretend to be their client and tell them that this is what their game must have. Students find this role play motivating. They imagine they are working in the games industry and having to satisfy a client. Working collaboratively on a Google Doc, students decide on an idea, do research, design the interface and write pseudocode algorithms. The Google Doc is shared with me, and I track their progress, provide feedback verbally or as comments, and ensure their game idea is feasible. The game requirements are in the header, so they are constantly reminded of the success criteria.

Teaching computer science concepts

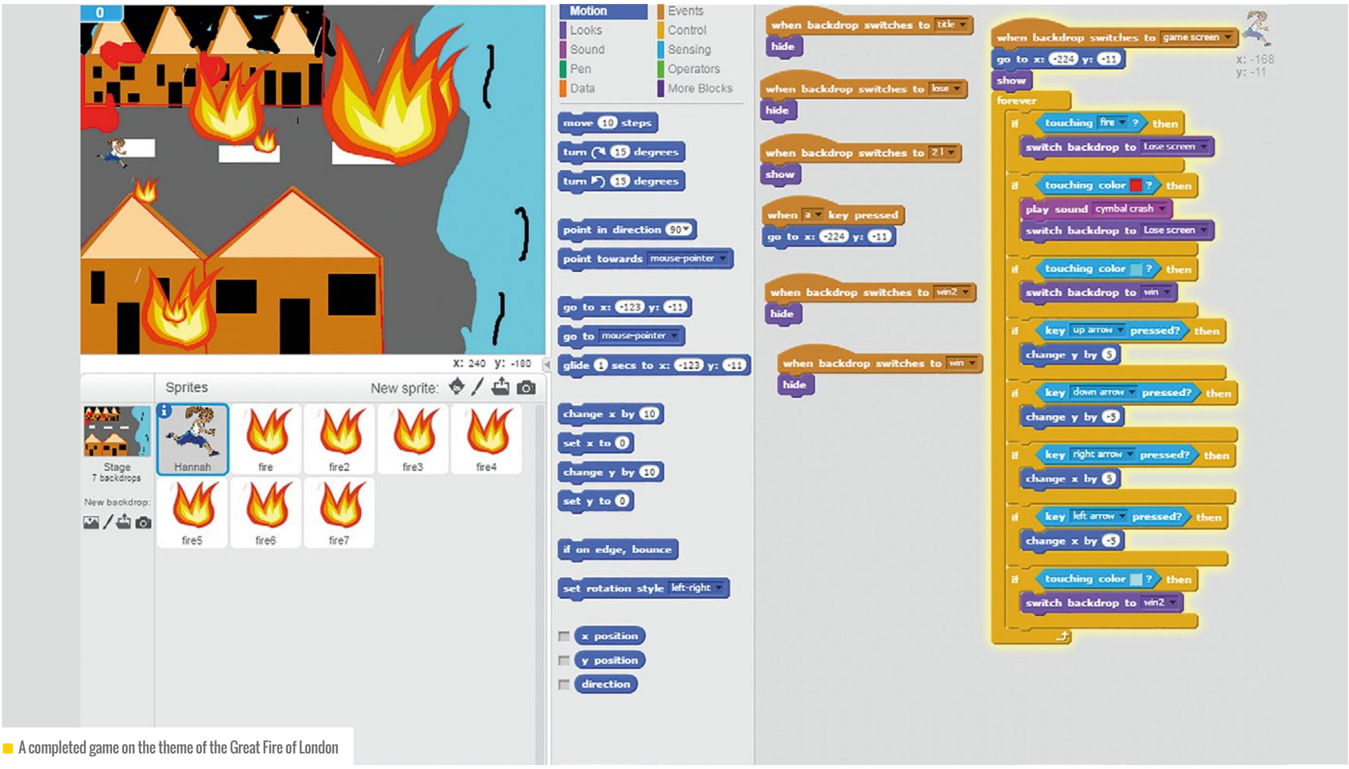
A project this collaborative and open-ended could be difficult for a teacher to manage. However, the game requirements impose some limitations on the students' ideas, and close monitoring of the planning stage ensures that their games are achievable. I also only let them create one of three types of games: platformer, scroller or maze. They know how to make a maze game as they made one in Year 3 when they learnt about Ancient Greece and Theseus and the Minotaur. They learnt the concepts of sequence, repetition, and selection, and how to control a character using the arrow keys.

Most know how to create a scroller game, since it is simply moving from left to right, and they have created animations in Scratch where a character has moved along the stage. A brief class discussion and demonstration will remind them of the blocks of code required to create these games. They should only need teacher assistance if their character jumps (if they choose a platformer game), or if they complete the core game and move onto the extension tasks.

Step four of my game design process is writing algorithms. I show them how to write pseudocode,

ASSESSMENT

- How did planning help you when creating your game?
- Where did you use sequence, selection, and repetition in your game?
- What problems did you encounter? How did you overcome them?

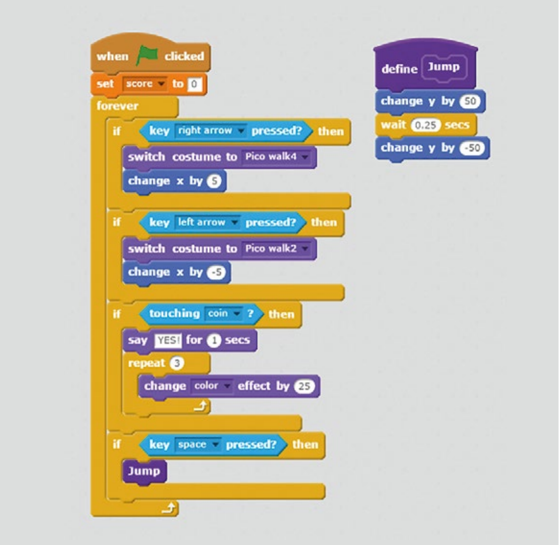


```
when backdrop switches to [title]
hide

when backdrop switches to [lose]
hide

when backdrop switches to [21]
show

when [a] key pressed
go to x: -224 y: -11

when backdrop switches to [win2]
hide

when backdrop switches to [win]
hide

when backdrop switches to [game screen]
go to x: -224 y: -11
show
forever
    if <touching [fire] ?> then
        switch backdrop to [Lose screen]
    if <touching color [ ] ?> then
        play sound [cymbal crash]
        switch backdrop to [Lose screen]
    if <touching color [ ] ?> then
        switch backdrop to [win]
    if <key [up arrow] pressed?> then
        change y by 5
    if <key [down arrow] pressed?> then
        change y by -5
    if <key [right arrow] pressed?> then
        change x by 5
    if <key [left arrow] pressed?> then
        change x by -5
    if <touching color [ ] ?> then
        switch backdrop to [win2]
```

A completed game on the theme of the Great Fire of London

emphasising that they must use basic English, a bulleted or numbered list, and specify the Scratch blocks needed. They write the pseudocode algorithms for each of their interfaces. Some will need reminders that algorithms are instructions for the computer – not the player! One group member can consult the Scratch menus to identify the names of the blocks they require while the others write the algorithms.

Scratch code for making a sprite jump and collect coins



```
when green flag clicked
set [score] to 0
forever
    if <key [right arrow] pressed?> then
        switch costume to [Pico walk4]
        change x by 5
    if <key [left arrow] pressed?> then
        switch costume to [Pico walk2]
        change x by -5
    if <touching [coin] ?> then
        say [YES!] for 1 secs
        repeat 3
            change [color] effect by 25
    if <key [space] pressed?> then
        Jump

define Jump
change y by 50
wait 0.25 secs
change y by -50
```

An example of a group's algorithm:

Algorithm for title screen:

- When green flag clicked
- Play button will appear. It is a sprite
- Bridge is now in the background
- When play button/sprite is clicked change to backdrop 2 and hide play button.

Algorithm for game screen:

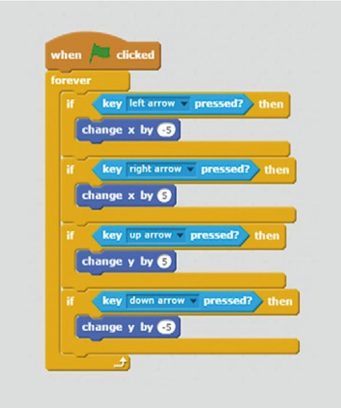
- When on backdrop 2 get the bridge in front
- Put the obstacles on the bridge
- Let the player use the arrow keys to move.

Algorithm for win screen:

- It will show the win screen when you have crossed the bridge and you are going onto the next stage
- If touching (brown colour) switch to win screen
- Switch backdrop to Win
- When on win screen put a trophy in the middle
- Have a sprite that has a button that says start again and on the other side have a button that says quit game
- The trophy will have fireworks around it.

Algorithm for lose screen:

- When you fall off the bridge (if touching blue) show lose screen
- You can lose by hitting one of the obstacles or hitting the side of the bridge
- When on lose screen put a sad emoji in the middle of the screen
- Have a sprite that has a button that says start again and on the other side have a button that says quit game. The sad emoji will have a sign saying try again.



```
when [green flag] clicked
forever
  if <key (left arrow) pressed?> then
    change x by (-5)
  if <key (right arrow) pressed?> then
    change x by (5)
  if <key (up arrow) pressed?> then
    change y by (5)
  if <key (down arrow) pressed?> then
    change y by (-5)
```

Scratch code for moving a sprite

When their algorithms are sufficiently precise, and their planning is satisfactory, students sign up to a class I created using my Scratch educator account. One student creates the title screen, one creates the game screen, another makes the win screen and another makes the lose screen. They design the necessary sprites and backdrops then start coding.

> ENCOURAGE THEM TO START SMALL BY CREATING A SIMPLE, WORKING GAME

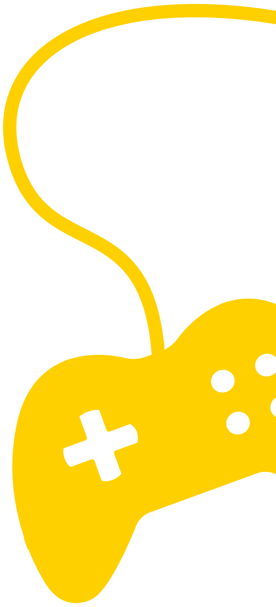
Using their algorithms, they locate the required blocks, and basically teach themselves event-driven programming. When the green flag is clicked, they know the title screen and play button sprite should show; and when the play button is clicked, the game screen must show and the play button should disappear. They are able to select blocks independently to manage events in their game, and control how a player wins and loses.

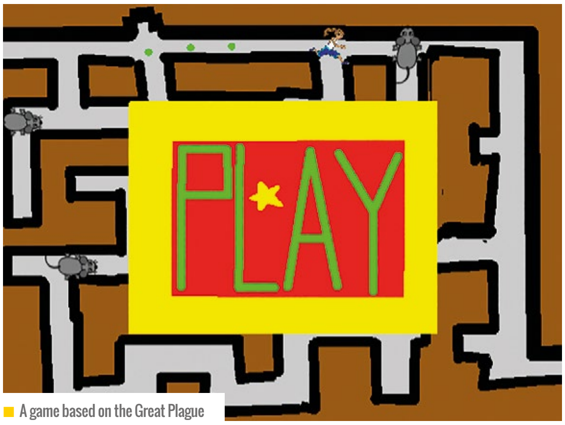
The computer science concepts I reinforce with my Year 4s are sequence, selection, and repetition, and I introduce procedures where they define a custom block to create a Reset or Jump block. Students' curiosity or ambition is often a cue for me to provide tips or advice to reinforce existing knowledge, or teach them something new, which they can implement in their game straight away. This results in deeper learning and more long-term retention, as it is a form of enquiry-based learning.

Finishing the game

The student creating the game screen has the most coding to do, so their project becomes the main project when the others in the group finish their interfaces. Backdrops, sprites and code from group members' shared projects are put into the backpack and transferred to the main project. Students take it in turns to assemble the main project while the others advise – paired programming of sorts. If students need more help with coding, they can follow a studio I created where I have added projects with commonly required code. They put the code from the projects in the studio into their backpack and incorporate it into their project – useful for students who require scaffolding, and similar to help sheets. It also ensures that you remain a facilitator by guiding students to code and debug their own games resiliently.

Groups will finish at different times. While other groups catch up, there are extension tasks built into the game requirements, or they can market their game by creating a print or TV advert, or use Makey Makey to create a controller for their game. End the project with groups assessing each other's games by playing them. This can be done by letting them add their projects to a studio you create. They can Favourite or Love projects and add comments. (HW)





A game based on the Great Plague

ALTERNATIVE ACTIVITY IDEAS

Explore and make

Adapt this project to challenge other age groups.

Unplugged

- Students design a game using a storyboard template.

7-10 years - Programming

- Students use variables to keep track of score and lives.
- Students design games using advanced computer science concepts: make sure the game requirements are more complex.
- Students make their game then use Makey Makey to create a controller.

11-13 years - Text-based Programming

- Students use a programming language such as Python to create a simple game.

FURTHER READING

A completed game from one of my student teams:

helloworld.cc/2h5FT5k

Commonly required code for games in my studio:

helloworld.cc/2vbxBz6

PYTHON AND SQLITE

Ben Garside introduces SQLite3, a Python library that allows you to create and manipulate local databases

Databases underpin our electronic lives. A significant number of IT systems, including websites, games consoles and mobile phone applications, rely on databases to function. When learning about databases, many students and educators face the challenge of setting up their own databases when they don't have an SQL server to connect to. A solution to this problem is to use SQLite, which is free and open-source. It creates a local database, stored in your document folders rather than on a server.

SQLite is used by developers. It is useful if you are developing an application that uses a database, but don't want the application to have to connect to an SQL server.

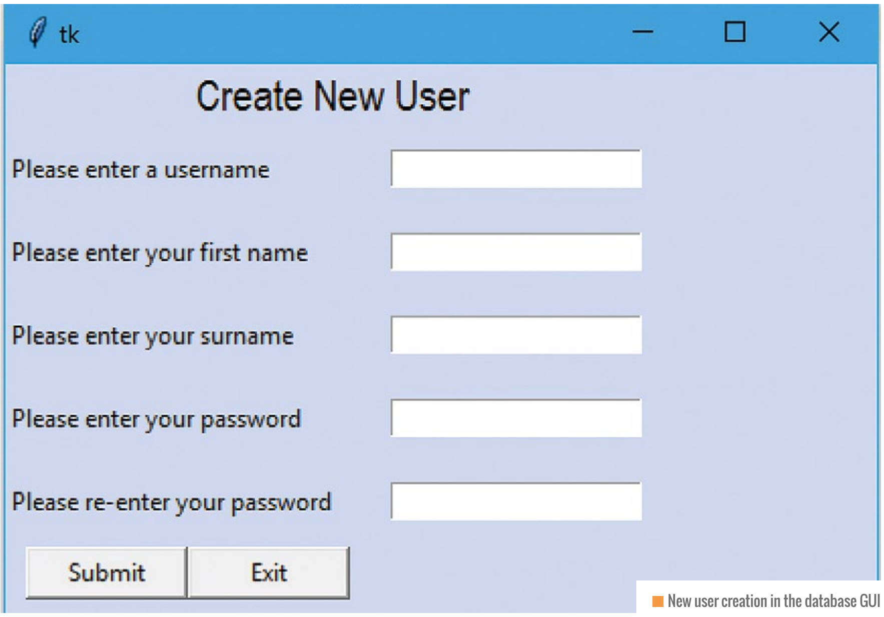
There are various ways to work with SQLite, but Python (from versions 2.5 onwards) features a standard SQLite library. In this tutorial I will show you how to set up a database that will store users' details. We will then write some code that will allow users to create their own accounts.

Getting started

Make a new Python file and give it a name that describes what it is going to do, which is to create a new database and make a user table. **Create Database** or **Create User Table** would be suitable. Open the file, import the sqlite3 library, and make a connection to the database:

```
import sqlite3
with sqlite3.connect("users.
db") as db:
    cursor = db.cursor()
```

If the database **users.db** doesn't already exist, the program will create it. If it does, this line will make a connection to the database that you can refer to later by using the variable name `db`. A cursor is a database object that is used to manipulate data. There are commands associated with a cursor such as `cursor.execute()`, which will run a query, and `cursor.fetchall()` which collects the results of the query for you to use.


tk
Create New User
Please enter a username
Please enter your first name
Please enter your surname
Please enter your password
Please re-enter your password
Submit
Exit

New user creation in the database GUI

When working with databases, the term cursor doesn't refer to a pointer on the screen - it is actually an acronym (CURrent Set Of Records). The cursor can perform many operations in a database, including holding the data that is to be added to the database, or storing the data returned from running queries. You will use the cursor in several ways in this tutorial.

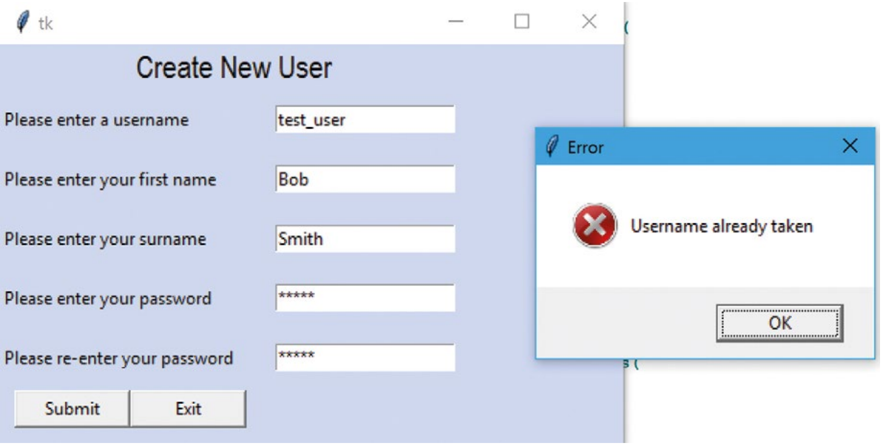
Creating a table

The next step is to make our first table, which is designed to store basic information about users.

```
import sqlite3
with sqlite3.connect("users.
db") as db:
    cursor = db.cursor()
cursor.execute('''
CREATE TABLE IF NOT EXISTS user(
userID INTEGER PRIMARY KEY,
username VARCHAR(20) NOT NULL,
firstname VARCHAR(20) NOT NULL,
surname VARCHAR(20) NOT NULL,
password VARCHAR(20) NOT NULL);
''')
```

We use the `cursor.execute()` function for the first time here. In the following brackets, we enter our SQL script as a string.

Our table will be called **user**, and will include the fields shown above. If you are familiar with MySQL, you will start to notice some differences in syntax here. You will notice that the Primary Key is declared next to the field. In sqlite3 you can declare this at the end of the SQL table creation statement by writing `INTEGER PRIMARY KEY`. This will not only make the userID field the Primary Key, but it will also automatically auto-increment the value with each user that is created, guaranteeing that each entry is given a unique identifier.



Create New User

Please enter a username test_user

Please enter your first name Bob

Please enter your surname Smith

Please enter your password *****

Please re-enter your password *****

Username already taken

Username validation check in the database GUI

SQL commands aren't case sensitive, but it is good practice to write the commands in capital letters for readability purposes.

Inserting data into the table

Let's create a test user to make sure that the table is accepting the data in a way that makes sense. We're going to use the `cursor.execute()` function to run a simple `INSERT INTO` script. We will add a new line at the end with the instruction `db.commit()`. This line saves the data to the database; without it the data would not be saved.

```
cursor.execute("""
INSERT INTO
user(username,firstname
,surname,password)
VALUES("test_
User","Bob","Smith","MrBob")
""")
db.commit()
```

Extracting data from your user table

To extract data from tables you use SELECT queries, which can be written to specify the format of the data. For example, you may want to extract just the usernames and passwords, and order them in ascending order by username; or you may want to pick out all users with the surname 'Smith'. This example will pull out all the data from the table so that we can see what that looks like. Then we will look at ways to use Python to manipulate the data.

To extract all the data from the table we are going to use the wild card * which represents 'all'. `SELECT * FROM user;` will pull out the data we need. We also need to tell the program to fetch the results.

Code:

```
cursor.execute("SELECT * FROM
user;")
print(cursor.fetchall())
```

Output:

```
[ (1, 'test_user' , 'Bob' ,
'Smith' , 'MrBob' ) ]
```

Code:

```
cursor.execute("SELECT * FROM
user;")
results = cursor.fetchall()
for item in results:
        print("Welcome:
",item[2],item[3])
```

Output:

```
Welcome: Bob Smith
```

The first example shows that `cursor.fetchall()` does print the data in the table, but the output is not ideal. It demonstrates that each row of the results is being stored in a list. In the second example, we have implemented a loop that would loop through all of the results (if there were more than one record in my table). This method demonstrates that we can be clever, and pick out only the specific fields we want to work with. For example, `item[2]` picks out the third item in the list.

FURTHER READING

Full SQLite tutorial:
helloworld.cc/2uhjYKV

Learn more about the syntax used with SQLite, and how to do more with relational databases, at the tutorialspoint website:
helloworld.cc/2susXGB

The YouTube videos on which this tutorial is based can be found here:
helloworld.cc/2txVv76

These videos take the project further, creating a relational database called Quiz that stores quiz questions and users scores.

This query is designed to demonstrate the data retrieval process. If you only wanted to extract the user's first name and surname, the query `SELECT firstname, surname FROM user;` would be more appropriate.

Making a function to add a user

Now that we have created a table, it would be useful to allow new users to create accounts for themselves, rather than expecting an administrator to enter all the users manually. The first step is to ask the user to create a username. We need to add some validation at this point to prevent a new user from choosing a username that already exists. The code below uses several of the concepts we have learnt about already.

```
def newUser():
    found = 0
    while found ==0:
        username =
input("Please enter a username:
")
        with sqlite3.
connect("Quiz.db") as db:
            cursor =
db.cursor()
        findUser = ("SELECT *
FROM user WHERE username = ?")
        cursor.
execute(findUser,[(username)])
        if cursor.fetchall():
            print("Username
Taken, please try again")
        else:
            found = 1
```

You may want to change the code above to give the user the option to exit the system - they might decide not to create the account because they're annoyed that their favourite username has been taken.

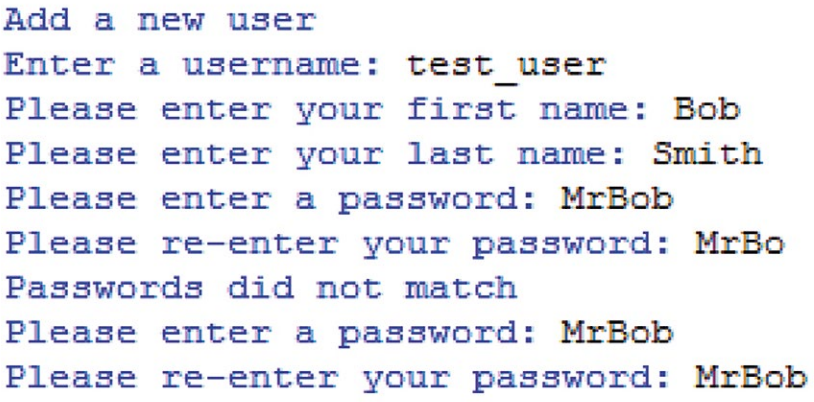
With this step we have constructed a different type of query. The query `find user` needs to take in the data stored in the variable **username**. The problem is that we can't enter the variable names, as it would read them as strings, not as the values being stored in the variable. You may be used to replacing variable values in a string with `%` and declaring them later, but this method leaves us open to the potential of an SQL injection attack. Instead we use `?` where we need to insert a variable.

On the next line, where the cursor executes the query, we can tell the database what to use instead of the question marks: `cursor.execute(findUser,[(username)])`

Next, we need to check whether the query has returned a value. To do this we use the If statement `if results`. If it has, we know the username has already been taken. If it returns nothing, we know we're ready to proceed with the account creation process.

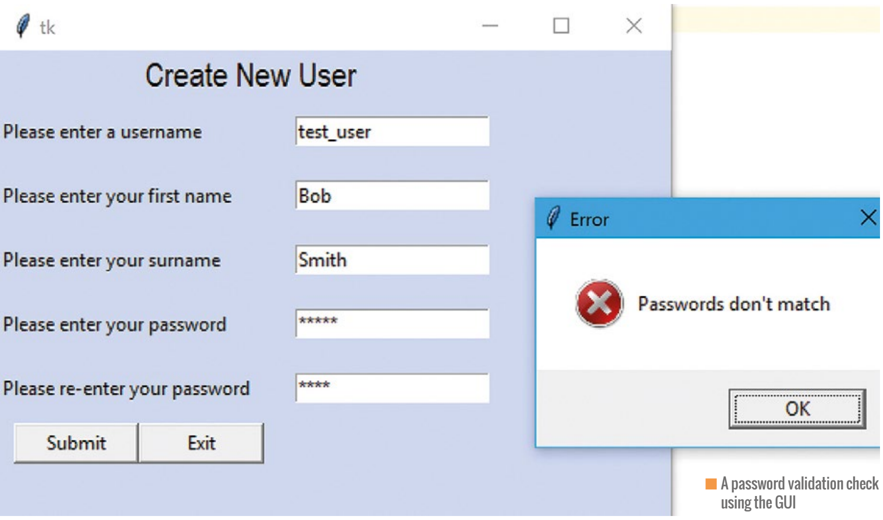
Now we want to ask the user to enter their details, do some more validation to check that they've entered the password correctly, and finally to execute a query to insert the values into the database.

```
def newUser():
    found = 0
    while found ==0:
        username = input("Please enter a username: ")
        with sqlite3.connect("Quiz.db") as db:
            cursor = db.cursor()
        findUser = ("SELECT * FROM user WHERE username = ?")
        cursor.execute(findUser,[(username)])
        if cursor.fetchall():
            print("Username Taken, please try again")
        else:
            found = 1
    firstName = input("Enter your first name: ")
    surname = input("Enter your surname: ")
    password = input("Please enter your password: ")
    password1 = input("Please reenter your password: ")
    while password !=password1:
        print("Your passwords didn't match, please try again")
        password = input("Please enter your password: ")
        password1 = input("Please re-enter your password: ")
    insertData = '''INSERT INTO user(username,firstname,surname,password)
    VALUES(?,?,?,?)'''
    cursor.execute(insertData,[(username),(firstName),(surname),(password)])
    db.commit()
```



```
Add a new user
Enter a username: test_user
Please enter your first name: Bob
Please enter your last name: Smith
Please enter a password: MrBob
Please re-enter your password: MrBo
Passwords did not match
Please enter a password: MrBob
Please re-enter your password: MrBob
```

SQLite output from the password validation check



tk

Create New User

Please enter a username: test_user

Please enter your first name: Bob

Please enter your surname: Smith

Please enter your password: *****

Please re-enter your password: ****

Error

Passwords don't match

A password validation check using the GUI

To call this function you need to write a line of code outside the function: `newUser()`. Test your code by adding a new user to your database, as well as trying to add a user that already exists.

You have connected to a database and successfully added data. Can you create another function that logs a user into your system by checking that their username and password match the records in your database? (HW)

Ben Garside is a teacher at Durham Johnston School. He trained in ICT but converted to Computer Science with a PG Cert in Computer Science from Newcastle University

MARK THORNBER TEACHER

MATHEMATICAL MUSINGS

Mark Thornber introduces RSA encryption and the mathematics of prime numbers

Modern encryption methods are based on the mathematics of prime numbers. The basic idea is that knowing the product of two large prime numbers does not make it easy to find the original numbers. As a challenge, I multiplied two prime numbers and the answer was 700179159413. Can you find the two numbers using only a simple calculator? The obvious method of trying every possible factor would take a very long time. A Python program to try all odd factors took 0.1 seconds.

```
import time
n = 700179159413
start = time.clock()
i = 1
notfound = True
while notfound:
    i = i + 2
    if n%i == 0:
        notfound = False
end = time.clock()
print(i, n//i)
print(end - start)
```

There are more efficient ways to attempt to solve this problem, but all the known methods involve a lot of systematic trials. The time taken increases to many years for the very large primes used in internet encryption.

FOLLOW-UP QUESTIONS

- How can we find very large (500 digit or more) prime numbers quickly?
- Our program uses the built-in Python function **pow** to do the hard work. How does this calculate enormous powers quickly?
- Running the program with a message 'Hello' shows that each letter always gives the same encrypted number. This makes the method vulnerable to frequency analysis. How can we avoid this?
- Since everybody knows the public key, how do you know the encrypted message comes from the person who claims to have sent it? How do we know it wasn't intercepted and replaced with a different message from some mischief-maker?

Government agencies from all major nations employ many mathematicians and computer scientists to find ways of working more efficiently with primes.

Using RSA

The first of the modern methods using primes is called RSA, after the inventors Ron Rivest, Adi Shamir, and Leonard Adleman, who published their method in 1978. To use RSA you find two very large prime numbers, p and q, and keep them secret. You then calculate the product, $n=pq$. Next, you find another prime d, between max (p,q) and n. The numbers n and d are your public key. You tell them to everybody. People can now use this public key to encrypt messages to you, but only those who know p and q are able to decrypt them.

The inventors illustrated the method with some small numbers: Suppose $p=47$ and $q=59$, so $n=2773$, and we now choose $d=157$. Now represent your message as a series of numbers smaller than n – e.g. by using ASCII codes. 'Help' becomes 72,101,108,112. For each number N in the message, replace it with $m=N^d$(mod n), the remainder when N^d is divided by N. 72^{157}(mod 2773)=288 etc.

To decode, we need to know p and q. First we find a number e so that $de=1$ (mod $(p-1)(q-1)$), and then calculate M^e (mod n) for each number in the message.

```
message = “Help”
d = 157
n = 2773
listmessage = [ord(letter) for letter in
message]
code = [pow(x,d,n) for x in listmessage]
print(code)
```

The fact that we can always find e, and that this gives us back the original message, follows from the mathematics of prime numbers discovered more than 250 years ago. In our simple case $e=17$ and $288^{17}=72$ (mod 2773), etc.

Mark Thornber has been a maths teacher at Durham Johnston School for the last 25 years. Mark has been interested in the mathematical parts of computing since owning his first ZX81.

GREG MICHAELSON PROFESSOR OF COMPUTER SCIENCE

ASSIGNMENT: SHARING OR COPYING?

We think of variables as name/value associations, abstracting from address/bit pattern associations, but what exactly are values and what happens when we assign them?

To begin with, let's recall that, for most languages, memory is typically divided into the stack (for variables and sub-program formal parameters) and the heap (for compound data structures like arrays and records).

THE STRANGE CASE OF C

Like many languages, C distinguishes **array structures**, where every element is the same type, and **record structures** (`struct`s), where fields may be of different types. C further distinguishes **structure variables** from **structure pointer variables**. For structure variable declarations, space is allocated for the structure on the stack, and the identifier is an alias for the first element. However, for pointer variable declarations, space is only allocated on the stack for an address. Subsequently, structure space must be allocated explicitly for the variable to point to.

Inconsistently, C does not permit array variable assignment, but does permit `struct` variable assignment, which is by copying. Now, if a `struct` has an array variable field, then it will be allocated stack space along with the rest of the `struct`. Thus, whole arrays may be copied on the stack through `struct` variable assignment.

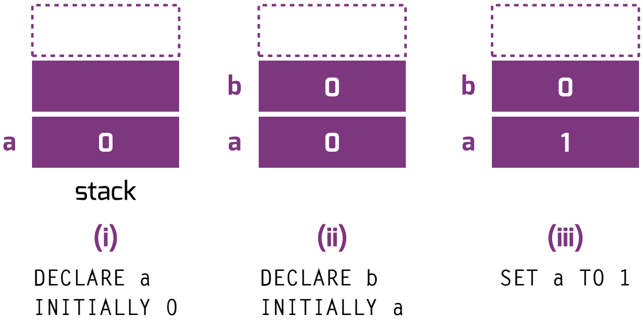
Suppose that variable identifiers are aliases for addresses on the stack. Let's consider the effects of:

```
DECLARE a INITIALLY 0
DECLARE b INITIALLY a
SET a TO 1
```

shown in **figure 1**.

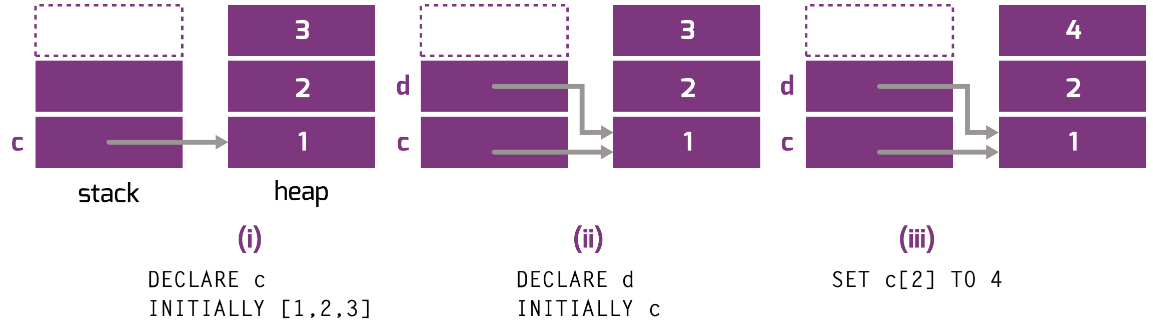
a is initialised to 0 (i). After b is initialised from a, each has their own copy of 0 (ii). Subsequently, changing a doesn't change b (iii).

Figure 1: integer variable to variable assignment



After assigning a base type from a source variable to a target variable, assigning to the source doesn't change the target

Figure 2: array variable to variable assignment with address sharing



```
DECLARE c
INITIALLY [1,2,3]
```

```
DECLARE d
INITIALLY c
```

```
SET c[2] TO 4
```

Array variable to variable assignment with address sharing. After assigning a structure type from a source variable to a target variable, changing via the source changes the target

Assigning array variables: sharing

Let's assume that the value of an array variable is the address of the first element of the array. Suppose, first of all, that assignment between array variables copies the array address. **Figure 2** shows the effects of:

```
DECLARE c INITIALLY [1,2,3]
DECLARE d INITIALLY c
SET c[2] TO 4
```

After the first declaration, c's value is the array's address in the heap (i). Initialising d from c copies that address, so c and d appear to share the array (ii). Thus, changing an element of the array via c will be picked up if the array is accessed via d (iii).

This is the approach taken by Java and, in effect, Python.

Assigning array variables: copying

Suppose instead that, for an array assignment, the whole array is copied. **Figure 3** shows the effects of the previous program fragment.

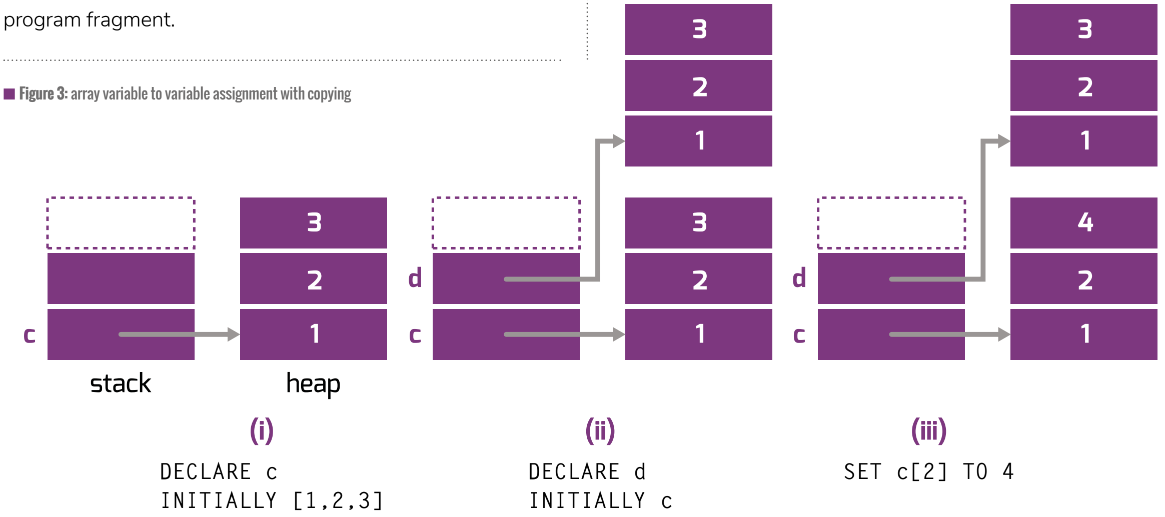
As before, c's value is a heap address (i). But now, d's value is the heap address of a copy of the array addressed by c (ii). Thus, changing an array element via c doesn't affect d (iii).

Few contemporary languages, apart from C, support copying assignment of structures.

Ancient languages, like FORTRAN and Pascal, distinguished sub-program parameter passing by value, i.e. copying, from passing by reference, i.e. sharing, but these are misnomers. In all programming languages, assignment, of which parameter passing is one form, is implemented as bit pattern copying. The key question is what the bit patterns represent, which in turn depends on the language and the implementation. (HW)

Greg Michaelson is interested in the design, implementation and analysis of programming languages, especially for multiprocessors. He has taught programming for nearly 40 years.

Figure 3: array variable to variable assignment with copying



```
DECLARE c
INITIALLY [1,2,3]
```

```
DECLARE d
INITIALLY c
```

```
SET c[2] TO 4
```

Array variable to variable assignment with copying. After assigning a structure type from a source variable to a target variable, changing via the source doesn't change the target

COMPETITIVE LEARNING

Programming perseverance produces prizes: when your learning objectives include 'to win a grand'!

STORY BY Neil Rickus

At the recent Scratch conference in Bordeaux, Mitch Resnick, Professor of Learning Research at MIT Media Lab, Director of the Lifelong Kindergarten research group and Head of the Scratch Team, used his keynote presentation to talk about his team's guiding principles: projects, passion, play and peers (**helloworld.cc/2tOO9g7**). While 'programming perseverance produces prizes' may not be quite as aspirational, the use of competitions in my teaching of computer science has allowed students to undertake a range of engaging projects, collaborate extensively, and visit locations across the country to showcase their work.

Combined motivation

Papert, in his writing on Constructionism, argued that students learn best when they are engaged in making things for others. Combining this motivation with the possibility of feedback from external judges, and success on an international level, competitions can facilitate enhanced student outcomes, engagement and motivation. Many competitions have a cash prize available for the winners, and introducing a unit of work with the phrase "the top prize is ten thousand pounds" is an extremely useful hook – however, care needs to be taken to manage expectations and ensure this is not seen as a substitute for well-planned, engaging lessons.

The competitions listed here have been used successfully as part of my computer science teaching with primary- and secondary-aged children. Each competition has several different focus areas, which gives children some flexibility in their choice of project, and encourages creativity. Students also need to plan their work, continually evaluate their progress, and collaborate with others – all essential experiences for developing their computational thinking skills.

TEENTECH AWARDS

URL: TEENTECH.COM

AGES: 11-18

Having guided groups of Year 7 students to the TeenTech Awards final on a number of occasions, I'm particularly fond of this competition. Students develop products linked to specific categories, which are judged externally, and feedback is given to all entrants. Finalists are asked to develop their project further, and invited to attend an event at the Royal Society in London. In addition to a final judging round, the event includes visits and talks from famous names in technology (Brain Cox, James May and Martha Lane Fox have all taken part over the past few years). The competition's patron, HRH The Duke of York, invites the eventual winners in each category to Buckingham Palace to receive a prize of £1,000.

ANIMATION 17

URL: HELLOWORLD.CC/2UEA3L7

AGES: 7-19

This popular competition, now in its tenth year, allows individuals and groups to develop a one-minute animation, which can be produced using a range of software, including Scratch. Students can work individually or in groups, with separate prizes of Amazon vouchers available for KS2, KS3, KS4 and KS5. Prizes are also available for a special e-safety category. Older children may also wish to consider the free digital-art camps run by 3Dami (**3dami.org.uk**).

PA CONSULTING GROUP'S RASPBERRY PI COMPETITION

URL: HELLOWORLD.CC/2H7SAWW

AGES: 8-18

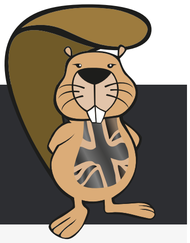
Image: PA Consulting Group

This competition has been running for five years, and the theme for 2016-17 was 'Innovation for all', which saw students producing products aimed at improving "the lives of people who have conditions that limit their ability to do things some of rest of us take for granted". The finals are held at the IET in London and the winners in each category receive £1,000.

BEBRAS

URL: BEBRAS.UK

AGES: 6-18



While the other competitions on these pages focus on students designing and developing projects, the Bebras Computing Challenge aims to introduce children to computational thinking. The competition runs in more than 40 countries, and the next event is scheduled to take place between 6 and 17 November, 2017. There are separate competitions for different age groups, and students can work on questions from previous years prior to the event. The top-achieving students are invited to a final round and celebration, in addition to being eligible to enter more challenging competitions to further develop their understanding. For information on Bebras, see page 68.

NESTA LONGITUDE EXPLORER PRIZE

URL: HELLOWORLD.CC/2V5S9G7

AGES: 11-16

Now in its second year, this year's Longitude Explorer Prize examined how the Internet of Things (IoT) can be implemented innovatively in healthcare. Finalists develop their projects in conjunction with IBM, as well as pitching their ideas and showcasing their prototypes to a judging panel. The winning school receives £10,000, with runner-up and individual prizes also available. The next Longitude Explorer Prize will run this autumn.

Introducing competition

How can you begin to use these competitions within your school? Start by visiting the URLs on this page. Most competitions have a mailing list or newsletter, and will send you a reminder when competition materials have been released or entries are open. Previous years' resources can be examined to determine where the competition will fit in with your schemes of work, and you could link the project with other curriculum subjects, such as science, or design and technology. Alternatively, a competition could be chosen as the focus area for a specific group of students, such as your school's Digital Leaders or as part of an after-school club. (HW)

RASPBERRY PI PIONEERS

URL: HELLOWORLD.CC/2JQNIJG

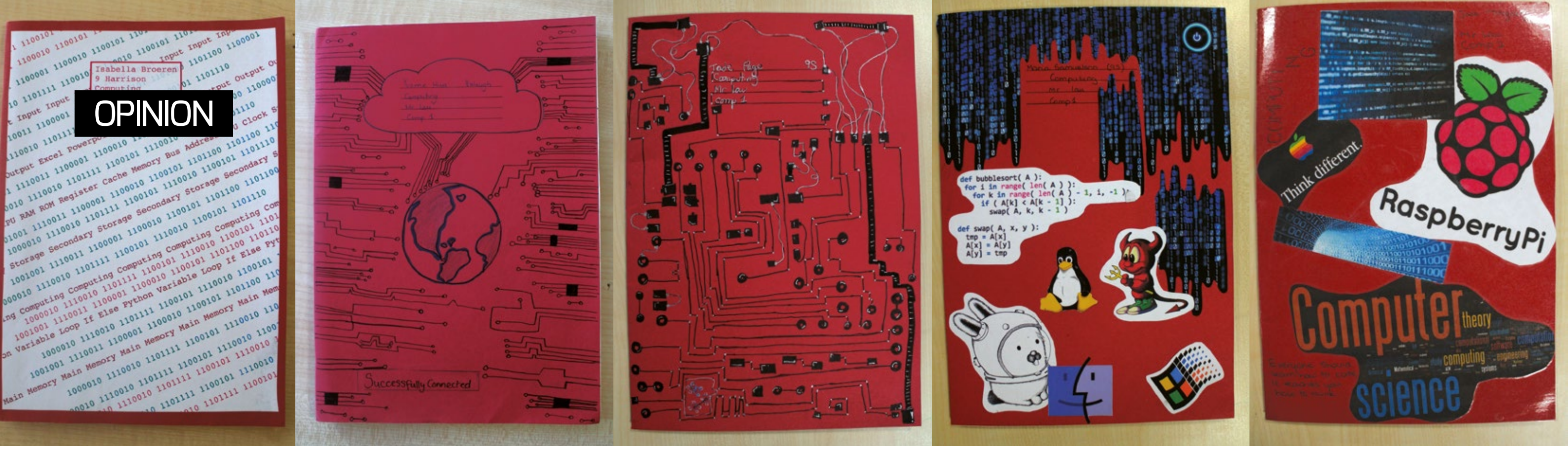
AGES: 11-16

Every few months the Raspberry Pi team sets a challenge for groups to get creative with technology. The latest challenge was entitled 'Make it outdoors', and the next tasks will be launched in September 2017.

OTHER COMPETITIONS

Information on a range of other competitions can be found on the CAS website at **helloworld.cc/2u4KCJk**, or in our FAQ (page 86).

Image: Raspberry Pi - Anna Gordon



WILLIAM LAU TEACHER

EXERCISE BOOKS IN COMPUTING

Give your students an indexed revision guide, full of their own coursework. Better still, let them write it themselves!

This is not an attempt to convince you to switch to exercise books, although some of you may be compelled to use exercise books having finished this article. Some readers will work in schools with standardised policies regarding the presentation of student work. Some schools demand that students use ring binders or exercise books. Four years ago, I joined a school which transitioned from students using ring binders to using exercise books. I was hesitant at first, given that I had used Google Classroom, GSuite and various VLEs before. However, after four years, I've developed a system that I'm happy with, and it is one of the keys to providing academic success for our pupils.

Judging a book by its cover

The first thing to consider is the front cover. In my first year, I made it compulsory for each student to decorate the cover of their book. It provided buy-in, and it also prevented students from using the excuse of mistaking their computing book for a subject which used the same colour. However, as the standard of decoration varied greatly, I now make this decoration optional, and make students aware of our high standards by showing them previous examples.

As books would get somewhat worn and tatty, we also bought plastic book covers. These kept the books in good condition throughout the year, and allowed students to tuck loose sheets into the back pocket between lessons.

We stipulate the key information that should be on the front cover, and we also state that there need to be two pages dedicated to a table of contents. This ensures that students can use the books as revision guides throughout the year.

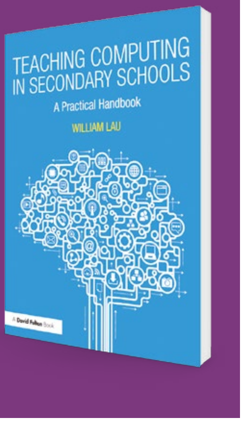
All pages are numbered and we give clear presentation guidelines such as:

- All written work to be completed in blue ink
- All drawings to be done in pencil
- All self-marking to be done in black ink
- All peer-marking to be done in green ink
- All teacher marking to be done in red ink
- All sheets to be stuck in flat, without folding
- All work to have a date and title underlined

We trim all our worksheets down to size so that they fit into the books. The images show some examples of exercise book work. We do not print out all their programs, but we certainly print out assessments. These usually have marksheets attached, and then students print out the improved copy.

TEACHING COMPUTING IN SECONDARY SCHOOLS

William has recently finished a book which provides a step-by-step guide to teaching computer science at secondary level. It offers an entire framework for planning and delivering the curriculum, and shows you how to create a supportive environment for students in which all can enjoy computing. The focus throughout is on giving students the opportunity to think, program, build, and create with confidence and imagination, transforming them from users to creators of technology.



■ Students have commented on how helpful their books are when completing NEA and controlled assessments

Reference books

Students have commented on how helpful their books are when completing NEA and controlled assessments, and they certainly look back over them during revision lessons. In many cases, it is much quicker to flick through their book to look up some key syntax than it is to search through hundreds of programs that they have written over the years. In the computing industry, many programmers use a notebook. As 80 percent of GCSE Computer Science is assessed through hand-written theory exams, it makes sense to use exercise books to organise students' thoughts and to assess students' handwritten work throughout the course.

> “ students can use the books as revision guides throughout the year

The use of exercise books also feeds into classroom pedagogy. It allows teachers to start every lesson at a desk away from computers with some form of low-stakes testing. Studies show that frequent, low-stakes, distributed and interleaved testing is beneficial for learning (Roediger, 2013). Often these low-stakes tests take the form of multiple choice questions. Many of these questions can be sourced from the quantum project at **helloworld.cc/2vkqgN0**.

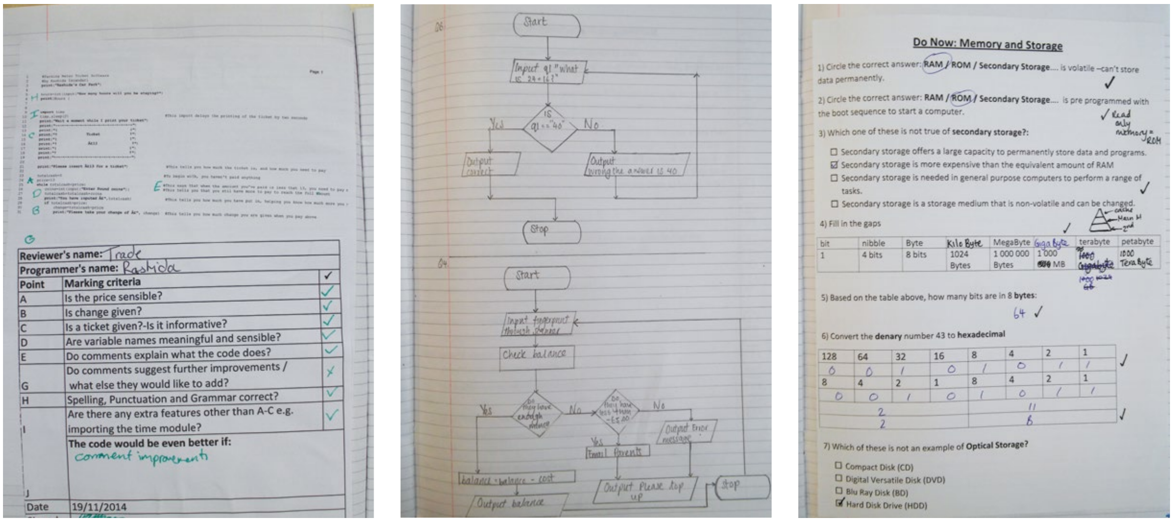
While I am not arguing that this is the only effective method of workflow, it has certainly worked well for our department and for our students. If you are interested in reading more about computing pedagogy, you may be interested to read some of my blog posts at **www.mrlaulearning.com**, or a more extensive and comprehensive guide in my book *Teaching Computing in Secondary Schools*, which is available for pre-order from Routledge or Amazon. (HW)

REFERENCES

Roediger III, Henry L. (2013). Applying Cognitive Psychology to Education: Translational Educational Science. Psychological Science in the Public Interest. 14(1) 1-3

William Lau is the Assistant Headteacher and Head of Computing at Greenwich Free School. He is a CAS Master Teacher, author of Teaching Computing in Secondary Schools, and winner of the Infosys Foundation USA/ACM/CSTA Award for Teaching Excellence in Computer Science.

■ Clear presentation guidelines make work in the books easy to follow



Page 1

G

Reviewer's name: Trade		
Programmer's name: Rashida		
Point	Marking criteria	✓
A	Is the price sensible?	✓
B	Is change given?	✓
C	Is a ticket given?-Is it informative?	✓
D	Are variable names meaningful and sensible?	✓
E	Do comments explain what the code does?	✓
G	Do comments suggest further improvements / what else they would like to add?	✗
H	Spelling, Punctuation and Grammar correct?	✓
I	Are there any extra features other than A-C e.g. importing the time module?	✓
J	**The code would be even better if:** comment improvements	
Date	19/11/2014	

Do Now: Memory and Storage

1) Circle the correct answer: [✓ RAM] / ROM / **Secondary Storage**…. is volatile –can't store data permanently. ✓

2) Circle the correct answer: **RAM** / [✓ **ROM**] / **Secondary Storage**…. is pre programmed with the boot sequence to start a computer. ✓ Read only memory = ROM

3) Which one of these is not true of **secondary storage?**:

- □ Secondary storage offers a large capacity to permanently store data and programs.
- ☑ Secondary storage is more expensive than the equivalent amount of RAM
- □ Secondary storage is needed in general purpose computers to perform a range of tasks. ✓
- □ Secondary storage is a storage medium that is non-volatile and can be changed.

cache
Main M
2nd

4) Fill in the gaps ✓

bit	nibble	Byte	Kilo Byte	MegaByte	Giga Byte	terabyte	petabyte
1	4 bits	8 bits	1024 Bytes	1 000 000 Bytes	1 000 MB	1000 Gigabyte 1024	1000 Tera Byte

5) Based on the table above, how many bits are in 8 **bytes**:

64 ✓

6) Convert the **denary** number 43 to **hexadecimal**

128	64	32	16	8	4	2	1
0	0	1	0	1	0	1	1
8	4	2	1	8	4	2	1
0	0	1	0	1	0	1	1
2				11			
2				B			

✓ ✓

7) Which of these is not an example of **Optical Storage?**

- □ Compact Disk (CD)
- □ Digital Versatile Disk (DVD)
- □ Blu Ray Disk (BD)
- ☑ Hard Disk Drive (HDD)

Use collaborative working in the classroom to demonstrate the real-life experience of a software developer

WHAT DOES A DEVELOPER DO?

Ever wondered what a software developer really does? Nicholas Tollervey explains…

STORY BY Nicholas Tollervey

I am a professional software developer.

I'm also called a software engineer, programmer and hacker (and ninja, superhero and rock star, but I'll come to that silliness later).

While I write code for a living, it is only one aspect of my job. My skill as a developer comes from using my creativity, experience and domain expertise to solve problems.

It's not all about the code

Often the most appropriate response doesn't involve writing code. A large part of my time is spent probing, questioning and playing devil's advocate for people who believe they have a technical problem, when in fact they need to address something else.

For example, working with NHS England, I was asked to write code to automatically create spreadsheets from complicated source data. What the client really wanted was an indication of patient waiting times. They had disguised the fundamental problem (I want to see current and historic waiting time information) as the need for more spreadsheets. We created a web-based dashboard to display the data, and an API so anyone could consume the data. The moral: solve the right problem – which may not be the problem you are given.

I also automate my work. I want to spend most of my effort on difficult problems, so I turn the repetitive stuff into software. For example, last year I helped create the Christmas food website for Marks & Spencer, and we could update the site with a single command. Everything was scripted in Python, with automated tests to check certain conditions were met during the deployment of the website, and Python would abort if problems were encountered.

By automating such activities, the knowledge essential for the smooth running of the company is expressed as code. Contrast this with the manual alternative: someone with the nerve-wracking and time-consuming job of deploying everything themselves. Often this was a single person and, if they went on holiday or left the business, such valuable knowledge was lost.

Most of the code I write consists of tests for the code that actually does something useful. Tests express how I expect my

Teachers need to challenge the stereotype of the socially isolated, young, male coding genius

code to run, and well-tested code is better because writing tests makes you reflect on how your code should behave. But tests are no guarantee that the code is bug-free! I may have missed a potential bug and failed to write a test to reveal it.

Tests are most useful as a way to deal with change: if I inadvertently make a breaking update – I add feature X, but this has an unforeseen negative impact on feature Y – then the tests for Y will fail.

Collaborative working

I rarely work alone. Often I will pair-program with another developer, where we take it in turns to write code (with the other watching, commenting and challenging their partner). It's an exercise in knowledge sharing and mentorship, and a way for experienced colleagues to work with new developers to help transfer knowledge about design decisions, conventions used in the code, and other business-related knowledge.

My colleagues and I constantly look for solutions on the internet. This is why open-source software is such a powerful idea: we have access to code, we can see how it works and we can learn from others. It's essential to develop the confidence to say "I'm not sure how to do this – let's see how others have solved a similar problem".

Knowing what to look for, assessing the responses from search engines, and evaluating how they apply to your problem – these are important skills that require practice. Memorising things in a programming context is a Sisyphean task and a complete waste of time: programming languages and libraries are constantly changing!

I ensure my code is properly commented in plain English so my colleagues (and my future self) understand what it does when they come across it on the internet. I name things such as variables, classes, and functions in a way that makes it clear what they represent. Comments explain intentions rather than implementation. It's important to know why something was written to see how it adds value to the wider system. Comments are also a good place to explain if the code in question was created in response to a subtle or hidden requirement.

> WRITING CODE ISN'T THE FOCUS – SOLVING FUNDAMENTAL PROBLEMS TO HELP PEOPLE IS

I also try to make my code as simple as possible. As British computing pioneer Sir Tony Hoare explains, "There are two ways of constructing a software design: one way is to make it so simple that there are obviously no deficiencies, and the other way is to make it so complicated that there are no obvious deficiencies. The first method is far more difficult."

Simple code is easier to understand, easier to think about, and easier to change. The opposite of simple code is called spaghetti code (and makes developers curse!).

Finally, my role is one of many involved in software development. I work with testers (who break things), platform specialists (who make our digital infrastructure work), UX specialists (who ensure a good user experience), designers (who make the application look good), business analysts (who represent the business) and scrum-masters (who facilitate the smooth running of the project).

Rejecting stereotypes

Contrast all of the above with a stereotypical software developer: He is a man in his early twenties. He lacks social skills, holds infinite technical details in his head and sits on his own in a windowless room pounding out code for hours on end. He says incomprehensible things like, "I've created a neural network to bypass the firewall and reverse the polarity of the mainframe's memory banks." Often, he is the focal point of some important activity since he's the only one with the skill to pull it off.

A side effect of such nonsense can be found in the job market. HR managers, playing to the stereotype, advertise for 'JavaScript Ninjas', 'Ruby Rock stars' or 'Python Superheroes'.

This is toxic stuff, and education is a key factor in challenging the damaging stereotype of the lone, young, male coding genius with a Messiah complex. Writing software is collaborative and cooperative, and involves input from a wide variety of people. Diversity is a strength. Writing code isn't the focus – solving fundamental problems to help people is.

If you are a teacher, I have a challenge: ask yourself how many of your lessons focus on group-based problem solving. If 'coding' means individual students progressing through a set of worksheets or an online course, try to replace it with collaborative group learning.

You're the experts in pedagogy, and I'd love to hear about ways in which you promote a community of collaborative problem solvers in your own lessons.

Best of luck! (HW)

Students working hard at this year's finals in Oxford

BEBRAS IS BEAVERING AWAY FURIOUSLY!

Chris Roffey explains how to enter your students into the annual Bebras Challenge for computational thinking, and how to use their new teaching resources

STORY BY Chris Roffey

Founded in Lithuania in 2004 by Professor Valentina Dagienė, the annual Bebras Challenge aims to enthuse school students from around the world in the fields of computer science, problem solving and computational thinking. Classic problems, of a kind that computer scientists regularly have to solve, are re-worked into accessible tasks, which often involve beavers ('Bebras' is Lithuanian for beaver)! Last November more than 1.6 million students from over 40 countries took part.

UK Bebras is a relative newcomer to this community, but it has been growing rapidly. In 2013 we ran a pilot with 21,000 students. In 2014 the Raspberry Pi Foundation and ARM generously sponsored us, enabling us to offer this opportunity to all students in the UK free of charge. In the same year, Oxford University became the National Organising Body for UK Bebras. In October last year, Google took over the sponsorship, asking us to produce an ambitious plan for expansion, and then funded us to follow it.

In 2015, Oxford University ran a finals weekend for the 10-18 age groups, hosted by Hertford College and the Computer Science Department. This was immensely successful, so this year the university expanded the event, running it over two weekends instead of one, with twice the number of students invited.

Bebras is absolutely free for schools and students. If we were to charge, schools might only enter their top sets, and this would undermine one of our main aims – alerting talented students to an important skill they may possess. We want schools to enter, if at all possible, all of their students. A student who has, for example, a specific learning difficulty, but is a great problem-solver, might be discovered and celebrated, and exciting new opportunities opened up for him or her.

Traditionally, when the UK has wanted to find talented problem-solvers, we put a difficult crossword in a national newspaper. The Bebras Challenge is a far sharper tool, and is already unearthing some amazingly talented students. At the finals in Oxford

UK BEBRAS AGE GROUPS

Kits	6 - 8
Castors	8 - 10
Juniors	10 - 12
Intermediates	12 - 14
Seniors	14 - 16
Elite	16 - 18

AN EXAMPLE BEBRAS PROBLEM

Agents Boris and Bertha communicate using secret messages.

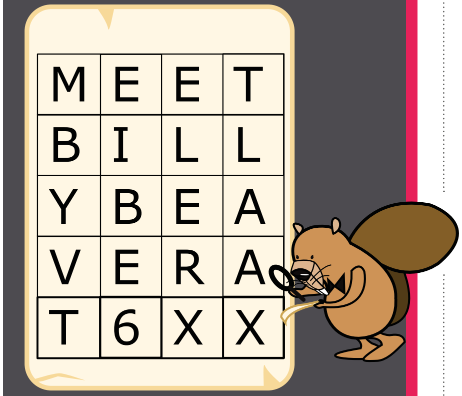
Boris wants to send Bertha the secret message:

MEETBILLYBEAVERAT6

He writes each character in a 4 column grid from left to right and row by row starting from the top.

He puts an X in any unused spaces.

The result is shown below:



M	E	E	T
B	I	L	L
Y	B	E	A
V	E	R	A
T	6	X	X

Then he creates the secret message by reading the characters from top to bottom and column by column starting from the left:

MBYVTEIBE6ELERXTLAAX

Bertha then uses the same method to reply to Boris. The secret message she sends him is:

OIERKLTEILH!WBEX

QUESTION :

What message does Bertha send back?

over the last two years, we have been astounded by the number, and difficulty, of problems some of the UK's talented students can solve, accurately and quickly. This experience has been inspiring, and makes us even more determined to make Bebras available to as many young people as possible.

Why enter your students?

There are many reasons why schools might want to take part .

Bebras can enthuse students in computational thinking and computer science. The tasks provide a baseline test that is relevant, targeted at computer science, and can be used to show progress.

The tasks are easy to administer. There is a simple upload procedure for students' details, a two-week window when students can take part during their normal computing lessons, and there is no software to install. We do the marking and provide detailed results, certificates and an answer book with full explanations and details on how the tasks relate to computer science – all for free, and within two weeks of the challenge.

The competition takes place in November, and can provide a focus for a teaching unit in computational thinking in the autumn term.

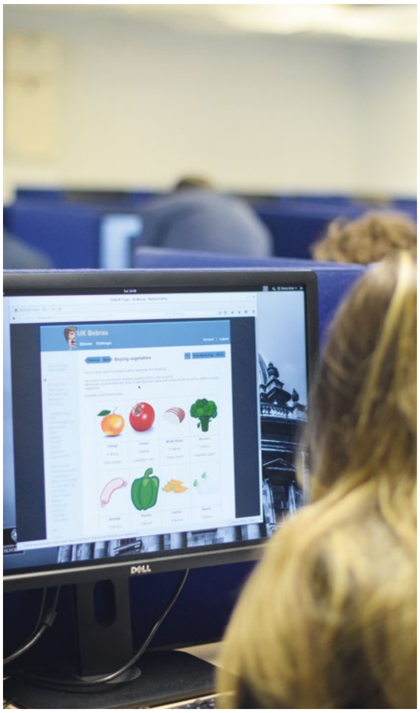
It's not just for two weeks!

The problems set are all archived for schools to use at any time during the year. Last year 81,000 UK students took part during the Bebras Challenge itself, but the resources were used by more than 100,000 students during the rest of the year! You can easily have a look at the interactive, auto-marked challenges by visiting **bebras.uk** and clicking on the Past Challenges button halfway down the page.

Future developments

In March this year, thanks to the kind sponsorship of Tata Consultancy Services, we ran a pilot follow-on Challenge for the top ten percent of achievers in the UK Bebras Challenge. Again, these resources have been archived for schools to use with all their students throughout the year. A number of other countries in the Bebras community are thinking along the same lines, and have expressed an interest in collaborating with us. Working together, we plan to ensure that, over future years, the quality of tasks remains of a consistent, high standard. It is hoped that, when we are fully established, other organisations that run challenges or competitions might approach us about inviting some of their students to take part in this new challenge, so that other routes of entry become available. To find out more, visit **tcsocc.uk**.

ANSWER: OKIWILLBETHERE!



In another collaboration, the UK and Germany are working on the development of a Bebras App for students, to run on mobile devices, and available by the end of September. The UK is also developing a Schools' Platform, which will enable registered School Coordinators to register teachers at their schools, as well as registering their students. The teachers will then be able to build their own self-marking challenges for their classes, made up from as many or as few of our archived tasks as they wish. Selecting suitable tasks should be easy, as all our tasks are now being categorised with a two-dimensional system based on the computational thinking skills they require, and the relevant computer science domains, as well as important key words. This new facility will be continuously improved, with more of our past catalogue added as time goes by. (HW)

LINKS

Bebras UK: bebras.uk

TCS Oxford Computing Challenge: tcsocc.uk

GOTO FOR EDUCATION?

The GOTO statement is outdated, and usually represents poor programming practice, but does this mean there is no educational value in it? I think not!

STORY BY Paul Powell

```
PRINT "What is your name? "
INPUT name
sayname:
SAY "hello " + name
PRINT "Shall I repeat that?"
INPUT answer
IF answer="yes" THEN GOTO sayname
```

Figure 1: a GOTO can be used to make loops like this do/while loop

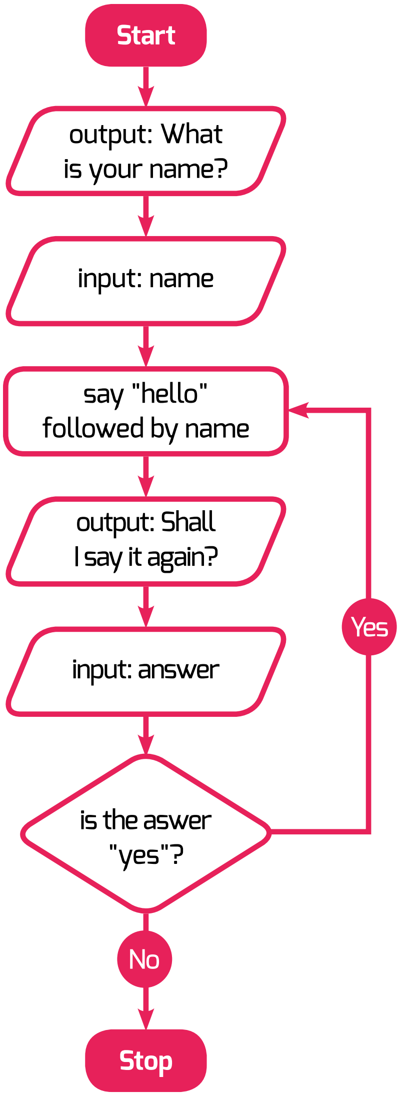

Start
output: What is your name?
input: name
say "hello" followed by name
output: Shall I say it again?
input: answer
is the aswer "yes"?
Yes
No
Stop

Figure 2: the same program as a flowchart - note the identical structure

In many subjects, we teach students basic ideas, approximations and, on occasion, outright lies. We do it to help students to grasp fundamental concepts before moving on to a subtler understanding in later years. This is often based on historical developments in understanding. For example, we teach that matter is made up of atoms which are indivisible. Students then learn about subatomic structure, and that under specific circumstances one element can become another. Later they learn that protons, electrons and neutrons are made up of quarks, and so it goes on.

In science, we also show the scientific method by contrasting Priestley's phlogiston with Lavoisier's oxygen, or geocentric theory with heliocentric theory. Of course, any professional scientist would be ridiculed if they believed in atoms that were indivisible, an orbiting sun or phlogiston. That doesn't mean that the ideas don't have educational value.

What is a GOTO?

A GOTO allows you to jump to another line in your code that you have labelled (or a line number if your language supports these). This works like a flowchart: you are drawing an arrow to the box you want to go to next. It is conceptually simpler than a loop, which requires us to analyse the problem at a higher level before implementing the loop. It also means that we can code directly from a flow chart, rather than reverse engineering a flow chart to contain the loops we need.

Before there were loops in code, we used GOTO to repeat sections. Before there were statement blocks, we used GOTO to skip or run sections of code. Today, when we use our modern programming tools, they get translated back into jumps in machine code.

A missing link

The first great thing about the GOTO command is that it is easy to understand. The second is that it can be used to explain what loops and other items of structured programming actually are (see **figure 1**). They provide an easy bridge from simple sequences to branching and iterating. When debugging, students can follow the code and say 'It shouldn't go to here – it needs to go to there instead'.

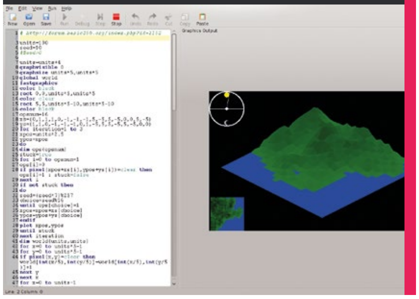
Of course, in time, students will leave behind the GOTO. As they advance onto structured programming and object orientation they will see it as quaint and simplistic, like counting on their fingers or rhymes that tell them how to form letters.

Caveat

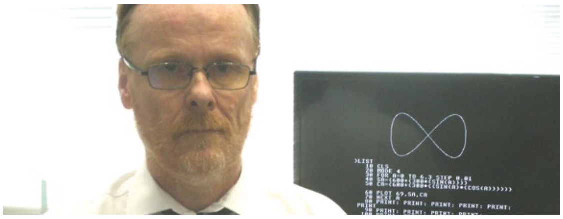
I'm writing the schemes of work for introducing text-based programming next year, based on my thoughts here. I've not run a class through it yet. I'll write another article to let you know how it went! (HW)

TRY IT OUT

Some modern languages (notably Python and Java) do not use GOTO. BASIC 256 is an open-source project designed for education. It has some useful features, such as embedded text-to-speech, easy graphics, music, and loading images. Download for Windows, Mac or Linux at **basic256.org**.



```
# http://forum.basic256.org/index.php?id=2112

units=100
seed=10
#Seed=0

units=units+4
graphvisible 0
graphsize units*5,units*5
global world
fastgraphics
color black
rect 0,0,units*5,units*5
color clear
rect 5,5,units*5-10,units*5-10
color black
opsnum=16
xs={0,1,1,1,0,-1,-1,-1,5,-5,5,-5,0,0,5,-5}
ys={1,1,0,-1,-1,-1,0,1,-5,5,5,-5,5,-5,0,0}
for iteration=1 to 3
xpos=units*2.5
ypos=xpos
do
dim ops(opsnum)
stuck=true
for i=0 to opsnum-1
ops[i]=0
if pixel(xpos+xs[i],ypos+ys[i])=clear then ops[i]=1 : stuck=false
next i
if not stuck then
do
seed=(seed*3)%257
choice=seed%16
until ops[choice]=1
xpos=xpos+xs[choice]
ypos=ypos+ys[choice]
endif
plot xpos,ypos
until stuck
next iteration
dim world(units,units)
for x=0 to units*5-1
for y=0 to units*5-1
if pixel(x,y)=clear then world[int(x/5),int(y/5)]=world[int(x/5),int(y/5)]+1
next y
next x
for x=0 to units-1
```



```
>LIST
   10 CLS
   20 MODE 4
   30 FOR A=0 TO 6.3 STEP 0.01
   40 SA=(600+(300*(SIN(A))))
   50 CA=(600+(300*((SIN(A)*(COS(A))))))
   60 PLOT 69,SA,CA
   70 NEXT A
   80 PRINT: PRINT: PRINT: PRINT: PRINT:
PRINT
   90 PRINT: PRINT: PRINT: PRINT:
```

RICHMOND MATHEWSON TEACHER

BACK TO BASIC

Richmond Mathewson explains why he is teaching BBC BASIC with BBC Microcomputers in 2017

Over the last few years I have run summer courses where I have introduced children aged 8 to 16 to LiveCode (**livecode.org**), as a way to get them programming quickly.

While LiveCode allows beginners to rapidly produce simple programs, it has certain drawbacks, including a lack of transferable skills to take to other programming languages and environments. Educators attempting to introduce children to programming with block-based approaches such as Scratch often encounter the same problem: children become extremely adept at producing programs with Scratch, and Scratch alone.

Formative experiences

I began to reflect on my own history in computer programming, which started with punching holes in Hollerith cards, and then progressed to FORTRAN IV. Programming really hooked me when my school obtained a Research Machines 380-Z, and at university I spent a year delving into Pascal. In 1989, I bought a BBC Master Compact and spent all my free time learning BBC BASIC, mainly because I was not interested in the games that came with the computer, and because I wanted the computer to do something for me.

A strong foundation

I use LiveCode to develop in-house programs for content delivery in my EFL school in Bulgaria. I began to suspect that, while it could be used to teach programming, it should be preceded by a programming language and environment that does not conceal the building blocks of programming behind objects such as buttons and fields.

BBC BASIC filled this slot for a number of reasons. Excellent, free BBC Micro emulators are available for Mac OS, Windows, Linux and Android. This means that children can program using BBC BASIC at home, as well as at school.

The BBC Micro does not have a GUI. As soon as it is switched on it is ready for programming – there is no entry barrier and no distractions. This forces a 'let's do some programming' message, rather than the 'let's play games' messages from computers with graphical user interfaces. They do not work at the speed of light, and they have a direct connection to the user. Nothing is hidden behind icons or inside colourful folders.

There's no copy and paste option, so children really have to pay attention to what they are doing, unless they want to keep retyping lines of code.

In four or five lines of code, I can illustrate the building blocks of coding in an easily digestible fashion, rather than having code hidden behind blocks or objects.

Boosting confidence

With BBC BASIC, my students rapidly completed new programming tasks. They even began to explore BBC BASIC at home, expanding their skills and experience beyond our classroom projects. Many of the children attending the classes had previously used Scratch. Those children have all said that while Scratch is fun, it is like a toy compared with what we could do with the BBC computers.

When we moved on to LiveCode after using BBC BASIC, the students seemed more confident. I noticed that none of them came up with the usual questions, along the lines of "Why doesn't this button do something?"

The children have just finished duplicating all the programming they did in BASIC in LiveCode, so they are aware of the similarities and the differences between the two languages. Next week we are going to start producing simple animation with LiveCode. It made me very happy to hear two children ask, "Can we try that in BASIC as well?" The answer was, of course, "Yes, why not?" (HW)

Richmond Mathewson has worked at all levels of education from primary to university, and now runs his own English language and programming school in Bulgaria.

YEAR 9: FROM ZERO TO MAKER IN SIX WEEKS

Sue Gray took a hands-off approach with her teaching so her students could have a hands-on approach to their learning

STORY BY Sue Gray

For the last half term my Year 9 students have been playing around with micro:bits. We began with a couple of starter lessons, and I put together a Getting Started presentation to introduce them to the possibilities of the equipment. They learned how to make a simple light display, display and scroll text, and edit the Fortune Teller game from the BBC website.

Pick your own project

After that it was hands off: no more teaching – just exploring, playing, developing ... and making. I'd made sure that I had most of the bits and pieces needed to create a lot of the Tech Will Save Us projects (**techwillsaveus.com**). I created a list of projects that included the ingredients for each make and put this into our Google Classroom. Students were then told to find a project!

At first, the students were reluctant to branch out on their own – they were so used to having me stand at the front of the class and tell them what to do, and to all be doing the same thing. They were uncomfortable with working independently, so I encouraged them to try out the Bourne To Code projects (**helloworld.cc/2v2TwaR**) at first. I needed a Post-it note stuck on my forehead that said, 'Read the instructions!' They were not keen on working without my guidance!

On-screen coding and off-screen making with the micro:bit

Getting stuck in

After four lessons, they got the idea. Yay! I should have been drinking tea with my feet up. Instead, I was in constant demand to delve into my cupboard and bring out Sugru, tin foil, copper tape, card, etc. They loved the freedom and creativity, the teamwork and, finally, the satisfaction of making something.

We had a lesson where it seemed the whole class wanted to make the pocket pet. I ran out of Sugru by the time two classes had completed this project! I also discovered that Sugru is not reusable. I think I'll encourage the micro pet project in future – it's cheaper and less messy! We also found that Sugru sticks like crazy, and I now have at least two micro:bits that may or may not work again.

> WE HAD A LESSON WHERE IT SEEMED THE WHOLE CLASS WANTED TO MAKE THE POCKET PET

TECHNOLOGY WILL SAVE US

Technology Will Save Us is a small UK company producing simple 'make it yourself' kits and online tools to lower the barrier facing children and teachers who want to get started with the physical side of digital making. Their micro:bit projects are available online at **helloworld.cc/2v2HQEX**.

Taking steps

Then we had a lesson where everyone discovered the step counter. Elastic bands were in high demand this time. I quickly realised that the potential for chaos might outweigh the learning objectives as students asked if they could go off around the school, "... to count how many steps it takes to get to Science." We made a list of routes and destinations, and students were told to work in pairs as they gathered data about the number of steps each one took to get from A to B. The data was then put into spreadsheets and they used the results to calculate averages. It was still chaotic, but at least there was some structure!

Everyone had a great time exploring the micro:bit, and I know that if I said we were going to revisit this topic, all the students would be up for it – and they would even love to make more spreadsheets! (HW)

TEACHING SCRATCHJR TO YEAR 1

Matthew Wimpenny-Smith explains how to use ScratchJr and tablet computers to get five- to seven-year-olds started with coding

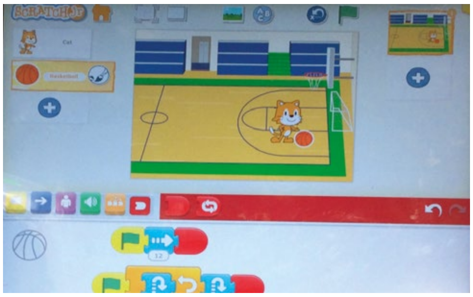
STORY BY Matthew Wimpenny-Smith

I am sure many of you, like me, have been using Scratch with seven- to eleven-year-olds for many years. The same team that brought us Scratch has also produced an iPad app called ScratchJr, which is designed to teach five- to seven-year-olds to code. It's available for Android tablets and Chromebooks, too.

Teaching with ScratchJr

I currently teach computing to two Year 1 classes for half an hour every week, and the ScratchJr app has given us a perfect introduction to coding. Luckily, I have enough iPads that each child can have their own, but I am sure pupils could easily work in pairs. I started by giving the pupils 'tinker time' – simply letting them play with the app while I observed what they could do. Very quickly, I could see the pupils working out that they could make the sprites move or act by connecting the program blocks together.

Can you tell the story of the three little pigs, make a virtual fish tank, or even code a cat to play basketball? You can with ScratchJr!



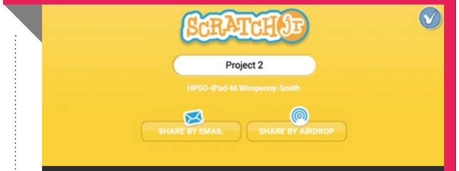
After a couple of sessions of 'tinker time', I started teaching the pupils about the different blocks and how they work together, using AirPlay to link my iPad to the classroom screen. Starting with the blue motion blocks and the green flag, I asked the pupils to predict what movements the sprite would make. Very quickly the class was ready for their first challenge: to make the cat dance using any of the blue motion blocks. Towards the end of the lesson I selected a few to play on the whiteboard.

Structured activities

During the next eight lessons, I gave the classes structured coding activities to follow, which involved using blocks from all the different categories. This helped the pupils to really grasp how the different blocks work together to make more and more complex code. Many of my ideas for these activities came from simplifying code that I had created using Scratch. I also found a number of activities on the ScratchJr website, **scratchjr.org**. At the end of each lesson I chose code from different pupils to show to the class.

> “ THEY COULD MAKE THE SPRITES MOVE OR ACT BY CONNECTING THE PROGRAM BLOCKS TOGETHER

FURTHER INFORMATION



The ScratchJr app is free on iTunes, Google Play and Chrome OS. I used shared tablets, so it was important that each pupil returned to the same device for every lesson, in order to access their locally stored projects. Projects can be named by tapping the yellow tag in the top right-hand corner of the app. Within that screen, users can also share their projects using email or AirDrop.

Developing their own code

Once the pupils had a good understanding of the coding blocks, they were able to move on to their own projects. During this term, the topic for Year 1 was Houses and Homes, so I asked them to make an animation to retell the story of the three little pigs and their different houses. In another lesson I showed them a fish tank, and the challenge was to make their own fish tank with all the fish swimming around.

My experiences in these lessons have shown that ScratchJr provides an excellent introduction to coding, which has given my pupils a good foundation for moving onto Scratch. (HW)

SMELLY CODE

Do we pass on best practice when we teach block-based programming to primary school pupils?

STORY BY Jane Waite

When we teach literacy in primary school, we use high-quality texts to model the features we would like our pupils to learn. In poetry, we demonstrate alliteration and onomatopoeia through the work of popular children's poets.

When teaching programming, do we do the same? Do we know who the good programmers are in the Scratch community? Are we even sure what good quality programs look like?

What is smelly code?

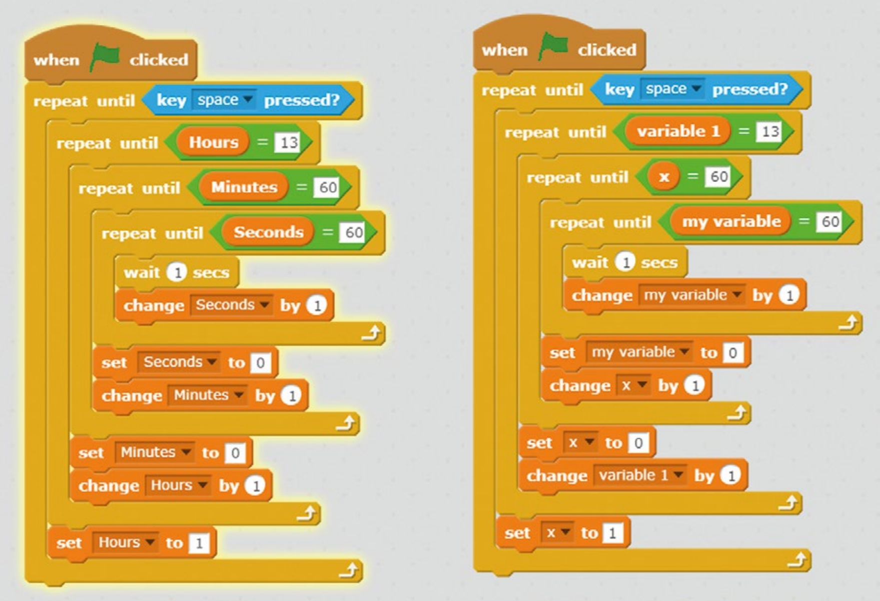
Smelly code is code that displays some feature that indicates an underlying problem with the program. The program may do some things correctly, but good code should be high quality and readable, too! Unreadable or convoluted code is much more likely to contain subtle bugs, hiding in those whiffy places.

Several researchers have investigated programs from the Scratch community, and concluded that most are smelly! One researcher looked at more than a million programs, while another checked 250,000 (see box for links). One of the research teams concluded that some of the things we want children to learn and make progress in for programming were lacking from the code sampled.

Don't worry – the researchers did not check all this code by hand! They have programs that do their checking for them. There is also the fabulous Dr. Scratch that you can use to analyse your own code.

Names matter

One example of a bad smell is a sprite that has not been renamed, and still uses its default name. Why is this important? Renaming a sprite might make our code easier to read, and indicates that we are thinking about what our program is doing. So, if the Scratch Cat is the narrator for an animation, rename it 'Narrator'. If the Elephant asks questions in a quiz, call it 'Quiz Master'.



```
when [flag] clicked
repeat until <key [space] pressed?>
    repeat until <(Hours) = [13]>
        repeat until <(Minutes) = [60]>
            repeat until <(Seconds) = [60]>
                wait (1) secs
                change [Seconds] by (1)
            end
            set [Seconds] to [0]
            change [Minutes] by (1)
        end
        set [Minutes] to [0]
        change [Hours] by (1)
    end
    set [Hours] to [1]
end
```

```
when [flag] clicked
repeat until <key [space] pressed?>
    repeat until <(variable 1) = [13]>
        repeat until <(x) = [60]>
            repeat until <(my variable) = [60]>
                wait (1) secs
                change [my variable] by (1)
            end
            set [my variable] to [0]
            change [x] by (1)
        end
        set [x] to [0]
        change [variable 1] by (1)
    end
    set [x] to [1]
end
```

Which is the easiest version to understand and debug? (There is a bug in one of the scripts.) Thanks to Barefoot Computing for the original project. Code remixed by Jane Waite.

Variables, broadcast messages, backgrounds, and costumes need renaming too. Just calling a variable 'a', or 'variable' makes our code harder to read and change. Children (and professional programmers) work on a program over several weeks, or take an old program as the basis to start writing a new one, so this matters a lot.

Superfluous stuff

Another bad smell is caused by redundant stuff: code blocks, variables, sprites, and messages left lying around, which don't do anything. If children added a casual full stop or a random word into a sentence, we might question it. Sometimes we do pull code apart as we are debugging, and leave code fragments at the side as we work out what the problem is, but when we save and share our code, we should tidy it up.

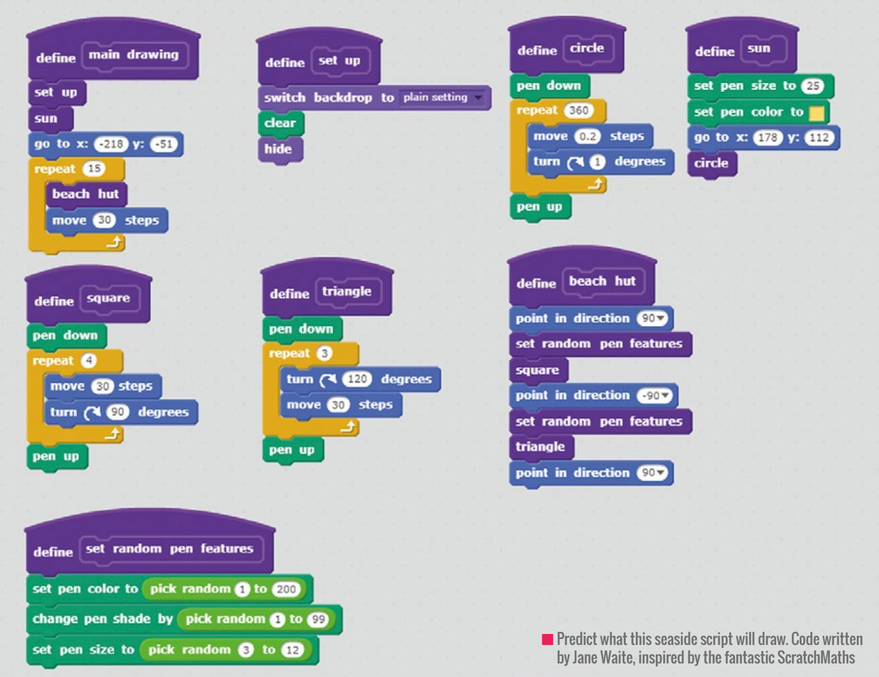
The researchers found other bad smells too, such as the same code being used for different sprites, and within the same sprite. Some repeated code for a single sprite can be implemented more elegantly, for example by using a loop. We can reduce repeated code by using clones, or by controlling the program using a main thread of commands, and synchronising activity through broadcasts and other techniques.

One of the smelliest aspects found by both researchers was very long scripts. Long scripts can be very hard to understand and debug, and often imply a lack of decomposition and design.

Make your own blocks

An aspect of Scratch that was missing from the sampled scripts was custom blocks. These purple blocks can be programmed to do a particular job for a sprite. Once a custom block is made, we can re-use it over and over again, and if it needs changing, we only need to change it in one place.

Using custom blocks demonstrates that we are breaking up our solutions into bite-



```
define main drawing
set up
sun
go to x: -218 y: -51
repeat 15
    beach hut
    move 30 steps

define set up
switch backdrop to plain setting ▾
clear
hide

define circle
pen down
repeat 360
    move 0.2 steps
    turn ↻ 1 degrees
pen up

define sun
set pen size to 25
set pen color to 🟨
go to x: 178 y: 112
circle

define square
pen down
repeat 4
    move 30 steps
    turn ↻ 90 degrees
pen up

define triangle
pen down
repeat 3
    turn ↻ 120 degrees
    move 30 steps
pen up

define beach hut
point in direction 90▾
set random pen features
square
point in direction -90▾
set random pen features
triangle
point in direction 90▾

define set random pen features
set pen color to pick random 1 to 200
change pen shade by pick random 1 to 99
set pen size to pick random 3 to 12
```

Predict what this seaside script will draw. Code written by Jane Waite, inspired by the fantastic ScratchMaths

FURTHER READING

Dr. Scratch:
drscratch.org

ScratchMaths:
helloworld.cc/2tYsNfC

Find your CAS regional centre:
helloworld.cc/2tZhiEt

Find your CAS Hub:
helloworld.cc/2t4IjCY

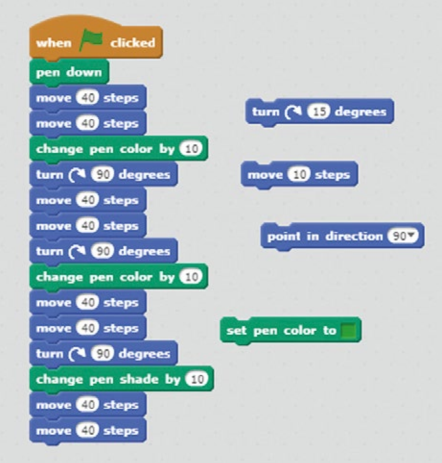
BCS Certificate in Computer Science Teaching:
helloworld.cc/2tZpq7V

Scratch research:
helloworld.cc/2sOnlw2
helloworld.cc/2tUFes7
helloworld.cc/2tZfxah
helloworld.cc/2uHZU4T

sized chunks. Learning to spot potential custom blocks and work with them is a very important habit, which gets children ready to learn more about procedures and functions in text-based languages.

Forever or until?

Another feature missing from the code was conditional repeats, or conditional loops. These commands allow us to control how a repeat ends based on another aspect of the program: for example, repeating an action until the space bar is clicked, or repeating something four times. Conditional repeats are important, as they show that we are thinking carefully about why and how we are looping. Someone reading the code can see clearly how the loop will end.



```
when 🏁 clicked
pen down
move 40 steps
move 40 steps
change pen color by 10
turn ↻ 90 degrees
move 40 steps
move 40 steps
turn ↻ 90 degrees
change pen color by 10
move 40 steps
move 40 steps
turn ↻ 90 degrees
change pen shade by 10
move 40 steps
move 40 steps

turn ↻ 15 degrees

move 10 steps

point in direction 90▾

set pen color to 🟩
```

Sniff out the smells. Initialise, tidy, reorder, and use a repeat. Code snippet written by Jane Waite.

Do you naturally model code using 'forever' commands, rather than 'repeat until', or 'repeat x times'? Does evolving rather than designing our programs contribute to lots of 'forevers' and not many 'untils'? The use of forever loops is seen by many experts as a bad habit.

Get ready before you go

Something else that is often missing from scripts is resetting stuff: setting variables to a start value, resetting sprites to a start position, clearing up any shapes we have drawn with the pen, etc. This is sometimes called initialisation, and it is an important idea. If you do not include initialisation in your Scratch code, and you run it twice in the same sitting, the values and states from end of your last run will become your subsequent start points. This can be very confusing, and you can't be sure what your program will do.

Creating programs that always do the same thing, whenever you run them, is a very good habit. In fact, some code needs initialising to work at all. Learning about initialisation is an important part of preparing children for the transition to text-based programming.

In your scheme of work, does the example code you use reset sprites to a start position? Do you explicitly teach resetting variables to a start value? Are there targeted tasks and guided exploration activities in your planning which reveal the importance of initialisation? Whatever form of scaffolding you use to teach programming, is initialisation covered?

So what, and what next?

Does bad-smelling code matter? There is plenty of research that says it does. If you give novice programmers smelly code, they will either have a poorer understanding of what the code does, or they find it more difficult to modify the code, depending on the type of whiff!

I don't have all the answers, but there is emerging research in this area. I think that as a teaching community, we are ready to talk about what makes good-quality block-based programs, and to look at what kinds of habits we are exemplifying in our teaching.

How about investing in your CPD? Ask your local CAS Regional Centre or CAS Master Teacher about a Primary Programming course, attend a CAS hub meeting, study for the BCS Certificate in Computer Science Teaching (primary or secondary), and find out more so you can help your pupils to write lovely, sweet-smelling code. (HW)

CYBERSECURITY: LEARNING THROUGH HACKING

Hacking can provide students with the context to understand how computers work, and teach them to write better code

STORY BY David Hill

I'm guessing that most people's knowledge of 'hacking' comes from movies and TV. As a result, when people talk about hacking, it often has negative connotations. At its heart, hacking is about understanding how something works. Yes, hacking can be misused, but we teach students many things that could also be misused. We don't think twice about teaching young people to write, drive, or understand science – just think of all the dangerous things you could create with that knowledge!

I am a hacker myself, and for that I blame my dad. I wanted to know how a radio works, so he gave me an old AM radio and we took it apart. He talked me through all the parts inside, and at some point we talked about shortwave. I liked the idea of listening to broadcasts from around the world, so we took a trip to Tandy, bought what we needed, and hacked the AM radio so it could receive shortwave stations. From this, it was a short jump to hacking my ZX81 games to slow down the pace of the 3D monsters.

What does hacking teach us?

How can hacking help us to learn? On a basic level, hacking provides an understanding of a range of cybersecurity issues, and leaves the hacker better able to protect themselves. In addition, being able to hack something requires certain skills – problem-solving, logical thinking, perseverance, and creativity – skills that are going to benefit students' learning and development in many areas, not just computing. You need to understand the rules on which the thing you are hacking runs. By mastering the rules, you can find ways to exploit them, allowing you to push the boundaries of what you can create.

A Raspberry Pi offers a safe (and legal) way for students to explore cybersecurity issues

This hacking mindset makes you a better computer scientist. A malevolent hacker only has to find one weakness in a computer system. If you want to protect a computer system, or the information in it, then you need to try and think of all the ways in which the system could be compromised. When you are writing a program, you need to understand how each command works, and how each part of the program interacts with all the other parts. Is there a way someone could make the program do something it shouldn't? Developing these skills helps a student to move from mimicry to mastery.

Is hacking legal?

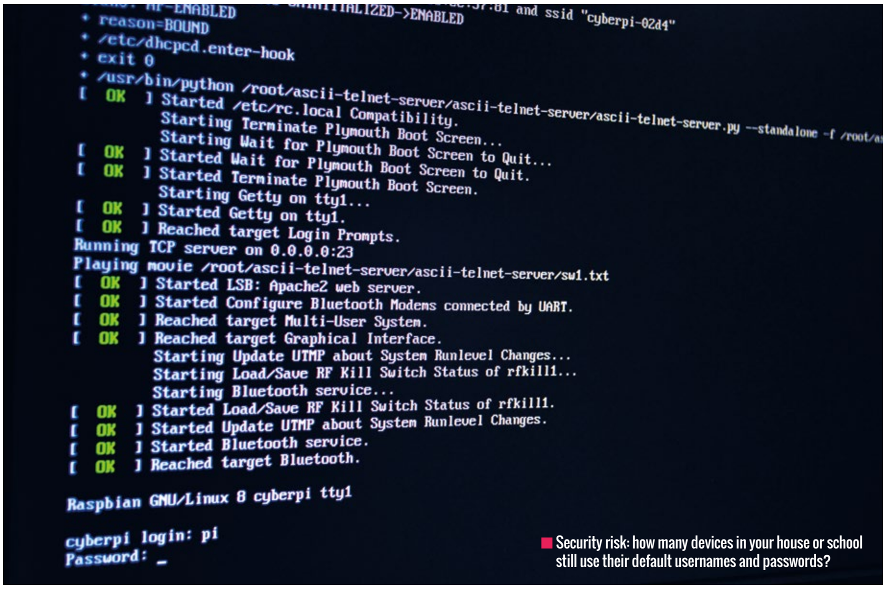
Hacking can be illegal. Simply put, under the Computer Misuse Act, it is illegal to gain unauthorised access to a computer system, or the material stored there. It is the 'unauthorised' part that is important, and this needs to be made clear before we start taking students on a journey into cybersecurity. There is a time and place for hacking, ethically and legally – whether this is the police investigating a suspect, or a business trying to recover information from their own computer system.

Before running any sessions with our students, to make sure that all our moral compasses are pointing in the right direction, we give them some scenarios to think about. Are there any circumstances under which it would be OK to hack a computer system? Are they technically possible? These scenarios can relate to real-world events, such as the controversy about bypassing an iPhone lock screen. How do the police usually get past a suspect's password? Answer: they ask them for it.

USEFUL WINDOWS TOOLS

Linux can handle most things on its own. However, if you want to use a Windows computer to hack your Raspberry Pis, this freeware will be useful:

Nmap:	allows you to scan an IP and see which ports are open
PuTTY:	allows you to use Secure SHell and Telnet to access other computers
FileZilla:	a piece of software that allows you to copy files between computers using FTP



```
      AP-ENABLED       UNINITIALIZED->ENABLED       :37:81 and ssid "cyberpi-02d4"
 * reason=BOUND
 * /etc/dhcpcd.enter-hook
 * exit 0
 * /usr/bin/python /root/ascii-telnet-server/ascii-telnet-server/ascii-telnet-server.py --standalone -f /root/a
[  OK  ] Started /etc/rc.local Compatibility.
         Starting Terminate Plymouth Boot Screen...
         Starting Wait for Plymouth Boot Screen to Quit...
[  OK  ] Started Wait for Plymouth Boot Screen to Quit.
[  OK  ] Started Terminate Plymouth Boot Screen.
         Starting Getty on tty1...
[  OK  ] Started Getty on tty1.
[  OK  ] Reached target Login Prompts.
Running TCP server on 0.0.0.0:23
Playing movie /root/ascii-telnet-server/ascii-telnet-server/sw1.txt
[  OK  ] Started LSB: Apache2 web server.
[  OK  ] Started Configure Bluetooth Modems connected by UART.
[  OK  ] Reached target Multi-User System.
[  OK  ] Reached target Graphical Interface.
         Starting Update UTMP about System Runlevel Changes...
         Starting Load/Save RF Kill Switch Status of rfkill1...
         Starting Bluetooth service...
[  OK  ] Started Load/Save RF Kill Switch Status of rfkill1.
[  OK  ] Started Update UTMP about System Runlevel Changes.
[  OK  ] Started Bluetooth service.
[  OK  ] Reached target Bluetooth.

Raspbian GNU/Linux 8 cyberpi tty1

cyberpi login: pi
Password: _
```

Security risk: how many devices in your house or school still use their default usernames and passwords?

Refusal to reveal the password could lead to a five-year jail term if the data on the phone involves national security!

The Raspberry Pi gives students the opportunity to explore some aspects of cybersecurity in a safe and legal environment. There is very little damage that can be done, that can't be rectified by re-imaging the SD card. Just remember to tell the students before starting the exercise that you are giving them permission to access the material on the Raspberry Pis.

Setting up Raspberry Pis to hack

There are plenty of configurations you could use to provide a target for hacking. You will learn a lot by creating a hackable Raspberry Pi for yourself, which is why I will not be providing detailed instructions. You will also find that your setup evolves over time, as students find different approaches to solving the problem. You could present your hacking challenge as a single exercise, or use each step as a chance to introduce an area for discussion and learning.

Our basic setup consists of a number of Raspberry Pis connected to a hub, which is then connected to the local network, or to a WiFi router. Each Pi has an LCD HAT that displays the IP address given to it when it is switched on. Each Pi runs a Telnet server and Web/MySQL Server.

Each student is assigned a specific Pi, and challenged to replace what is displayed on the HAT with a message of their choice. They then try to access their assigned Pi from a desktop computer or laptop. These are usually running Linux, but with some extra tools installed you could use Windows machines.

If the students are accessing the Pis over WiFi, the first thing they should do is look for the network and try to connect. This can lead to a discussion about SSIDs and encryption. The aim here is to think about what makes a secure password, as the password we use is one of the top five most common passwords. Not sure what the most common ones are? A quick Google search will give you the top 10,000 or more. Ask the students to think about how a list like this could be produced.

> “ REMEMBER TO TELL THE STUDENTS THAT YOU ARE GIVING THEM PERMISSION TO ACCESS THE MATERIAL

When they have access to the network, and have located an IP address, they can use Nmap to see that ports 22, 23 and 80 are open. Looking at the website gives nothing away (it is a simple holding page) and Telnet is just broadcasting Blinkenlights' ASCII Star Wars animation. Connecting via SSH is more rewarding, as they will be asked for a username and password. This gives us a chance to talk about default usernames and passwords. Think about all those webcams, printers, routers, etc. where people have never bothered to change the default login details. When the students have logged in, they find that the account has been restricted (we've changed the root login and file permissions), but they have access to an image file. When they work out how to copy files between computers, this will give them some encrypted text that conceals a web address. It's not very well encrypted – it just uses a simple Caesar cipher.

This takes them back to the web browser, where they find a couple more challenges. The password is handled through server-side code in a language they're not familiar with, to show the commonalities and demonstrate why some things should be handled on the server. Finally, they find a message board. Using some SQL injection to bypass the login box, this will allow their message to appear on the Raspberry Pi.

Completing this exercise leaves the students with a great sense of accomplishment, as well as a better understanding of cybersecurity. They've had the chance to learn new skills, and to discuss the moral and legal implications of their actions. As Alexander McQueen said, “You've got to know the rules to break them”. (HW)

10 TIPS SUPPORTING INDEPENDENT LEARNERS

Ideas for keeping your independent students motivated and engaged

STORY BY Rik Cross

1 Start small

When transitioning learners to more independent styles of learning, start by giving them the freedom to make decisions in the context of more structured activities. Projects that provide opportunities to personalise and extend on a basic project provide a structured environment in which learners can start to gain some autonomy. An example of this is the Code Club ChatBot talking robot project, in which learners are encouraged to make their own choices about their preferred robot character, the questions it asks, and the way in which it responds to replies (**helloworld.cc/2uP4ot0**).

2 Provide motivation

Learners will be more likely to see the benefit in what they are learning if they are given the opportunity to work on a project that interests them personally. The amount of freedom you are able to give learners will depend on the context, but being invested in the success of a project will help learners to stay motivated, and to have the determination to overcome any challenges that they may face. This doesn't have to mean giving learners the freedom to choose anything they like. Instead, you could give learners a choice of open-ended project ideas, for a range of interest areas.

3 Manage expectations

Giving learners the freedom to decide on their own project ideas can result in incredibly complicated project plans, often before learners are ready to tackle a large project. Rather than encouraging learners to choose something else to work on, help them to distil their ideas – discard

the more difficult or unnecessary aspects of the project and allow them to focus on something much more manageable. This will reduce frustration and help ensure that their project is seen through to the end.

4 Develop organisation

Encourage your learners to take responsibility for as much of the digital making process as possible. How will they manage their time? What are their priorities? Will they need time to undertake some research before starting work on their project? How and when will they review their progress? It may even help to agree on a process as a group, so that learners have a framework for organising themselves.

5 Provide resources

Working with independent learners means that your role as educator becomes that of facilitator. You are no longer the expert to whom learners immediately turn when they have a question. Be explicit about the places that learners can go to find answers to their questions: links to documentation, other example projects, online support forums. This works well when you're asked a question to which you don't know the answer yourself, as you're then able to model how you'd go about trying to find the answer. It also serves to make the point that the educator isn't an oracle who knows everything!

6 Create a collaborative environment

Working independently doesn't mean that learners have to work alone, and there are many ways in which your learners can support and learn from each other. Learners may have a problem that another member of the group has experienced, meaning that they are well-placed to offer support. Projects can be shared with others, either within a group or more widely through online communities. Sharing projects with others allows them to be seen by an authentic audience, which can be incredibly motivating. This audience can also offer valuable feedback, and often learners will pay more attention to detail, and are more willing to fix bugs, if they know their work will be seen by others.

7 Embrace debugging

One of the most difficult things about fostering independence is supporting learners when they hit a problem. Debugging is often where the most valuable learning takes place, as it provides an opportunity to deepen understanding and test assumptions. It's important that learners take control of debugging their own projects, so that they can support themselves later on in their digital making journey. Ask questions to point learners in the right direction. Encourage them to take a step back and think logically through a problem, decode error messages, and seek peer support.

8 Test regularly

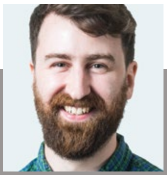
Debugging is made much more straightforward when a project is tested regularly. Testing each new project feature means that learners are more likely to pinpoint any problems with that feature. Encourage learners to follow a short, iterative development cycle, testing each step as they go. Learners may even find that their projects work in unexpected ways if they allow others to test their work.

9 Model failure

Being ambitious and taking risks will inevitably result in times when things don't go to plan. Digital making can often follow an unpredictable path, and learners will benefit hugely from being exposed to this process. When supporting learners, try not to gloss over mistakes that you make yourself, or times when you aren't able to fix a problem first time. Instead, be transparent about the process, modelling the resilience needed to work through these setbacks.

10 Celebrate success

At the end of this process, learners will have created something unique that they will rightly be proud of, and want to show off to others. Make time at the end of a project for learners to share their creations, and if possible to be reflective and share their experience with their peers – even if their project isn't finished. This is a great time to be positive and share successes, but it could also be valuable to allow learners to talk about challenges they faced, and how they tackled them. Your learners will be exposed to a range of interesting ideas that may provide a basis for their next digital making project! (HW)



Rik Cross
Rik Cross is a former secondary teacher, responsible for developing the Raspberry Pi curriculum. Rik enjoys learning new skills and is currently learning to brew beer.

BLUFFER'S GUIDE TO NEA TASKS

The new Computer Science GCSEs in England and Wales all have an assessed component requiring students to spend 20 hours on a practical programming project. Alan O'Donohoe, leader of exa.foundation, looks at the implications for teachers

What is an NEA?

NEA stands for Non-Examination Assessment. This requirement was introduced to replace previous coursework components. It is a practical component that is formally assessed as part of the GCSE, with a weighting of 20 percent of the final GCSE grade.

Why do the NEAs exist?

A large amount of Computer Science involves practical problem solving. While the new GCSEs include an exam that tests students' knowledge and understanding of programming, computational thinking, and algorithms, the NEAs allow students to submit some coded solutions to problems. There have been some historical problems with coursework, and the NEAs are intended to provide a more robust mechanism to ensure a level playing field for all schools. As the NEAs are relatively new, there are many procedural changes to assessment to implement. Schools may take a while to adjust to the new processes.

PLEASE NOTE:

The author neither works for, nor represents, any examination bodies. As such, the information in this guide is intended only as practical advice for a wide audience, not as an official guide to one GCSE NEA. Since examination bodies vary in their rules for the NEA, teachers should familiarise themselves with the specific rules governing their choice of qualification.

KEY:

[!] **Indicates areas with a significant degree of difference between different exam awarding bodies. Please check the details for the body that you are using.**

What will the students be assessed on?

Students are required to solve a sequence of programming challenges. These typically include a requirement to analyse each problem, design algorithms, then develop, test and evaluate their solutions to the challenges. Teachers will need to familiarise themselves with the specific requirements of the relevant specification.

Lack of familiarity with the requirements has, for example, led to some teachers stressing about two-dimensional arrays when the requirement of their specification is only for one-dimensional arrays, or worrying about teaching SQL when the requirements are only to write to and read from a file. While there is absolutely nothing wrong with students using these methods, the tasks have been designed to require only the skills listed for the NEA.

When are students expected to complete this?

Students are allowed 20 hours of supervised time to complete the NEA. The teachers can choose when to start and end the 20 hours, and even stop before the 20 hours if students have finished. However, the NEA tasks are not released until 1 September, and students' completed work must be marked and sent for moderation by 31 March. [!]

Schools can choose when to schedule the NEA sessions within that window, but NEAs can only be attempted during the terminal year of assessment, i.e. the same year as the exam. There is no requirement for schools to log every lesson allocated for the NEAs, but it would be sensible to do so. Teachers should plan ahead to consider what to do with students who complete their projects in less than the allocated time. If a student 'gives up' after just three hours, this could cause problems for the remaining 17 hours.

It would be wise to start the NEA as soon in the Autumn Term as possible, and complete it as late as possible. This would allow teachers the flexibility to plan for refresher lessons during the 20 hours, and allow for other unplanned circumstances such as teacher or student absence.

Teachers are free to run after-school programming sessions as long as they are not teaching to the NEA task

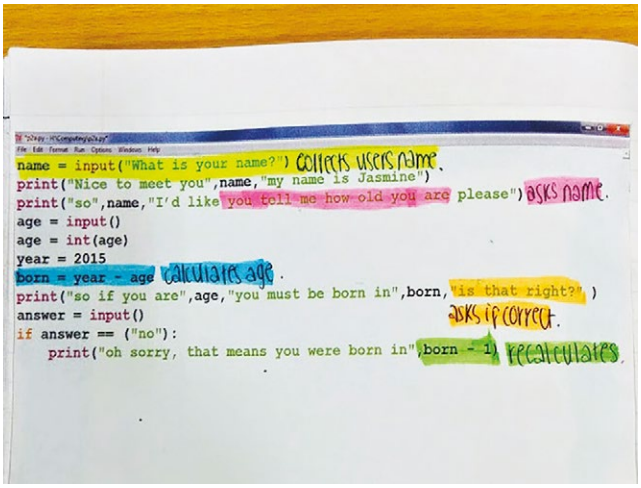
Splitting the 20 hours up over an extended period would also go a long way towards keeping students' motivation levels high. Their attention is likely to be sharper in the non-NEA sessions where teachers provide refresher programming lessons, since the students could then apply this to their NEA solutions.

How can teachers best prepare their students for this beforehand?

Throughout Year 10, plan for lots of practice with lots of stepped paired-programming challenges designed to slowly build confidence and competence. While it's tempting for teachers to focus purely on teaching systems theory in Year 10, doing so would seriously disadvantage students with the NEAs. Some teachers choose to split their timetabled lessons 50/50 between systems and programming; I chose a 20/80 split, with more time allocated for the programming element. In Year 11, a refresher course at the beginning of the autumn term will put students back onto a firm footing before starting the NEAs.

It would be wise to introduce students to the restrictions imposed by the NEA as early as possible in Year 10, so that it's less of a shock for them when they start the NEA. This would allow any Resource Banks to be developed throughout Year 10.

Students should be prepared before the NEA begins - teachers cannot give specific help or feedback during the NEA task



p2a.py - H:\Computing\p2a.py
File Edit Format Run Options Windows Help

```
name = input("What is your name?") collects users name.
print("Nice to meet you",name,"my name is Jasmine")
print("so",name,"I'd like you tell me how old you are please") asks name.
age = input()
age = int(age)
year = 2015
born = year - age calculates age.
print("so if you are",age,"you must be born in",born,"is that right?" )
                                                    asks if correct.
if answer == ("no"):
    print("oh sorry, that means you were born in",born - 1) recalculates.
```

Should teachers stop teaching programming as of 1 September?

No. However, it would be best to ensure that students have been properly prepared before then, as teaching to the tasks is strictly prohibited. If the NEA sessions are scheduled to be spread out across the autumn and spring terms, this would allow more opportunities for refresher lessons.

Can teachers offer after-school programming sessions?

There is absolutely no problem with this as long as teachers provide general support and advice with the programming requirements, and do not teach to the NEA. After 1 September, students are prevented from bringing in any electronic, written or printed resources from this additional support to their NEA sessions.

What if teachers spot something after the NEAs are live that they omitted from their teaching?

Again, as long as any teaching post 1 September does not include support or solutions to the live NEAs, there is absolutely no problem with this. For example, if after the NEAs are published the teacher feels that their students need to use a particular data structure that they have not taught, the teacher could plan a general lesson on data structures and include the required structure among several that are not required. However, the teacher should be very careful not to drop hints or suggest which data structure to use as this is classed as providing solutions.

What if the teacher has a very limited amount of time available to prepare students?

In much the same way as learning any other new skill (driving a car, drawing or learning the violin), it is possible to learn certain things in a reduced amount of time. However, to really learn problem solving, algorithms, computational thinking, and programming to a level sufficient for the NEAs takes hours and hours of repetitive practice. Anything short of this is unlikely to prepare students properly. In my experience, teachers and students who have not had sufficient preparation tend to look for shortcuts out of desperation, e.g. searching for solutions from others, or looking for the answers to an important test or exam. ▶

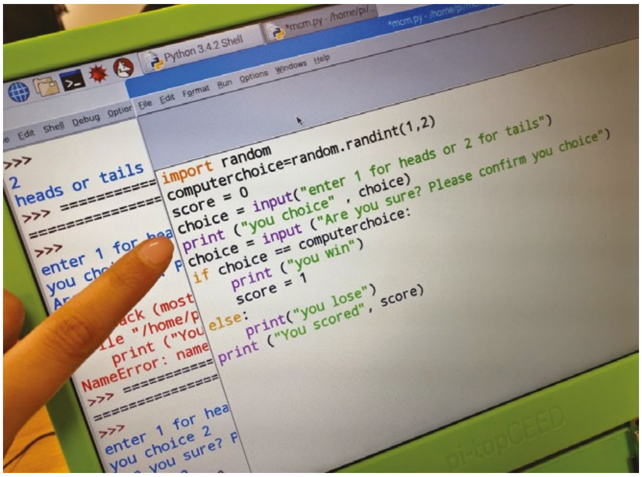
Can teachers find solutions to the live NEAs?

While this may be possible, it is a very dangerous path to go down and I would caution teachers against it. Teachers are prohibited from providing their students with solutions to the live NEAs. If the teacher subsequently discovers what they consider is a better solution to a problem than the one their students would consider using, it would be very tempting to show these solutions to the students.

It is worth stating that there can be many different approaches to solving NEA-type problems. Once you have seen one solution it's impossible to 'un-see' it, and hard to develop any alternative solutions. My advice would be to avoid looking at solutions to NEAs, and caution students against it as well. Also, some solutions shared in the past have contained errors. If you learn from these or share them with your students, you're not just committing malpractice, but also proliferating the errors.

How can teachers support students during the NEAs?

In the actual NEA sessions, support from teachers is strictly limited. Outside the NEA sessions, teachers can give generic advice and support, providing it does not refer to the NEA tasks or include solutions. Students may also do their own research outside the NEA sessions, but most exam boards prevent students from bringing evidence of this research into the NEA sessions – students may only use what they can remember. Teachers might decide to plan some non-NEA sessions during the 20-hour period for refresher lessons, e.g. file-handling practice, data structures, etc.



```
import random
computerchoice=random.randint(1,2)
score = 0
choice = input("enter 1 for heads or 2 for tails")
print ("you choice" , choice)
choice = input ("Are you sure? Please confirm you choice")
if choice == computerchoice:
    print ("you win")
    score = 1
else:
    print("you lose")
print ("You scored", score)
```

The resource bank can contain examples of the students' own solutions to problems set during Year 10

> THROUGHOUT YEAR 10, PLAN FOR LOTS OF PRACTICE WITH LOTS OF STEPPED PAIRED-PROGRAMMING CHALLENGES DESIGNED TO SLOWLY BUILD CONFIDENCE AND COMPETENCE

If a student asks how to do something 'generally', but it relates to the NEA, what should the teacher do?

As long as the teacher is only giving general advice and support, there is no problem with this. It would be worth clarifying this with students, and explaining that there are strict rules teachers have to follow, regardless of student difficulties.

What feedback can the teacher provide to students?

The simple answer is 'absolutely none'. There are strict rules limiting teachers from providing feedback to students. It is likely that you'll still want to provide students with some regular assessment feedback during the period of the NEA, so why not set a much lighter parallel project on a completely different theme, unrelated to the NEA? This would also serve as a useful teaching aid, but it would need to be carefully planned to avoid similarities to the NEA.

What resources are students allowed to use in the NEAs?

This is where the widest variations exist between examining bodies. For some GCSE qualifications, for example the OCR GCSE, no internet access is allowed at all during the NEA sessions, but students can be provided with a pre-prepared resource bank. After 31 August, nothing may be added to the resource bank, but resources may be removed. Eduqas/WJEC and Edexcel allow students to be provided with examples of syntax, but no coded solutions or examples of NEAs. For AQA, students are allowed full access to the internet during the NEA. Personally, I think that this is a distraction – I think that students should be prepared before the NEAs. If a student needs to search for help during an NEA, it shows poor preparation.

For some teachers, this limit on resources has been a major factor in choosing awards. If you're joining a school where the resource bank has not yet been prepared, you could probably use one from another school – providing it was prepared before 31 August. [!]

How should students reference resources (when allowed)?

This is quite simple. If students use something in their NEA that they have copied from a resource, they should reference it. If they already know it, they don't need to.

How will students manage without internet access?

Some exam bodies restrict use of the internet. [!] Having conducted controlled assessments with internet access being allowed, I'm not convinced it helps the students during a formal assessment. It can lead to laziness and the additional risk that solutions may be shared. Not having the internet available for the NEAs is a challenge, but it forces a different way of thinking and discipline – using the resources you have available to address the problems you're being asked to solve.

To prepare students for working on NEAs without internet access and using the resources provided, set some practice tasks before the NEAs to help them get used to the restrictions and develop good habits. This will also reduce any potential stress for students working under these controls.

What should an OCR resource bank include?

Teachers following the OCR GCSE course can put resources that students might find useful into a resource bank, up to 31 August. This includes wall displays, electronic resources, and paper-based resources, but not templates or writing frames. Where allowed, it is recommended that teachers provide students with the following reference materials:

- the students' own solutions to previous challenges set throughout Year 10 to remind students how they solved previous problems
- some carefully selected examples of previous students' projects e.g. from previous years
- reference materials reminding students of the syntax of the chosen programming language

Which resources should I buy for my students?

Thankfully there is a great variety of free downloadable resources available to support students (where allowed), so in theory you shouldn't need to buy anything. However, some publishers have released paid content which schools can purchase. Teachers who have purchased these have said they found them really helpful. Ask other teachers which resources they've found most useful.

How should we store the students' work?

All student work must be stored securely to prevent students from accessing their work outside the NEA sessions. Some schools have created exam accounts with restricted access to the internet, USB ports and publicly shared drives. It would be a good idea to get students into the habit of using these before the NEA starts.

Who should I ask for advice?

Make sure you ask the right people the right kind of questions. While you should always check with exam board representatives and documentation for official rules and regulations, asking other teachers as well will help you to find out how they are approaching the NEAs and preparing their students. However, you should be careful how much trust you place on advice from others who do not represent exam board officials – just like the advice in this article!

Isn't this a lot of hassle for just 20 percent of a student's final grade?

I think it's dangerous to assume that programming is only worth 20 percent, when computational thinking, algorithms and programming combined are worth 60 percent in total. In my teaching, I've managed to reduce the time required to teach theory down to about 20 percent, allowing about 80 percent of the time for computational thinking, algorithms and programming. The students were least familiar with the latter, so a 50/50 split would not have worked as well. A lot of the theory they can learn themselves without direct supervision from me.

Security

To maintain integrity, the task instructions and student work are not allowed to leave the NEA sessions. Students are not allowed to bring additional resources into the sessions. [!]

What if some schools decide not to follow the rules?

Through the Joint Council for Qualifications (JCQ), the exam boards have jointly said they will be conducting an unspecified number of centre visits this year, along with statistical analysis of the NEA results compared to exam results. Schools can expect a visit at any time, and the exam boards don't need an excuse to trigger a visit. (HW)

MORE INFORMATION

- Blog post on NEAs: **exa.im/helpwithnea**
- Teachers at Outwood Grange Academy Trust have compiled a helpful NEA resource: **helloworld.cc/2uHij4m**
- Chris Roffey has shared these Code Cards: **helloworld.cc/2vICM5U**
- Mark Clarkson's guide to Python: **helloworld.cc/2gQdGzm**
- OCR Guide to pseudocode and Python: **helloworld.cc/2vFa7BO**

GOT A QUESTION?

If you've got a question for our FAQ team, you can ask us on Twitter via **@helloWorld_Edu** or using the **#HelloWorld** hashtag. Alternatively, email us (**contact@helloworld.cc**) with 'Teacher Question' in the subject line.

YOUR QUESTIONS ANSWERED

Teaching computing and digital making requires a whole new set of skills, and understanding the various options and requirements can be incredibly challenging. Don't be afraid to put your hand up and ask us anything...

Q How do I introduce my students to coding on a mobile phone?

A Although many mobile devices are also functional computers, the reality is that the devices' software is often locked down to prevent users from running arbitrary code on their mobiles and tablets. This can make learning to code on a mobile tricky, and the potential young coder will have to resort to curated apps from the vendor's app store, or in-browser applications.

There are plenty of apps out there that can serve as introductions to programming. For very young learners there is **ScratchJr**, which is a mobile-friendly and simplified version of Scratch (**scratchjr.org**, featured on page 73). If you want your learners to experience Python, then apps such as **Pythonista 3** on iOS (**helloworld.cc/2sWQdCc**) or **Pydroid** on Android might be suitable (**helloworld.cc/2tzukaJ**).

There are also Android apps such as **DroidScript**, which can help students to learn JavaScript. As well as allowing students to build games, utilities and novelty apps, DroidScript also provides full programming access to the features of modern mobile devices, including GPS, networking, USB, camera, Bluetooth, accelerometer, text-to-speech, and voice recognition. The app provides a number of downloadable plugins for controlling external devices such as the micro:bit, LEGO Mindstorms and Arduino project boards, which encourage students to explore the world of physical computing (**droidscript.org**).

Although all these apps can help begin a student's journey into programming, the small screens, software keyboards, and file system restrictions mean that they are only really suitable for very simple scripts and projects. In reality, there is no genuine alternative, as yet, to learning programming on a fully fledged general-purpose computer.

Q HOW DO I KNOW WHAT MODULES ARE AVAILABLE FOR PYTHON, AND WHICH I SHOULD BE USING WITH MY STUDENTS?

A Like any good programming language, Python allows you to import modules into your code, so that you do not have to rewrite code that has already been produced by somebody else. Some modules are provided with every Python install – these are part of the standard library. Modules such as **math**, **random** and **time** are all included on install.

There are thousands of modules that have been produced by developers, all over the world, that are not included in the standard library. Finding the correct modules can sometimes be a bit of a challenge.

Often the easiest way to find a particular module is to use a search engine. Typing 'Python package for making games', for example, will provide a fairly long list of modules.

As for which modules you should be guiding your students towards, the short answer is 'whatever they need to complete their projects'. If they want to use graphics, then **PyGame** is a good choice. If they need to produce a GUI, then maybe **guizero** is what they need. Just be aware that different modules have different qualities of documentation. The better documented a module is, the easier it will be for your students to use.

Q HOW DO I TEACH THE PRINCIPLES OF CODING TO VERY YOUNG CHILDREN?

A It is possible for very small children to build basic programs and begin to understand the logic of programming. Many schools begin with physical toys such as **BeeBots** (or the newer **BlueBots**), which are robust programmable floor robots. Press the directional buttons to program the path of the BeeBot, and then set it free to see whether the program works. Pair kids up and have them take it in turns to navigate to a teddy (**bee-bot.us**). **Cubetto** is another coding toy, Montessori-approved and aimed at children of three and older. Coloured blocks are placed into a control board to guide a robot through various adventures and maze puzzles (**primotoys.com**).

Slightly older children might enjoy **Griditch**, described by its creator as the programming equivalent of an abacus. Place small electronically scannable tiles called 'grits' on a grid to pose a problem for a friend, and use more grits to solve it. Scanning the tiles links the physical world to an app where you can set conditions such as 'if touching enemy, lose a life' (**griditch.com**).

Scratch is one of the most popular applications for teaching kids to code, but many people are not aware of **ScratchJr**, which is a free tablet-based app aimed at kids aged five to seven years. The app sticks with the familiar drag-and-drop blocks interface, but removes much of the reading requirement by replacing the words on the blocks with simple symbols (**scratchjr.org**, featured on page 73). Sticking with tablets, children may also enjoy **LightBot**, a website where you write simple programs to navigate a robot around a 3D grid, lighting up squares as you go (**lightbot.com**).

Those looking for a creative challenge should check out *Hello Ruby* by Linda Liukas (featured on page 34), a book with accompanying unplugged exercises ranging from creating art with algorithms to finding out what's inside a computer (**helloruby.com**).

Q I've been told that Processing is as easy to learn as Python. Which language should I be teaching to my students?

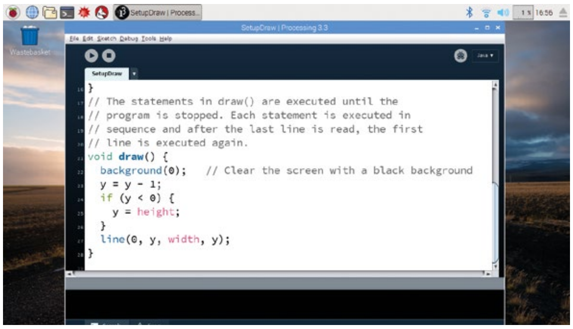
A Processing is a programming language and development environment that is geared towards learning to program in the context of the visual arts. The Processing language is based on Java, so if you're only familiar with Python, Processing will look completely different. For example, blocks of code are indicated using curly brackets, and each line of code is terminated with a semicolon. If you have any experience with Java or even C, Processing should feel familiar to you.

The Processing development environment (the program where you write and execute code) will also feel very familiar to educators and students who have used the Arduino development environment. That's because the Arduino team based their software on Processing. Like Arduino, the Processing development environment is appropriate for beginners; you write your code in the window and simply hit the Play button to see it run without much hassle.

As a learning tool, Processing is very well supported with documentation and examples. Its visual arts focus means that students who are interested in being creative with code and want to create things on-screen are given a powerful tool to experiment and iterate. And while Processing is geared towards visual arts, it is full-featured enough to take on pretty much any task that you'd expect from a programming language.

Even though Python isn't specifically designed for visual arts, there are Python modules such as **turtle**, **Tkinter**, **Kivy**, and **guizero** that enable you to use Python to draw on screen. There's also a new Python mode in Processing which allows you to use Python syntax in the Processing development environment.

Between Processing and Python, you have two very good choices for education. Since there's a lot of overlap between the two languages, I'd suggest trying out both and sticking with the one that's closest to the languages you're already familiar with.



```
}
// The statements in draw() are executed until the
// program is stopped. Each statement is executed in
// sequence and after the last line is read, the first
// line is executed again.
void draw() {
  background(0);   // Clear the screen with a black background
  y = y - 1;
  if (y < 0) {
    y = height;
  }
  line(0, y, width, y);
}
```

Q I'D LIKE TO MOTIVATE MY STUDENTS BY TAKING PART IN COMPETITIONS. CAN YOU TELL ME WHAT'S OUT THERE TO INSPIRE THEM?

A There are plenty of competitions and contests available for your students to participate in, depending on their age and ability.

For the younger students there are very few regular competitions, but they tend to pop up every now and again, run by various companies, organisations and government departments. It's always worth looking at the CAS competitions page to see whether any new competitions have been added (**helloworld.cc/2u4KCJk**). Some of these are featured on page 62.

The **Bebras** competition (featured on page 68) runs free annual challenges for students aged 6-18. You can take a look at past challenges, and find out more, at **bebras.uk**.

If you run a CoderDojo, take a look at the annual **CoderDojo Coolest Projects** event, where students can showcase the projects they have been working on (**helloworld.cc/2sWOhtv**). The Raspberry Pi Foundation also runs a termly challenge called **Pioneers** for students who would like to have a go at some digital making (**helloworld.cc/2jqNljG**). Then there is **Apps for Good**, which is a very popular general technology competition (**appsforgood.org**).

For older students there are lots of options. The **Cyber Security Challenge** will really stretch their abilities, investigating codebreaking and cryptography (**helloworld.cc/2t56RLc**). Then there are hackathons at various institutions all over the country, run in conjunction with **Major League Hacking** (MLH), which will really push your students to their limits (**mlh.io**).

Image: CoderDojo Foundation

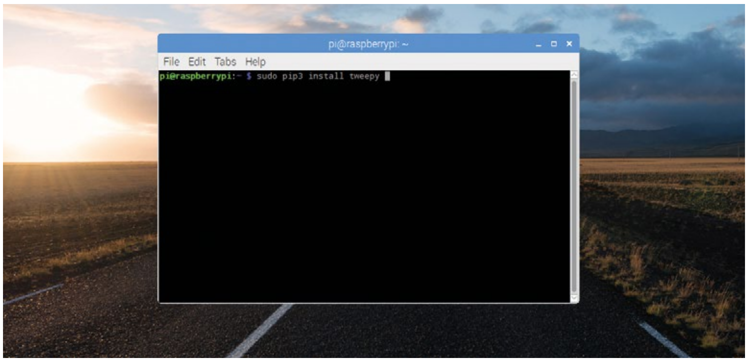
Q I have been told to install a Python package using pip. What is pip and how do I use it?

A Pip is a package manager, used to install Python modules onto your computer. It's a bit like an app store for Python modules, but all the modules are free and open-source. If you wanted to use Twitter from your Python code, you could use pip to install a module such as Tweepy.

In Linux, this would be as simple as opening a terminal or command prompt and typing:

```
> sudo pip3 install tweepy
```

If you want to learn more about pip, especially how to use it on Microsoft Windows, have a look at **rpf.io/pipguide**.



```
pi@raspberrypi:~ $ sudo pip3 install tweepy
```

Q I'd like to teach my students about the history of programming. Are there any resources you would recommend?

A We have a rich history of computer science and programming in the UK, much of which is preserved thanks to the efforts of volunteers at the National Museum of Computing at Bletchley Park (famed for its code-breaking operations during the second world war). With lots of hands-on activities as well as access to some very early computers, such as Colossus, the museum is well worth a visit with your students (**www.tnmoc.org/learn**).

The Centre for Computing History based in Cambridge, UK, also has some great learning resources on their website (**helloworld.cc/2tZzk9l**). In the US, the Computing History museum in Mountain View is certainly worth a visit (**www.computerhistory.org**) to learn about early programming and computers, and where better than in the heart of Silicon Valley?

For some fun background information you can access the free educational YouTube channel *Crash Course Computer Science* videos on Alan Turing and early computing (**helloworld.cc/2t5I1Lb**). There are also books, such as *The Programmer's Odyssey* by Gareth Eaglesfield (**helloworld.cc/2v3qxAn**, reviewed on page 89). To go really old school, the original Usborne programming books from the 1980s, including *Introduction to Computer Programming* and *Practical Things to do with a Microcomputer*, have been made freely available as PDFs by the publisher (**helloworld.cc/2u50FqP**).

Image credit: Jason Long

Q What is Git and how can I use it?

A Git is a type of version control system (VCS). It allows you to store versions of your files as you write them, and then easily revert back to previous versions at any time.

Git's real power is that it can be distributed. This means that rather than one person working on the files, whole teams of people can work on them. As each person makes changes to the files and pushes them to a remote server, others can pull in those changes.

If you want to learn more about using Git and remote systems such as GitHub, have a look at **rpf.io/gitguide**.

Q I NEED TO TEACH SQL. HOW CAN I GET SUITABLE SOFTWARE INSTALLED ON MY NETWORK?

A Many students are now required to understand the basics of SQL database queries, which can lead to some problems when identifying which software to use.

It's always worth remembering that Microsoft Access is capable of performing SQL queries, although it has its disadvantages. It is a Windows-only program, and more complicated than it needs to be when all you need to do is query a couple of linked database tables.

There are several online tools that you can use to help students learn about SQL: **sqlfiddle.com** is a site that will allow you to build your own tables, and then provide them to your students so they can experiment with various queries.

There are also sites, such as **codecademy.com**, that provide online tutorials for your students; guiding them through the process of creating tables, adding data and querying the data.

Q IS THERE A SIMPLE WAY TO PRODUCE GUIS IN PYTHON?

A Yes! Lots of schools have begun to introduce Tkinter for GUI programming, but found the syntax cumbersome and difficult to work with. The **guizero** library has been created by an experienced teacher and removes the unnecessary complexity from creating a GUI. It is simple to install even without administrator access (so you don't have to pester the school technicians) and the most basic of Hello World GUI apps can be written in just four lines of code. It is designed to be simple enough to be used by eight- to ten-year-olds, yet powerful enough for use in complex projects (**helloworld.cc/2u9Xn6o**).

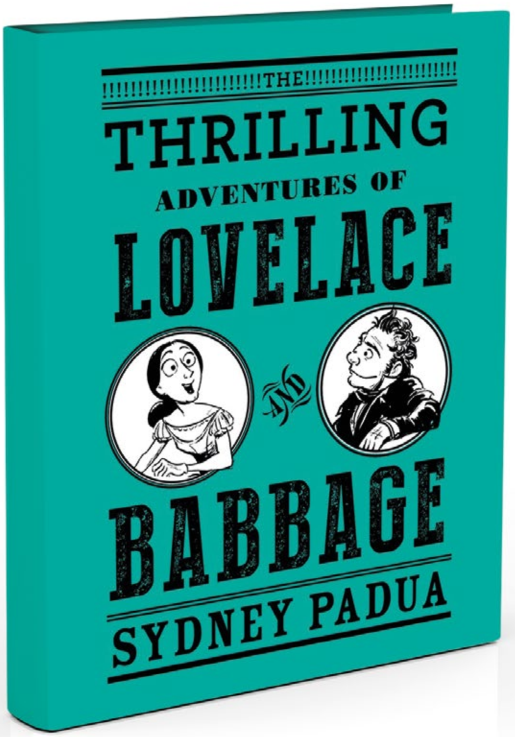
Remaining with the simplified theme, but with different customisation options, is an alternative library called **appJar**. This removes unnecessary boilerplate code to simplify making GUIs with a variety of attractive layouts (**appjar.info**).

Q I need to teach programming, but I'm not allowed to install anything on the school's network. What do I do?

A First of all, you should try to explain, to both your network team and your senior management, the importance of having access to the appropriate tools. Asking an educator to teach programming without the required software is like asking an English teacher to teach Shakespeare, but not allowing them access to any of the plays' texts.

If, however, you have no luck with this, there are still a few choices available to you. If you teach younger students, then online platforms such as **Scratch** will be perfectly suited to introducing students to programming. As they progress, you can switch them over to an online Python IDE. **Trinket.io** is an online IDE with plenty of functionality, and would certainly be sufficient for teaching students how to program up until they start their GCSEs.

If you want students to explore HTML, CSS and JavaScript, then all you need is a text editor. **Notepad** in Windows is fully capable of writing code that can then be run in a browser. For debugging purposes, you have online access to the developer tools of all the popular browser suites, including Chrome, Edge and Safari.



THE THRILLING ADVENTURES OF LOVELACE AND BABBAGE

A graphic novel about an alternate reality with Ada Lovelace and Charles Babbage, and a completed Analytical Engine!

INFO **WRITTEN/DRAWN BY** Sydney Padua | **PUBLISHER** Pantheon Graphic Novels (Penguin Books)
PRICE £12.99 | **ISBN** 978-0141981536 | **URL** 2dgoggles.com

Sydney Padua is a graphic artist and animator, who was asked to draw a short web comic about Ada Lovelace for Ada Lovelace Day. If you know Ada's history then you realise this didn't have a terribly happy ending, so drawings imagining an alternative, better, and more thrilling universe were added, and such was the hunger for more details that she expanded them into this novel.

The book contains a number of stories showing an alternate universe where Charles Babbage's Analytical Engine does get built (called the Difference Engine, because it sounds cooler), and Ada doesn't die at 36. There are lots of footnotes, linking the alternate universe to our own, extensively and meticulously researched, building up a fantastic picture of Ada, Charles, and Victorian society. My favourite story has Marian Evans having her novel (the one written by her 'friend' George Eliot) analysed by the Difference Engine to correct all the errors, but the (shorter) story in which George Boole visits and is asked 'Tea or coffee?' and responds 'Yes' would make a great introduction to Boolean logic!

> “ You'll learn a lot about Ada and Charles, Victorian society and the principles of computer engineering from this book

Endnotes add more detail, including cartoons from Punch (the one showing the Old Lady of Threadneedle Street bailing out irresponsible banks is from 1890!) and other Victorian publications.

Ada is depicted as a pipe-smoking, leather-clad hacker, equally at home with a wrench fixing a crash in the innards of the Difference Engine, or hand-punching a program to print out kitten pictures to calm Queen Victoria down; while Charles comes across as the archetype of the absent-minded inventor.

The end of the book includes sections on Babbage's anticipating carriage, which (would have) shaved seconds off additions (modern fast adders use similar techniques to save nanoseconds), programs, punch cards, and logic and loops, all of which add detail to the Analytical Engine. A moving epilogue has Ada and Charles talking as they wander through the towers of cogs inside the Engine.

You'll learn a lot about Ada and Charles, Victorian society and the principles of computer engineering from this book. There are lots of computer in-jokes to be found, and you'll have a lot of fun as you learn. (HW)

THE PROGRAMMER'S ODYSSEY

A fascinating life in computer programming

INFO **BY** Garth Eaglesfield | **PUBLISHER** Pronoun | **PRICE** £4.15 (Kindle)
ASIN B01LTHF1S8 | **URL** helloworld.cc/2v3qxAn

In the preface, Garth Eaglesfield (born in 1946, degree in Philosophy) points out that the digital age 'still only spans a single working life' and his book gives 'a totally personal account' of such a life, describing the events that happened and his thoughts on them.

His 'drift' into programming makes an entertaining introduction, and there are bits of basic background which can be skipped (although look at the section on log tables if you've never used them, or for nostalgia if you have). There's a lot about COBOL, which played a large part in the odyssey; the phrase 'lipstick on a pig' makes an appearance; and a large part of the second half is devoted to a system called Tuxedo, with which he's been heavily involved.

My odyssey started eight years or so after his, and I have a completely different set of events, thoughts and opinions, but despite this I found it fascinating. We don't tend to look back very often (the last issue of Hello World was an honourable exception), but if we don't we're ignoring a valuable resource. (HW)

COMPUTATIONAL THINKING

Incisive insight and practical advice – a 'must-read' for students

INFO **BY** Karl Beecher | **PUBLISHER** BCS | **PRICE** £19.99 |
ISBN 978-1780173641 | **URL** shop.bcs.org

How do the small coding challenges attempted by beginners link to the marvels of software that drive our world? The connection, sadly missing from most programming guides, is made through computational thinking: a problem-solving framework of value to all, regardless of experience.

Do not be confused by the book's title – this is not another 'getting started' guide. It will resonate most with those who have started already.

In two distinct but related sections, Karl Beecher distils the core concepts involved. Logic provides the starting point; algorithms are the means by which decisions are stitched together. But while algorithms lie at the heart of computational processes, there is no algorithm for solving problems. Rather, our eyes are drawn to the creative art of CT.

Decomposition and other key ideas, such as pattern recognition, generalisation, and abstraction, are introduced in plain English. No prior familiarity is assumed or required, making this less a synopsis, and more a blueprint for teaching.

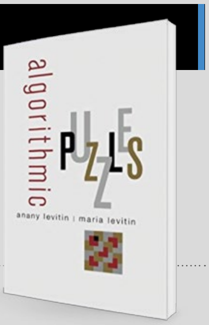
Practical explorations in Part 2, using Python, map to the concepts of CT. Expect 0 to 60 coding in a couple of chapters!

Throughout, Karl Beecher makes connections hitherto unseen, raising awareness and cutting through the frequently bewildering language of the discipline. As he admirably demonstrates, there is a lot more to software development than simply teaching people to code. (HW)

ESSENTIAL READING:

Test yourself with these thoroughly challenging puzzle books!

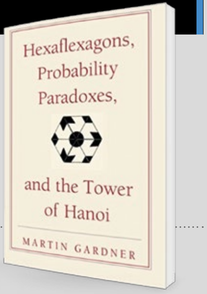
ALGORITHMIC PUZZLES



BY Anany and Maria Levitin
PUBLISHER Oxford University Press (USA)
PRICE £15.99
ISBN 978-0199740444
URL helloworld.cc/2vimLqN

The Levitins have put together a useful collection of puzzles, most of a mathematical nature. What sets this collection apart is the authors' commentary, which emphasises algorithmic as well as logical approaches to solving the problems.

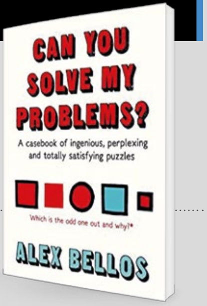
HEXAFLEXAGONS, PROBABILITY PARADOXES, AND THE TOWER OF HANOI


Hexaflexagons, Probability Paradoxes, and the Tower of Hanoi
MARTIN GARDNER

BY Martin Gardner
PUBLISHER Cambridge University Press
PRICE £12.99
ISBN 978-0521735254
URL helloworld.cc/2vnNrWF

A classic. Gardner wrote a monthly column on mathematical games and recreations for Scientific American for 25 years: this book collects and expands many of his best pieces. Gardner has an engaging, accessible style, and there's much here that would inspire young computational thinkers today.

CAN YOU SOLVE MY PROBLEMS?
A CASEBOOK OF INGENIOUS, PERPLEXING AND TOTALLY SATISFYING PUZZLES


CAN YOU SOLVE MY PROBLEMS?
A casebook of ingenious, perplexing and totally satisfying puzzles
ALEX BELLOS

BY Alex Bellos
PUBLISHER Guardian Faber
PRICE £8.99
ISBN 978-1783351152
URL helloworld.cc/2wlc17H

Bellos has a weekly puzzle column in the Guardian - this book is a collection of some of the best of the thought-provoking puzzles that have been featured. Some of these require some cunning, but others provide a great starting point for programming exercises.

EVENTFUL INSPIRATION: PART 2

Our guide to running a successful tech-based event, packed with advice from people who are already running theirs

Are you considering running a hackathon, techmeet or Raspberry Jam? Getting like-minded people together to encourage digital making, coding, and community? In issue 2 of Hello World, we looked at the first steps: getting started, booking a venue, issuing tickets, and thinking about equipment and activities. In this issue, we take an in-depth look at the activities you could offer at your event.

WHAT IS A RASPBERRY JAM?

Raspberry Jams are independently organised community events where people get together to share knowledge, learn new things, and meet other Raspberry Pi enthusiasts. Jams provide opportunities for people to get involved in digital making, develop their abilities, get together, have fun, and socialise. They are usually free or very cheap to attend.

This series of articles is based on The Raspberry Pi Foundation's guide to running a community event. *The Raspberry Jam Guidebook* is aimed at Jam organisers, but the advice it offers will be helpful to anyone setting out to run a tech-based community event. Ben Nuttall, Community Manager at the Raspberry Pi Foundation, compiled the guidebook. He collected advice from existing Jam organisers throughout the UK, on everything from finding a venue to managing your finances, and from planning your activities to managing social media. Packed full of great, first-hand advice, the guidebook is designed to help you to run the best event you possibly can.

Your unique event

Your event could feature a range of activities. From the informal drop-in, where people bring their own projects to work on, to more formal workshops and talks, the activities you offer will set the tone for your gathering.

> "IT'S ALL DOWN TO WHAT YOUR INTENTIONS ARE FOR THE EVENT, WHAT SPACE YOU HAVE IN YOUR VENUE, AND WHAT EQUIPMENT YOU HAVE AVAILABLE. FOR INSTANCE, FOR YOUR FIRST EVENT, YOU MAY NOT HAVE ANY KIT AND YOU MAY HAVE A SMALL VENUE, WHICH DICTATES A MIX OF TALKS AND/OR SHOW-AND-TELL. IN LARGER VENUES, YOU POTENTIALLY HAVE MORE ROOMS TO USE, SO YOU CAN EXPAND INTO WORKSHOPS (PROVIDED YOU CAN GATHER ENOUGH EQUIPMENT TOGETHER), MORE SHOW-AND-TELL, AND EVEN A MARKETPLACE IF YOU CAN PERSUADE VENDORS TO ATTEND."
>
> **MICHAEL HORNE**
>
> CAMBRIDGE RASPBERRY JAM

Drop-in

A drop-in is a popular informal activity. As the event organiser, you need to provide people with space to work on their own projects, or to look online for something to make or learn. This might seem like something your attendees

could do at home, but doing it at the Jam gives them the instant support of other community members.

Running a drop-in is easy. You will need to provide tables, chairs, and electricity, and ask your attendees to bring their own kit. If your venue can give you access to a PC suite, participants can bring their own Raspberry Pis and connect them to the venue's monitors, mice, and keyboards. You could provide printed worksheets for people who need guidance, or point them in the direction of some online resources. A small team of helpers offering advice and assistance will help your drop-in to run smoothly.

Beginner sessions

Use beginner sessions to introduce newcomers to digital making or coding. A beginner session could take the form of a Q&A for people new to a particular programming language or device. Alternatively, you could provide a team of volunteers to guide beginners through some basic tasks or projects. This could involve the use of a PC suite for coding activities, or support for people bringing their own kit that they haven't used before. Make sure you announce your beginner sessions in your publicity so that new attendees know that they will be welcome.

Show-and-tell

Run a show-and-tell as a way to inspire people to create their own projects. Invite people to bring along projects they are working on, or have completed, and to demonstrate them to other attendees. Demonstrating personal projects is a great way to show newcomers what hobbyists can achieve. Showcasing projects gives people the chance to ask questions about the builds, and for the owners to share their experiences.

It is best to organise your show-and-tell in advance. Compile a list of people and projects, and plan the space at your venue to accommodate them. You'll need to announce your show-and-tell in the publicity for your event, and make it easy for people to participate.

If you are prepared to let everyone exhibit their projects, offering an exhibitors' ticket type on Eventbrite will give you advance warning of how many projects you need to accommodate. If you have limited space, you could ask people to apply to take part, and choose the most appropriate exhibitors. Give yourself time to make contact with your exhibitors to make sure you know what space they need, and what other requirements they have. If you can, provide a small team of people to help your exhibitors to set up their projects.

Interactive show-and-tell projects are particularly popular with visitors. Robots that participants can control, or games that people can play, are very engaging, and demonstrate the possibilities of digital making.

> “ DEMONSTRATING PERSONAL PROJECTS IS A GREAT WAY TO SHOW NEWCOMERS WHAT HOBBYISTS CAN ACHIEVE

Workshops

Workshops provide structured practical experiences for your participants. They are more formal then drop-ins or show-and-tell sessions, and you will need to make sure that your workshop leaders are supplied with the equipment they need. You may to need to provide equipment for your workshop participants, or they may be able to provide their own. Make sure they know in advance what they need to bring.

Your first job as the organiser is to find leaders for your workshops. Ask your colleagues, people you know from other tech groups, or your regular attendees whether they would be interested in running a workshop, and make sure that the workshops for your event cover a range of projects. You could lead a workshop yourself, but make sure that you have a team of helpers to keep the event running smoothly while you are busy.

The workshop leaders will need to plan their content carefully, and decide how they intend to deliver it. You could structure workshops around worksheets, presentations, or both. Prepare your SD cards or hard drives in advance with all the software you need for the workshop, and check that the software runs correctly.

Schedule your workshops with clear start and finish times, and make sure you publicise the level of your workshop.

If a workshop is covering an entry-level subject, make sure that your participants are aware that they don't need to know anything about the subject to take part. If your workshop is covering a more advanced subject, make sure that your participants understand what prior knowledge they will need before they sign up.

Make sure you provide a team of helpers to set up and clear away each practical activity, and to help the activity leader to set up and run their workshop.

> "WHEN I RAN THE MANCHESTER JAM, PEOPLE WERE HAPPY TO WORK ON THEIR OWN PROJECTS. OCCASIONALLY WE RAN A TRACK OF TALKS, OR PUT ON A WORKSHOP, WHICH WAS A NICE CHANGE. NOW I RUN THE JAM AT PI TOWERS, THE ATTENDEES TEND TO BE MOSTLY BEGINNERS – USUALLY PARENTS WITH THEIR CHILDREN – SO WE MOSTLY RUN WORKSHOPS AND HAND OUT WORKSHEETS."
>
> BEN NUTTALL
> RASPBERRY JAM @ PI TOWERS

Talks

Another formal activity to consider is a series of talks. These can be lengthy conference-style presentations, or on-the-spot lightning talks. Scheduled talks give attendees the chance to share something with other enthusiasts: a project they've been working on, something they've learned, or an inspirational story.

If you are planning to run formal talks, you will need a space that is separate from the rest of your event so that the talk does not disrupt the other activities, and vice versa. You will need to provide a projector, a screen or a suitable wall, and a microphone and speakers. Ensure that each presenter is provided with the correct cables to connect their kit to the projector, and with anything else they need.

You will also need to invite people to speak, in advance of the event. You could issue a general call for speakers in your publicity for the event, or you could approach particular people and ask for their input. Make sure that anyone volunteering to speak knows how to contact you, and that you confirm their talk with them in advance to give them plenty of time to prepare. You will also need to find out what equipment and connectivity each speaker needs for their presentation, and for any demonstrations they are planning to run.

Assign one or two of your volunteers the role of managing the talks. They will need to ensure that the correct equipment is set up for each speaker in time for their talk to begin, that the capacity of the room is not exceeded, that the audience can arrive and leave the room again quickly, and that the talks start and end on time. If the speaker is planning to take questions from the audience, the manager should control this process.

Talks don't have to be very long: a series of short lightning talks can provide great opportunities for people to present their projects, or something they've learned and want to share. Talks of five, ten, or 15 minutes are a great way to give new speakers the chance to stand up and present on their chosen topic, without the need to prepare a long lecture. Encourage all attendees to consider giving a lightning talk, regardless of their age, experience, gender or background.

> IF YOU ARE RUNNING A SERIES OF TALKS, CONSIDER ORGANISING A PANEL SESSION

Panel session

If you are running a series of talks, consider organising a panel session. Give your attendees the chance to put their questions to a group of experienced individuals on a particular topic. Ask your speakers whether they would be willing to participate in a panel, as well as giving their own talk. Your panel could be a general Q&A session on digital making and coding, or you could focus on one device or programming language. Make sure you publicise the focus of the panel in advance.

You will need to provide microphones, speakers and other equipment to your panel members. You may also need a microphone for the audience. This could be a microphone on a stand that audience members queue up to use, or a radio mic held and managed by a volunteer.

Marketplace

Some organisers invite retailers to sell digital making and coding accessories and equipment at their events. This can be an effective way for retailers to reach customers in the digital making and coding community. It is also the perfect place for attendees to look at kit before they buy, and to put questions to the sellers in person.

Make sure that retailers know you are looking for sellers for your event. Approach relevant retailers directly, particularly local businesses. Look for contact information on retailers' websites, or reach out on Twitter.

Make sure you check with your venue before you organise a marketplace – some venues do not allow commercial transactions during events. Talk to vendors in advance and find out what equipment, power and facilities they will need. Provide a dedicated space for your marketplace, and make sure you provide a small team of volunteers to help your vendors to set up their stalls, and to fix any problems during the event.

> "WHEN A COMPETITION LIKE ASTRO PI OR A PROGRAMME LIKE PIONEERS COMES OUT, WE'LL DO TALKS, AND INVITE PEOPLE ALONG TO TAKE PART IN WORKSHOPS AT THE JAM. IT'S GOOD BECAUSE IT GIVES AN INCENTIVE FOR PEOPLE TO COME ALONG AND TAKE PART, AND GIVES THEM MOTIVATION TO WORK. WE CELEBRATE THE ENTRANTS, AND THOSE WHO'VE WON, AND INVITE THEM TO DO TALKS. WE ASK THEM TO MENTOR AT FUTURE RASPBERRY JAMS, OR RUN WORKSHOPS."
>
> **ANDY MELDER**
>
> SOUTHEND RASPBERRY JAM

Other formats

Your event doesn't have to be open to the general public.

- A TeachMeet is an event where teachers gather to give short presentations to their peers and share ideas for teaching. Why not run a digital making and coding TeachMeet, with practical workshops and short presentations? More information on organising a TeachMeet can be found on the Teacher Toolkit website (**helloworld.cc/2rT9rnC**).

- Your event could promote participation in national programmes and competitions. The Raspberry Pi Foundation runs Pioneers, a competition aimed at teams of 12- to 15-year-olds (**rpf.io/pioneers**), and Astro Pi, a competition to write code to be run on the International Space Station, aimed at under-16s and their teachers (**astro-pi.org**).

- Register your event as a Code Club, and try using the Code Club resources for beginners. First, register yourself as a host on the Code Club website (**codeclubworld.org**). Find a volunteer willing to support children aged 9-13 to work on projects in Scratch, HTML, and Python. Register them as an official volunteer on the website, and your Code Club will be activated.

- If after running a one-off event you are inspired to start a regular club that caters for young people aged 7-17, why not register your event as a CoderDojo? Register at **coderdojo.com**, wait for confirmation, and find a team of volunteers to help you. There are plenty of free resources on the CoderDojo website to support your event, including two e-learning modules for adult helpers.

This is not an exhaustive list of the activities you could run at your event! For more ideas, visit other local events and talk to other event organisers. (HW)

> "SOMETHING LIKE ASTRO PI IS A FANTASTIC THING TO GET INVOLVED IN IF YOU WANT TO REALLY GIVE CHILDREN SOME FOCUS AND DIRECTION. THE IDEA OF HAVING YOUR CODE RUN ON THE INTERNATIONAL SPACE STATION IS A REAL MOTIVATOR FOR YOUNG KIDS."
>
> **DAVE HONESS**
>
> RASPBERRY PI FOUNDATION

RASPBERRY
JAM
GUIDEBOOK
STARTING AND RUNNING A RASPBERRY JAM
ADVICE GATHERED FROM THE RASPBERRY PI COMMUNITY

DOWNLOAD

Download your copy of the Raspberry Jam Guidebook from: **helloworld.cc/2p7ZX98**

YOUR LETTERS

NEW RESOURCE

Dear Hello World,
I am writing to let you know about a new resource developed by CAS, exploring computational thinking and creativity in programming.

The recent CAS Tenderfoot program cites finding a solution to a simple version of The Knight's Tour as a fruitful example of computational thinking for secondary teachers and pupils. We have run with this idea, using guided discovery as a basis for discussion, allowing teachers and pupils to attempt to find a variety of human solutions to The Knight's Tour and related problems. The resource shows how a simple model of computational thinking (including creativity, heuristics, symmetry, induction, and deduction) underpins this approach, while using representations of the problem in online interactive boards and diagrams, and incorporating some ideas from graph theory.

Our theory is that finding human solutions to a problem can provide a basis for devising and implementing algorithms for program solutions. This in turn can lead to strategies for finding program solutions to more complex problems. In follow-up discussions, we look at how the human solutions we have developed may be transformed into program solutions, and to more general solutions.

The resource, entitled From *Human Solutions to Program Solutions: Computational Thinking, Creativity and Pedagogy in Programming*, can be found at **helloworld.cc/2ugOMyb**. We hope it will provide a useful reference for your readers.
Dave White, CAS Master Teacher

Our letters page is a place for you to join our conversation. If you've got a comment, a question or an announcement to share, contact us on Twitter via @helloWorld_Edu or using the #HelloWorld hashtag. Alternatively, email us with 'Teacher Letter' in the subject line (contact@helloworld.cc).

Theoretical inspiration

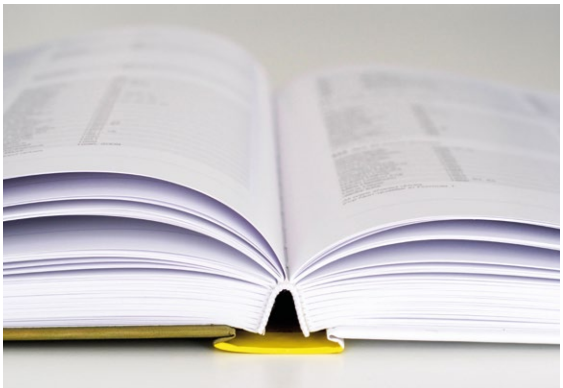
Dear Hello World,
I have some students who are very good at programming. They can crack any logic problem at their level, but they aren't very good at CS theory. They can code better than their classmates, but they're indifferent about the theory, displaying an 'I don't care' attitude.

They do need to know about theory. There are various theory papers in our syllabus, which cover NFA, DFA, Turing machines, artificial intelligence, ES, fuzzy logic, etc.

How should I manage these students? They are in the 17-19 age group.
Sagar V, Tanur, India

This is a fairly common problem, especially when a class contains a few self-taught coders who have been programming for a relatively long time. The first thing to do is ensure they have a thorough understanding of what their assessments will entail. Reading through a course specification is a little dry, but providing them with past exam papers and assessment tasks should make sure that they understand that there's more to Computer Science than just programming. Hopefully the realisation that they will need some theoretical knowledge in order to pass will provide some of them with the motivation they need.

The second thing you can try is to present the theoretical material you need them to learn as a programming challenge. After all, if a student can create their own model of a deterministic Turing machine or a Finite State Machine in code, they will first have to learn the concept to a sufficient depth before they can model it. (HW)

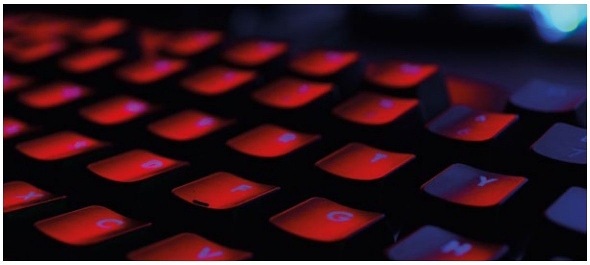


A level projects

Dear Hello World,
I'm new to teaching A level Computer Science, and I have started talking with my students about their final projects. Can students use Unity with C# to make a computer game for their A2 projects?
Lyndsey Cashen, Isle of Man

The key here is to communicate with your awarding body. Most exam boards will have a local area representative who you can contact, and although they might not have the answers you are looking for, they will be able to pass on your query to the relevant subject lead.

A cursory glance at both the AQA and OCR A level specifications would indicate that both boards are happy to have solutions written in C#, and there is no reason that a programming tool such as the Unity Game Engine would not be permitted.



Just remember that the awarding bodies prefer that the project provides a solution to a problem. 'A game to teach primary students basic maths skills' or 'A game to help develop spatial awareness' might be more appropriate than 'A game to save the Earth from invading alien hordes'.

A pitfall to watch out for in game development for a final project is students getting sidetracked by the creation of game assets. They are being assessed on their program and its accompanying documentation, rather than on the quality of the game. Remind them that the project is probably only worth about 20 percent of their final grade, so spending fifteen hours creating a photorealistic 3D model of car tyre is probably not a good use of time, when a primitive cylinder would suffice. (HW)

New grade boundaries

Dear Hello World,
What can we expect from the new grade boundaries for the exam element of the CS GCSE? The legacy boundary for an A* has been around 66 percent, which converted to the new grades would be very low, especially with reference to other subjects. Is there any guidance to allow us to predict grades for our Year 10 students?
Corinne Sherman, Bristol

This is not a question that anyone, possibly even within the exam boards, can answer with any certainty. There are a couple of things to take into account, however.

Firstly, the awarding bodies will be under scrutiny to ensure that there is no major shift in results between 2017 and 2018, when the new grading comes into effect. If your current cohort of Year 11s achieves 75 percent A*-C passes, and they are of comparable ability to your Year 10s, then you should be looking at about the same number of 6-9 passes. Unfortunately, you will have to perform your own calculations with regard to how this converts into possible grade boundaries.

Secondly, you might want to book some time with your school's Head of English and Head of Maths. They'll be receiving the new style GCSE results in August, and you might be able to learn from them how they tackled predicting grades, and what they are predicting for your current cohort of students. (HW)

SHARE YOUR EXPERIENCES

Dear Hello World,
I need help!

Last year, I started research looking for a magic bullet for teaching abstraction in primary schools. However, I discovered that research requires you to look at one tiny aspect of a big subject in order to gradually build a solid body of evidence. So now I am focusing on one aspect of abstraction: how we use design when teaching programming, and how we might reuse our expertise in teaching writing when we teach programming.

My work builds on research from across the world, but all of it with older pupils. From Israel to the US via Scotland, the Netherlands and Germany, I am knitting threads of theory and practice together – but for teaching younger learners.

If you teach programming to primary children, in school, out of school, formally or informally and have 15 minutes to spare, I would be indebted if you could complete our survey. You can find it at **helloworld.cc/2tPZjiR**.

Please do share the survey with your colleagues too!

Many thanks,
Jane Waite, Queen Mary University of London

"HELLO, WORLD!"

Everything you need to know about the new computing and digital making magazine for educators...

Q WHAT IS HELLO WORLD?

A Hello World magazine is a new magazine for computing and digital making educators. Written by educators, for educators, the magazine is designed as a platform to help you find inspiration, share experiences, and learn from each other.

Q WHO MAKES HELLO WORLD?

A The magazine is a joint collaboration between its publisher, Raspberry Pi, and Computing At School (part of BCS, The Chartered Institute for IT). Hello World is sponsored by BT.

WELCOME TO THE NEW MAGAZINE FOR COMPUTING & DIGITAL MAKING EDUCATORS
(Hello World)
Issue 3
Autumn Term 2017
helloworld.cc
10 TIPS TO SUPPORT INDEPENDENT LEARNERS
Keep your independent students engaged and motivated with our expert advice
BACK TO BASIC
Using BASIC and GOTO to teach the foundations of programming
CODE CLUB FOR TEENS
New resources for 9- to 13-year-olds
BUILD PONG IN SCRATCH
A lesson plan for 11-year-olds
APPROACHING ASSESSMENT
Our international panel of experts discusses innovative approaches and technologies for assessing programming skills
OPEN-SOURCE RESOURCES
Computing has benefited from open-source. Can education too?
EMBRACING PLAY
How play could make you a better educator
PLUS
LEARNING THROUGH HACKING • WILL A COMPUTER TAKE YOUR JOB? • CS UNPLUGGED • MATHEMATICAL MUSINGS • YOUTUBE CLASSROOM • RASPBERRY PI & CODERDOJO MERGE • WHAT DOES A PROGRAMMER DO? GETTING GIRLS CODING • APPS FOR GOOD AWARDS 2017 • ASSIGNING VARIABLES • SCHOOL LEADERS MATTER

Q WHY DID WE MAKE IT?

A There's growing momentum behind the idea of putting computing and digital making at the heart of modern education, and we feel there's a need to do more to connect with and support educators inside and outside the classroom.

Q WHEN IS IT AVAILABLE?

A Your new 100-page magazine will be available three times per year in time for each new term in January, April, and September. Would you like it to be available more frequently? Let us know!

IT'S FREE!

Hello World is free now and forever as a Creative Commons PDF download. You can download every issue from **helloworld.cc**. Visit the site to see if you're entitled to a free print edition, too.

WANT TO GET INVOLVED?

There are numerous ways for you to get involved with the magazine.
Here are just a handful of ideas to get you started:

- **Give us feedback**
  Help us make your magazine better – your feedback is greatly appreciated.
- **Ask us a question**
  Do you have a question for a FAQ section or a bugbear you'd like to share? We'll feature your thoughts and ideas.
- **Tell us your story**
  Have you had a recent success (or failure) you think the wider community would benefit from hearing? We'd like to share it.
- **Write for the magazine**
  Do you have an interesting article idea? We'd love to hear from you.

GET IN TOUCH

Want to talk? You can reach us at:
contact@helloworld.cc

FIND US ONLINE

www.helloworld.cc

@HelloWorld_Edu

fb.com/HelloWorldEduMag

SUBSCRIBE IN PRINT TODAY!
PAGES 32-33

MILES BERRY PRINCIPAL LECTURER

ONE LANGUAGE AT A TIME

Do students need breadth of experience or depth of skills?

Most of us working in computer science education seem to agree that coding is not the end in itself, but the means through which our students get useful things done, express themselves creatively, and develop their understanding of the fundamental principles of computer science. While some of our students will go on to be the software engineers and computer scientists of the future, almost all will find themselves using computers. An understanding of how these machines work, and how to think about problems so that the computers can help solve them, is what's important for these students.

> " in short, they need to become fluent (or at least conversant) in one, or two, or three programming languages

One language or many?

Where some of us disagree is on the best means to this end. On the one hand, there are those who argue that these principles, and indeed the constructs of programming (such as sequence, selection, and repetition), are best learnt generically and then applied in as many programming languages as time, teacher expertise, and student interest permits. Others take the view that actual, practical experience of coding in one language is the best way to learn to program – and through this to acquire, either through explicit instruction or guided discovery, an understanding of the underlying principles.

The arguments for the 'many languages' approach seem similar to those used in the ICT teaching past: that to avoid death by PowerPoint, we should allow students to choose Prezi, Google Slides etc. instead, as the software skills themselves are what matter. The 'many languages' advocates argue that there's more to programming than Scratch, or even than Python; and to avoid children becoming bored with Scratch, they should also learn Logo, Kodu, Hopscotch, Tynker, and so on.

Fluency counts

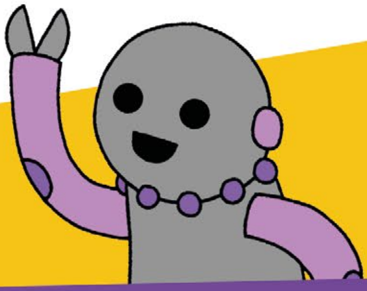
Somehow though, programming seems different from creating a presentation. If we're serious about students becoming adept at solving problems and expressing their creativity through programming, then they need to develop some sort of mastery of the medium: in short, they need to become fluent (or at least conversant) in one, or two, or three programming languages.

By analogy with learning human languages – while we might be concerned that students get to grips with comparative linguistics and deep structure, we're much more concerned that students learn a language, are able to hold productive conversations in it and, perhaps later, become able to write stories, essays or poems in the language.

There seems to be relatively little research comparing the one-language and many-language approaches to teaching programming: mainly, I suspect, because this question rarely arises in undergraduate CS education. It is simply assumed that introductory programming courses will teach a particular language, although this still allows plenty of argument over the choice of paradigm and language.

While the brightest and best will, I think, be able to transfer the concepts from one language to another, I fear we do a disservice to those who struggle to express their ideas in a first programming language by rapidly introducing them to many more. Changes in syntax, grammar and vocabulary do little more than add to their already high cognitive load. Let's learn the lesson from the mathematics education community, that mastery comes through depth of experience, not superficial acceleration. (HW)

Miles Berry is principal lecturer in computing education at the University of Roehampton. He serves on the boards of Computing At School, the BCS Academy and the Computer Science Teachers Association, and is a Member of the Raspberry Pi Foundation.

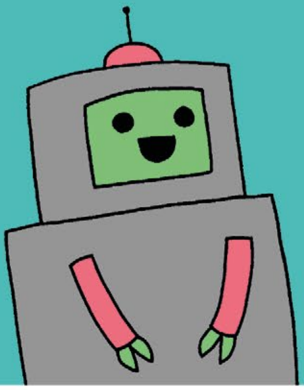


START A CODE CLUB IN YOUR SCHOOL!

Code Club is a network of volunteers and educators who run free coding clubs for young people aged 9-13.

Our aim is to inspire the next generation to get excited about computer science and digital making.

> "We use Code Club's fun educational resources to run a weekly after-school club for Year 7 and Year 8 pupils. The students benefit considerably from the extra challenge!"
>
> **Karen Dadd, Computing Teacher**



- ☑ **Code Club is free**
- ☑ **Code Club provides step-by-step guides for Scratch, Python, HTML, and Sonic Pi**
- ☑ **Code Club helps children develop skills including logical thinking, creativity, and resilience**

We have over 6000 clubs across the UK teaching more than 80,000 young people to code—come and join us!"

Find out more at www.codeclub.org.uk

Code Club is part of the Raspberry Pi Foundation. Registered Charity Number 1129409

(Hello World)

helloworld.cc