Bober 2024/25

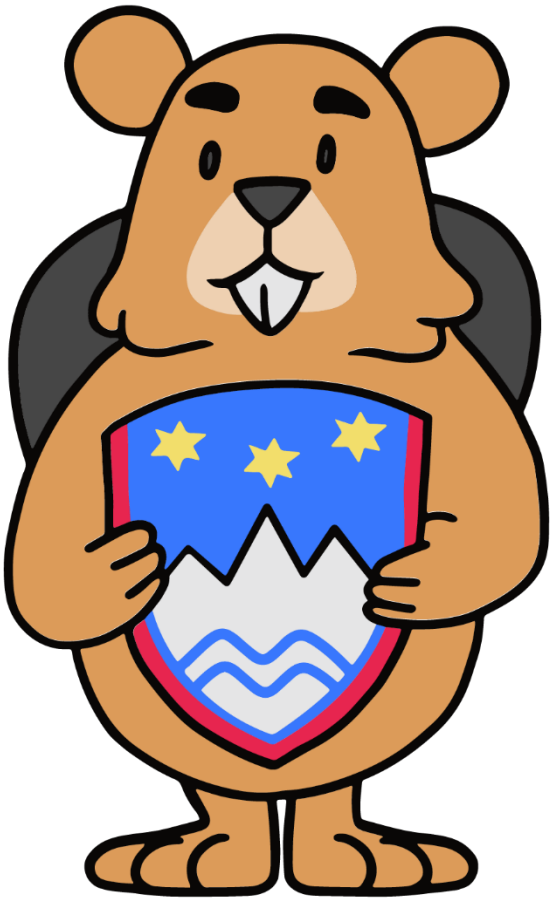
Naloge za tekmovanje smo uredili, prevedli, priredili in oblikovali:

Alenka Kavčič (UL FRI)

Nežka Rugelj (OŠ Trzin)

Rok Požar (UP)

Špela Cerar (UL PEF)



Bebras

ACM Slovenija

Naloge in rešitve

Programski svet tekmovanja ACM Bober za šolsko leto 2024/2025:

Alenka Kavčič (UL FRI)
Nežka Rugelj (OŠ Trzin)
Rok Požar (UP)
Špela Cerar (UL PEF)

Razvoj tekmovalnega sistema:

Gašper Fele Žorž (UL FRI)
Gregor Jerše (UL FRI)

Slika na naslovnici:

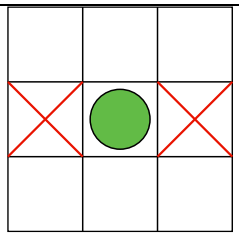
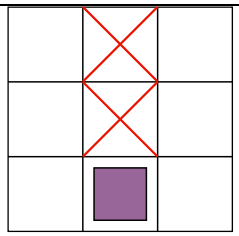
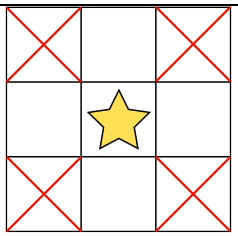
Miha Goršin (GZS)

Kazalo nalog

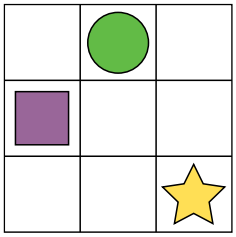
Igra simbolov

6. do 7. razred

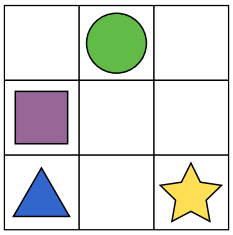
Manca igra igro simbolov, pri kateri postavlja krog, kvadrat, trikotnik in zvezdo na mrežo velikosti 3x3. Pri tem uporablja naslednja pravila:

Noben simbol ne sme ležati v isti vrstici kot krog.	Noben simbol ne sme ležati v istem stolpcu kot kvadrat.	Noben simbol ne sme ležati na isti diagonali kot zvezda.
		

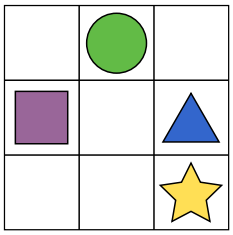
Manca je na mrežo že postavila krog, kvadrat in zvezdo.



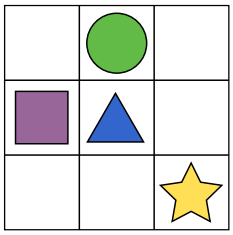
Kako bi lahko izgledala Mančina mreža, ko je postavila še trikotnik?



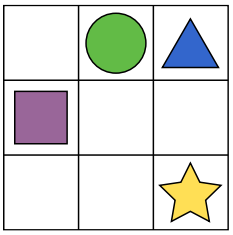
A



B



C

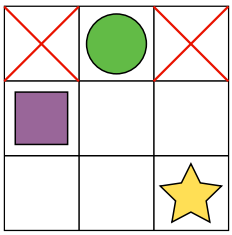


D

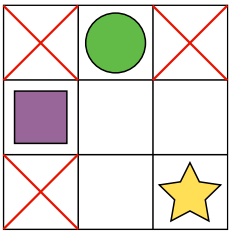
Rešitev

Pravilen odgovor je B.

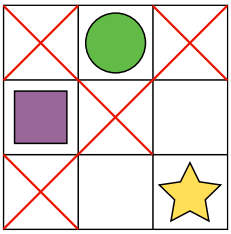
Sledimo pravilom. Ker je krog v zgornji vrstici, tam trikotnika ne more biti.



Ker je kvadrat v levem stolpcu, v njem ne more biti trikotnika.



Po pravilu za zvezdo, ki pravi, da noben lik ne more biti na isti diagonali, trikotnik ne more biti niti na sredini.



Ostaneta nam le dve možnosti, kje lahko Manca postavi trikotnik: v desni stolpec druge vrstice in v sredinski stolpec tretje vrstice. Med odgovori, ki so na voljo, je edini pravilni odgovor s trikotnikom v eni od teh dveh polj odgovor B.

Računalniško ozadje

Pri tem izzivu moramo uporabiti posebna pravila in omejitve pri umeščanju simbolov v mrežo, kar zahteva preverjanje pogojev, kot sta vrsta simbola v celici in njegov položaj v mreži. Ta pravila delujejo kot logični pogoji, ki določajo, ali je dejanje, kot je postavitev novega simbola, veljavno ali ne.

Kocke

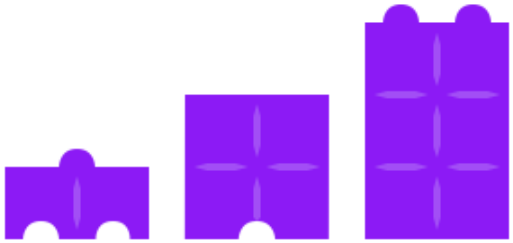
6. in 7. razred

Podjetje Bebro izdeluje kocke s štirimi značilnostmi:

- širina (možne vrednosti: ozka, povprečna ali široka)
- višina (možne vrednosti: nizka, povprečna ali visoka)
- število polkrogov zgoraj (možne vrednosti: nič, ena ali dve)
- število izrezanih polkrogov spodaj (možne vrednosti: nič, ena ali dve)

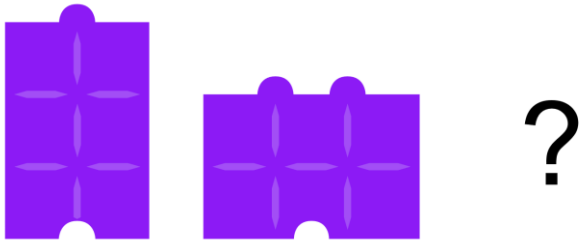
Pri pakiranju delavci združijo kocke v skupine po 3 po pravilu: po vsaki značilnosti se vse kocke v skupini bodisi ujemajo po vrednosti, bodisi povsem razlikujejo (ima vsaka kocka drugačno vrednost).

Primer:



Zgornje kocke so lahko v enem paketu, saj se popolnoma ujemajo v širini, povsem razlikujejo pa po višini, številu polkrogov zgoraj in številu izrezanih polkrogov spodaj.

Delavec David je končal izmeno in pustil nedokončan paket. Katarina, ki ga je zamenjala, mora dokončati ta paket. Katero kocko naj doda?

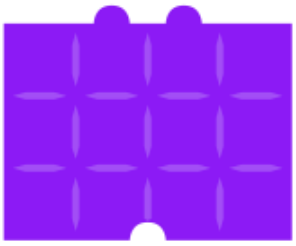


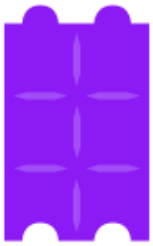
A) 

B) 

C) 

D) 

E) 

F) 

Rešitev

Ker imata prvi dve kocki enako število izrezanih polkrogov spodaj, različni pa sta po višini, širini in številu polkrogov zgoraj, mora izbrati tretjo kocko, ki se s prvima dvema ujema v številu izrezanih polkrogov spodaj, v vseh ostalih lastnostih pa se mora razlikovati od obeh. Torej je naša kocka široka, nizka, nima polkrogov zgoraj in ima en izrezan polkrog spodaj.

To je kocka B: 

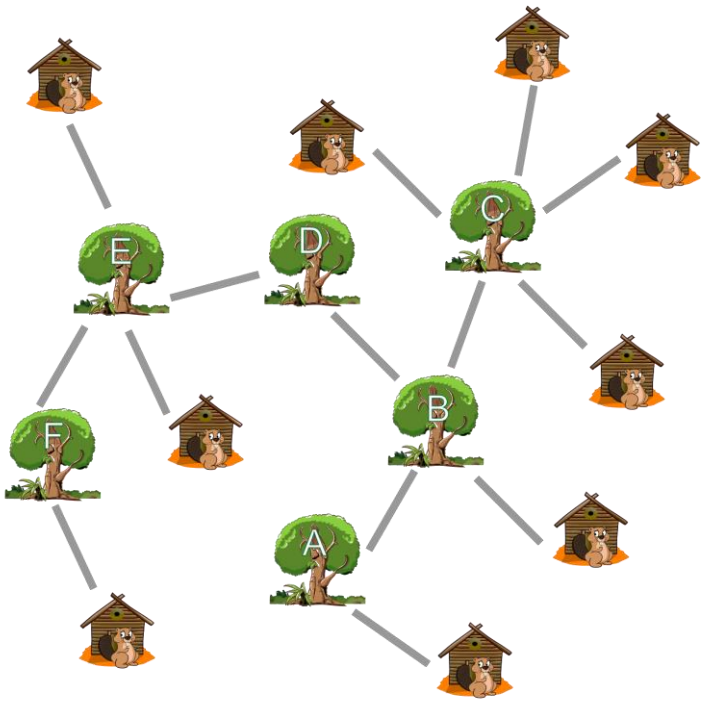
Računalniško ozadje

V nalogi smo morali poiskati vzorce v podatkih. Vzorci so lahko različni, odvisno od obravnavanega problema; v naši nalogi so vzorce opredeljevale oblike kock. Iskanje urejenih vzorcev v velikem sistemu ali strukturi se pogosto uporablja v matematiki (kombinatorika, teorija grafov), računalništvu in fiziki. Omogoča vpogled v obstoj urejenosti in strukture v navidezno kaotičnih sistemih. Vpliva tudi na področja, kot so analiza omrežij, problemi optimizacije in kriptografija.

Smučarski Raj

6. do 7. razred

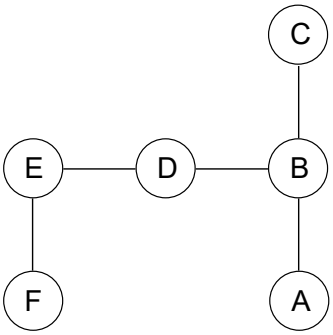
Bobri so zgradili novo smučarsko središče Smučarski Raj z devetimi kočami. Vse koče in drevesa so povezali z enako dolgimi cestami (sive črte na sliki). Manjka jim le še jedilnica, ki jo bodo zgradili pri enem od dreves, ki so označena s črkami od A do F. Pod katerim drevesom naj postavijo jedilnico, da bodo imeli tudi najbolj oddaljeni gostje čim krajšo pot do jedilnice?



Rešitev

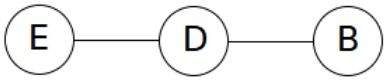
Jedilnico morajo postaviti pod drevesom D.

Zaradi lažjega razumevanja predpostavimo, da vsaka siva črta predstavlja razdaljo 1 enote. Ne glede na lokacijo srečanja mora vsak bober najprej prepotovati 1 enoto, da doseže bližnje veliko drevo. To nam omogoča, da ne upoštevamo hišic in se osredotočimo le na velika drevesa. Ustvarimo lahko poenostavljen graf, kot je prikazan spodaj, kjer je vsako veliko drevo prikazano kot označen krog.



Kot vidimo, sta A in B povezana, pri čemer ima B dodatne povezave do drugih velikih dreves, medtem ko ima A povezavo le do B. Za bobre, ki se nahajajo na A in B, ni razlike med izbiro A ali B za lokacijo jedilnice, saj morajo v obeh primerih nekateri bobri prepotovati 1 enoto. Za bobre, ki se nahajajo pri drugih velikih drevesih, pa bi bilo bolj ugodno, če bi bila jedilnica na B, saj bi morali prepotovati 1 enoto manj, kot če bi bila jedilnica na A. Iz tega razloga je za vse bobre ugodneje, če se bobri z A premaknejo na B.

Podobno se zgodi z vozliščema C in B ter vozliščema E in F. Bobri s C bi se morali premakniti na B, bobri s F pa na E. Jedilnica torej ne sme biti na A, C ali F, kandidati za jedilnico ostanejo le B, D in E. Tako dobimo še preprostejši prikaz, ki je prikazan spodaj.



Iz zgornjega grafa lahko razberemo, da je smiselno jedilnico postaviti na D, saj bi morali do tja vsi bobri prepotovati le še eno dodatno enoto.

V takšni ureditvi je največja razdalja, ki jo prepotuje kateri koli bober od svoje koče do jedilnice, 3 enote. To je najmanjša možna razdalja, ki izpolnjuje dane pogoje.

Računalniško ozadje

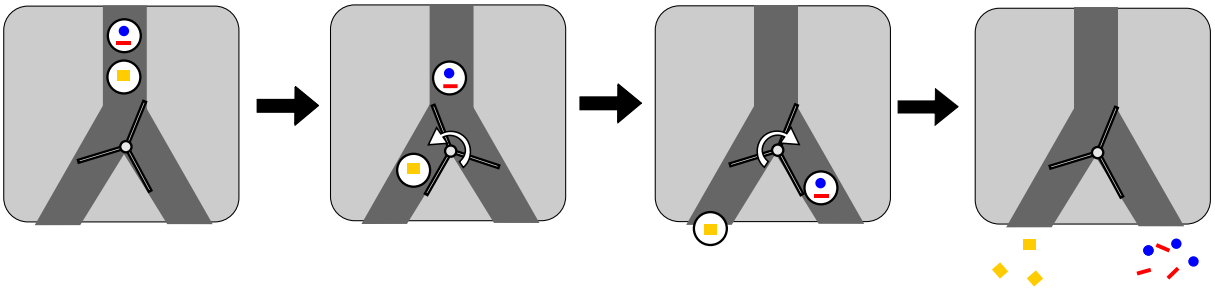
Reševanje naloge smo poenostavili tako, da smo problem opisali z *grafom*, pri čemer vsak krog predstavlja *vozlišče*, vsaka črta pa *povezavo*. Graf je podatkovna struktura, s katero lahko predstavimo številne situacije v resničnem svetu (pravzaprav smo v nalogi uporabili posebno vrsto grafa, ki se imenuje *drevo*). Center grafa je vozlišče, pri katerem je razdalja do najbolj oddaljenega vozlišča najmanjša.

V nalogi smo pri iskanju centra grafa uporabili tehniko rezanja (angl. *pruning*). Rezanje se pogosto uporablja pri iskalnih algoritmih in strojnem učenju za zmanjšanje velikosti odločitvenih dreves z odstranjevanjem delov drevesa, ki niso kritični ali pa so odvečni za razvrščanje primerov.

Ustvarjanje vzorca

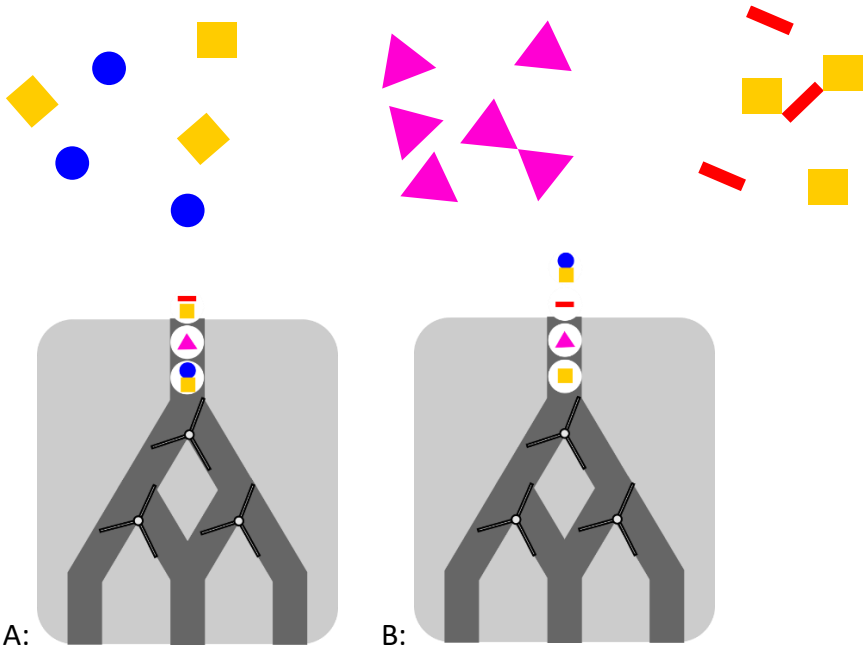
6. do 7. razred

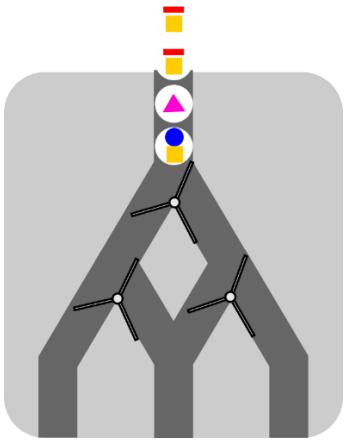
V kočah novega zimskega središča Smučarski Raj bodo za izdelavo vzorcev na tleh uporabili stroj, ki je prikazan na spodnji sliki. V vsaki krogli so delčki, ki ustvarijo specifičen vzorec, krogle pa sledijo smeri, ki jo določajo vrata. Ko krogla preide skozi vrata, se vrata samodejno obrnejo in pošljejo naslednjo kroglo v drugo smer.

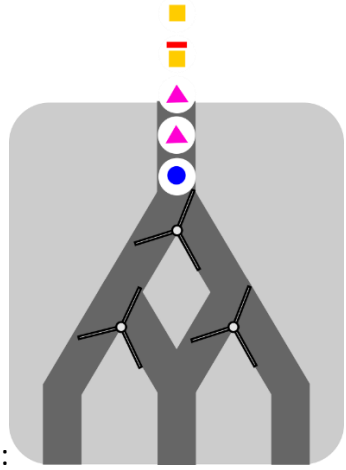


Primer najprej prikazuje vrata, ki so odprta na levi, zato gre prva krogla levo. Pri tem se vrata obrnejo in naslednja krogla gre v desno. Druga krogla ponovno obrne vrata. Vsaka krogla ima oznako, katere oblike bo vseboval vzorec, ki ga bo ustvarila. Če različne krogle zapustijo stroj na istem mestu, se bodo tam oblike porazdelile po tleh. Če dve enaki krogli pristaneta na istem mestu, bo rezultat enak, kot če bi tam pristala le ena.

Katere krogle bodo na tleh ustvarile naslednji vzorec?

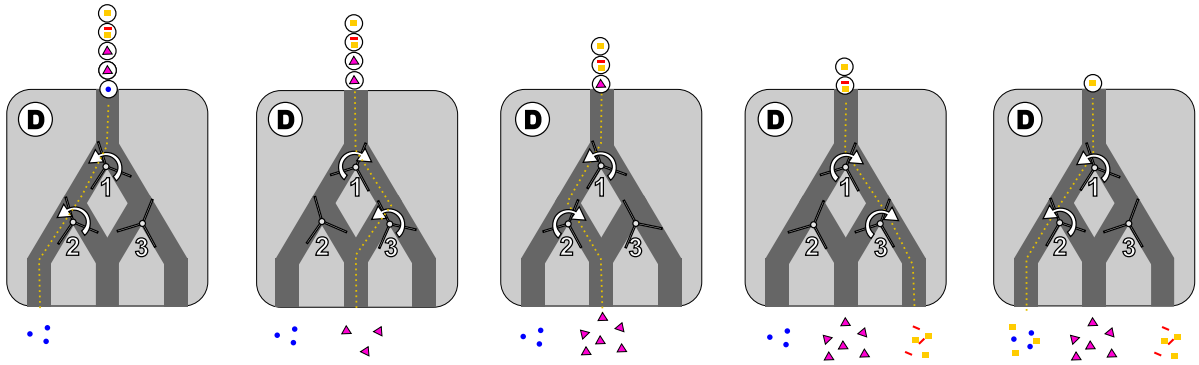


C: 

D: 

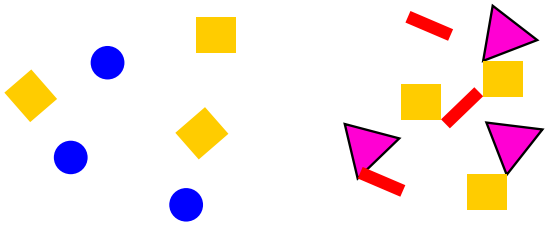
Rešitev

Pravilen odgovor je D:



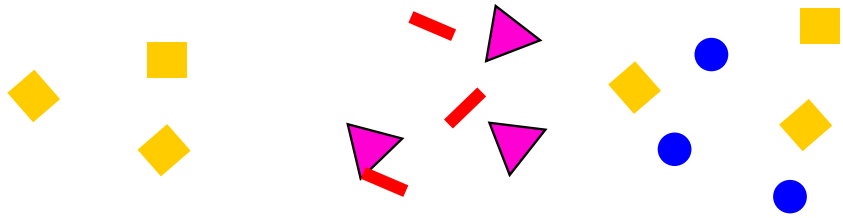
Najprej bo padla krogla z modrim vzorcem, ki bo šla dvakrat na levo stran in naredila vzorec na levi strani. Pri tem bo obrnila vrata 1 in 2. Zato bo šla naslednja krogla (roza) najprej v desno in nato v levo, zato bo naredila vzorec na sredini. Po prvi roza krogli so vrata 1 zopet odprta na levo, vrata 2 in 3 pa na desno. Tako gre druga roza krogla najprej levo, nato pa v desno in tudi ta naredi vzorec na sredini (in obrne vrata 1 in 2). Krogla z rumeno-rdečim vzorcem gre dvakrat v desno, zato naredi vzorec na desni strani (in obrne vrata 1 in 3). Zadnja krogla pa ima oboja vrata odprta na levo, zato doda rumen vzorec na modrega na levi strani.

Vzorec, ki bi ga naredile krogle v primeru A, je

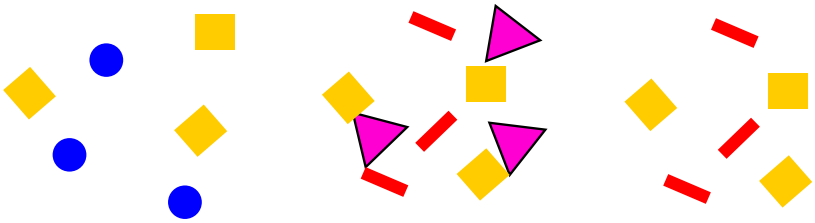


.

Vzorec, ki bi ga naredile krogle v primeru B, je:



Krogle v primeru C pa bi naredile vzorec:



Računalniško ozadje

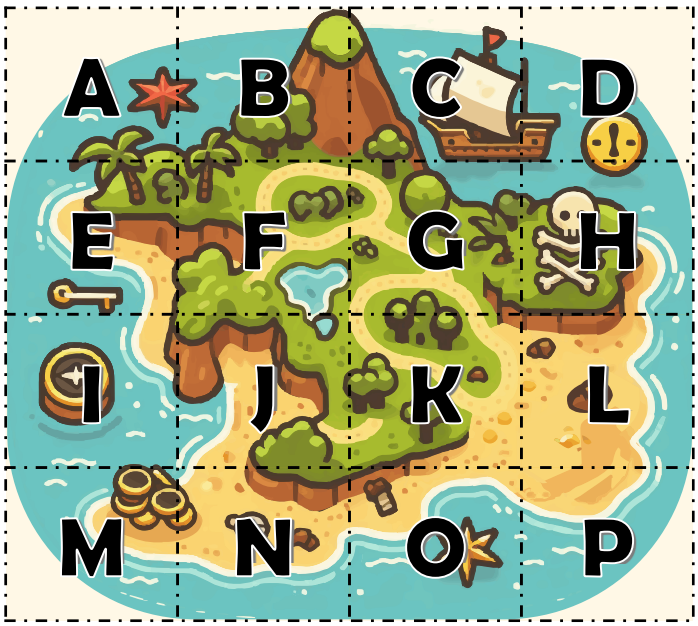
Računalniki uporabljajo zelo majhne komponente, ki delujejo podobno kot ta vrata v stroju. To so tranzistorji in delujejo kot majhna električna stikala. Tranzistorji so nenadomestljivi elektronski elementi v skoraj vseh sodobnih elektronskih napravah, v računalniku pa jih je na milijarde. Tranzistorje v računalniku lahko nastavimo na vključeno ali izključeno, tako kot so vrata v nalogi nastavljena na levo ali desno. Na začetku so vsa vrata v nalogi nastavljena na levo, a lahko vidimo, da posamezna vrata vplivajo na vrata, ki jim sledijo. Predstavljajte si, da bi lahko vsa vrata upravljali neodvisno, ali ne bi bila naloga veliko lažja?

Iskanje zaklada

6. do 9. razred

Kapitan Boris na otoku išče skriti zaklad.

Kapitan ima zemljevid otoka. Zemljevid je razdeljen na 16 polj, vsako polje pa je označeno s črko od A do P. Kapitan Boris lahko v posebno napravo vnese poljubno število črk, ki označujejo polja na zemljevidu in naprava mu pove, če je zaklad na enem od teh polj ali ne. Na primer, če naprava pokaže »JA«, ko kapitan Boris vnese A+C+D, potem se zaklad nahaja ali na polju A ali na polju C ali na polju D.



Najmanj kolikokrat mora kapitan Boris uporabiti napravo, da bo zagotovo našel zaklad?

Rešitev

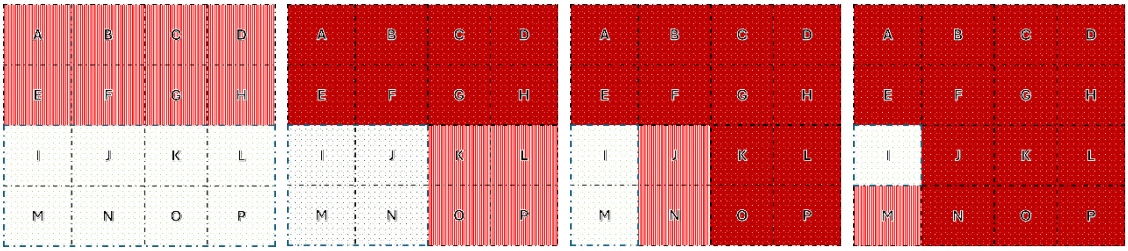
Kapitan Boris mora napravo uporabiti vsaj štirikrat.

Prvič mora kapitan vnesti črke polovice polj na zemljevida, potem pa ima dva možnosti:

a) Če je zaklad na tem delu zemljevida, potem mora kapitan ponovno razdeliti ta del zemljevida na dva dela in vnesti črke s polj, ki so na eni od polovic tega dela zemljevida.
b) Če zaklada ni na tem delu zemljevida, potem mora kapitan razdeliti na dva dela drugo polovico zemljevida in vnesti v napravo črke s polj, ki so na eni od polovic tega dela zemljevida.

Kapitan nadaljuje s tako delitvijo delov zemljevida, dokler ne najde polja, na katerem se nahaja zaklad.

Na primer, če najprej vnese polja A+B+C+D+E+F+G+H in naprava vrne »NE«, lahko v naslednjem koraku vnese v napravo polja K+L+O+P (katera koli kombinacija 4 polj s sprva neoznačenega dela zemljevida bi bila ustrezna). Če naprava ponovno vrne »NE«, potem razdeli preostala do sedaj še nikoli označena polja na dva dela (na primer J+N). Če naprava vrne »NE«, se zaklad nahaja bodisi na polju I bodisi na polju M, zato preveri eno od teh, na primer M. Če naprava vrne »NE«, se zaklad zagotovo nahaja na polju I.



Računalniško ozadje

Da bi našli zaklad na kar najučinkovitejši način (s čim manj uporabe naprave), moramo območja zemljevida vedno razdeliti na pol. V računalništvu uporabljamo podobno metodo za iskanje podatkov v urejenem seznamu, kjer prostor za iskanje večkrat razdelimo na polovico. Na ta način vsakič izločimo polovico podatkov, da iskane podatke najdemo v čim manjšem številu korakov. Opisana metoda, ki se imenuje *binarno iskanje*, omogoča hitro in natančno iskanje v velikih količinah podatkov.

Binarno iskanje ima številne uporabe v računalništvu, na primer:

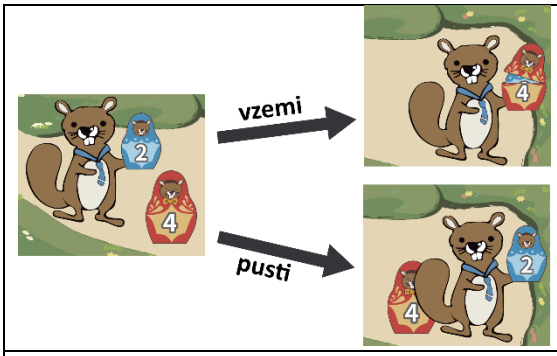
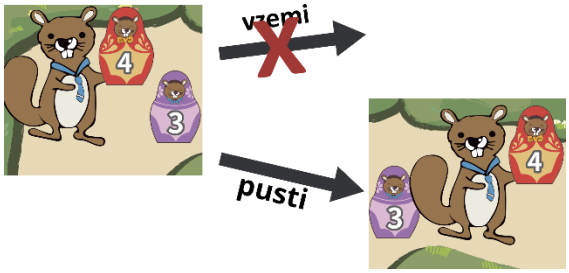
- iskanje besede v slovarju ali telefonske številke v imeniku,
- iskanje lokacije datoteke ali zapisa v podatkovni bazi,
- iskanje najboljšega premika v igri, kot sta šah ali križci in krožci,
- dešifriranje sporočila ali gesla z uporabo binarnega iskanja,
- analiziranje podatkov, na primer o rasti prebivalstva ali intenzivnosti potresov.

Labirint

6. do 9. razred

Bobrica Dana stoji na vhodu v labirint. V roki ima babuško velikosti 1. Želi se sprehoditi skozi labirint do izhoda in pri tem pobrati babuške, ki so raztresene po labirintu.

Dana se mora premikati v smeri puščic in slediti naslednjim pravilom vsakič, ko pride do babuške.

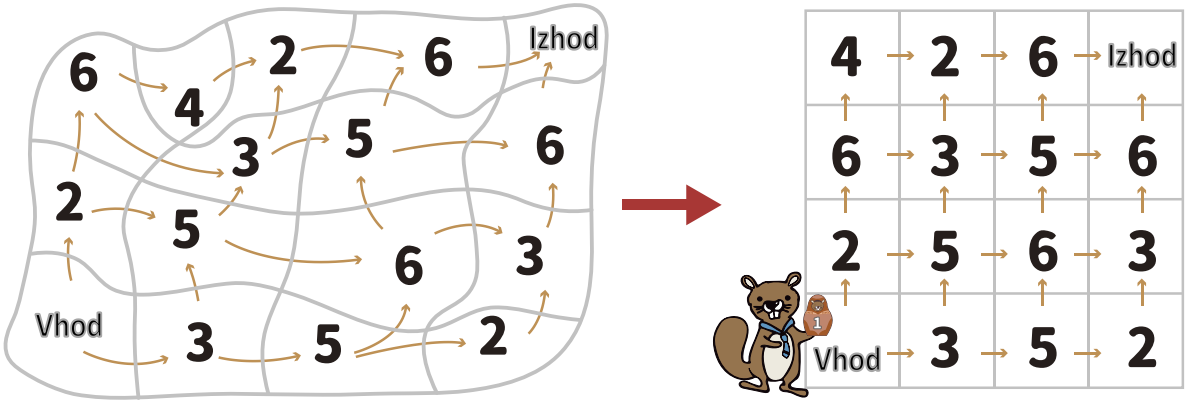
	Če je babuška večja od vseh babušk, ki jih trenutno nosi, jo lahko vzame ali pa pusti na svojem mestu
	Sicer mora babuško pustiti na svojem mestu

Največ koliko babušk, vključno z babuško velikosti 1, lahko Dana odnese s seboj iz labirinta?

Rešitev

Pravilen odgovor je 5.

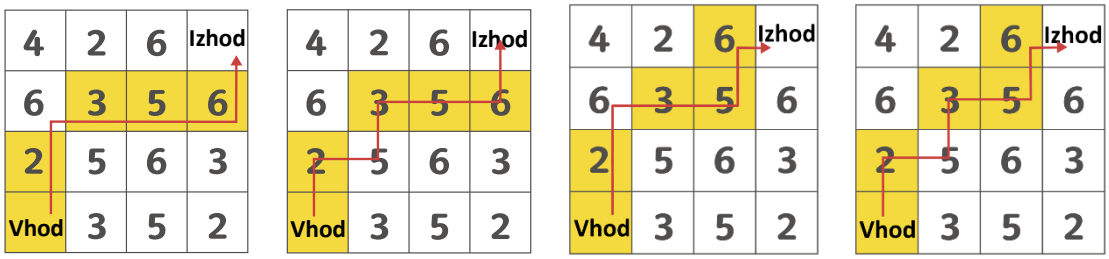
Zemljevid labirinta lahko poenostavljano narišemo kot mrežo, v kateri puščice kažejo navzgor ali desno.



Vse poti od vhoda do izhoda so dolge 6 korakov, zato lahko bobrica poleg svoje prve babuške pobere še največ 5 babušk in jih iz labirinta odnese 6. Po prvem pravilu morajo to biti pobrane babuške velikost 2, 3, 4, 5, 6, ki jih pobere v tem vrstnem redu.

Pot do babuške velikosti 4 je ena sama in na njej ni babuške velikosti 3. Če bi torej pobrala babuško velikosti 4, ne bi mogla pobrati babuške velikosti 3, niti babuške velikosti 5, saj od babuške velikosti 4 do izhoda le-te ni več. Torej bi poleg babuške velikosti 4 lahko pobrala še največ dve babuški, to sta babuški velikosti 2 in 6. Tako bi iz labirinta odnesla le 4 babuške.

Iz labirinta lahko odnese 5 babušk, če izpusti babuško velikosti 4. To lahko naredi na štiri načine:



Računalniško ozadje

Ta naloga zahteva iskanje najdaljšega naraščajočega zaporedja velikosti babušk preko vseh poti od vhoda do izhoda. Za posamezno pot je iskanje najdaljšega naraščajočega podzaporedja standardni algoritmični problem, ki ima učinkovito rešitev z dinamičnim programiranjem. Iskanje najdaljšega takšnega zaporedja preko več poti pa je bolj zapleten

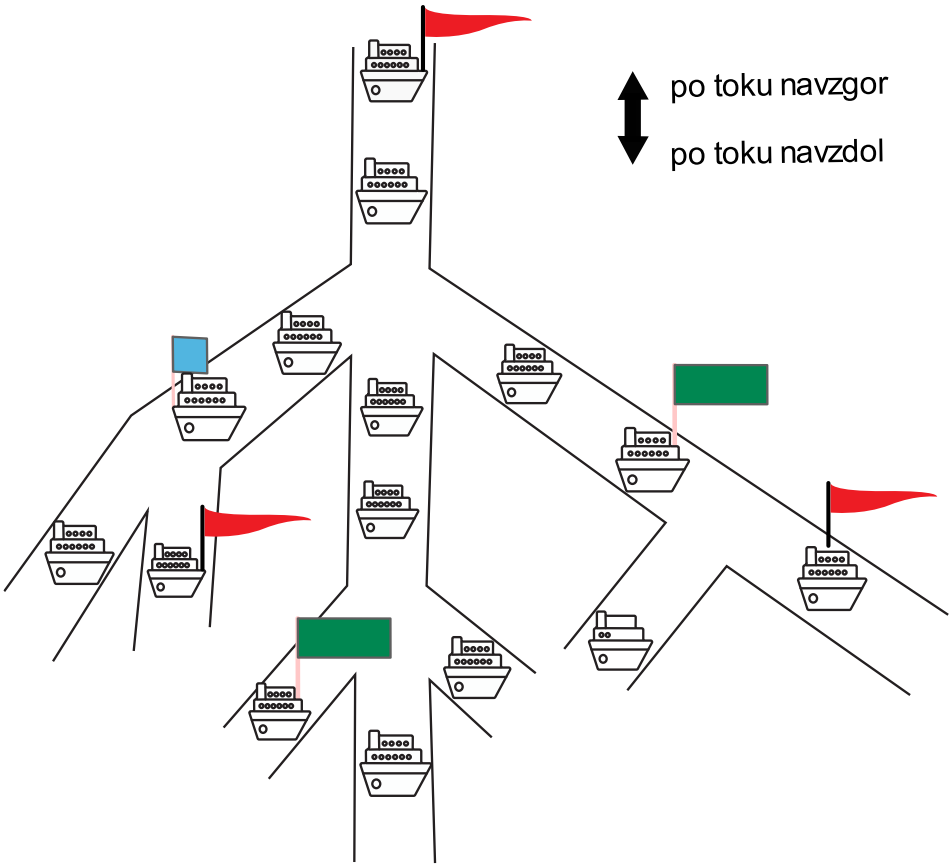
problem. V tem labirintu imajo poti preprosto strukturo (vse so enake dolžine, vodijo le gor in desno), zato je bila naša preprosta analiza zadostna, da smo našli odgovor.

Za bolj zapletene labirinte pa potrebujemo bolj splošen pristop, ki omogoča sistematično raziskovanje vseh možnih poti. Igranje iger, kot sta šah ali Go, vključuje reševanje podobnega problema. Pregledati moramo vsa možna zaporedja potez, da ugotovimo, katero vodi do najboljšega izida. V večini primerov ni smiselno raziskovati čisto vseh poti, zato poskušamo prepoznati zaporedja potez, ki ne vodijo do dobrega izida, in taka zaporedja zavržemo takoj, ko jih prepoznamo (torej predčasno prenehamo raziskovati take poti).

Ladje

6. do 9. razred

Med mestnim festivalom morajo biti vse ladje v delti reke okrašene z zastavicami različnih barv in oblik: na voljo so rdeče trikotne, modre kvadratne in zelene pravokotne zastavice. Vsaka ladja uporablja le eno vrsto zastavic. Nekatere ladje so že okrašene, kot kaže slika:

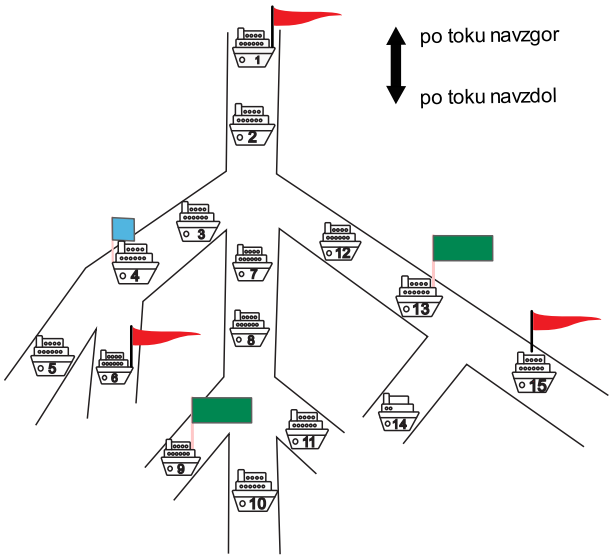
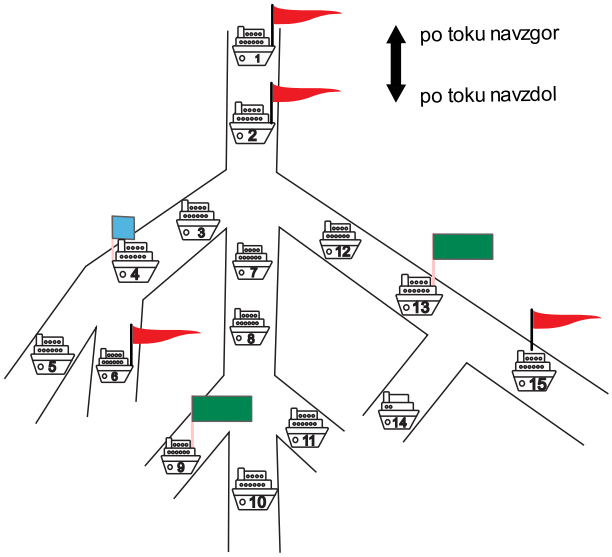
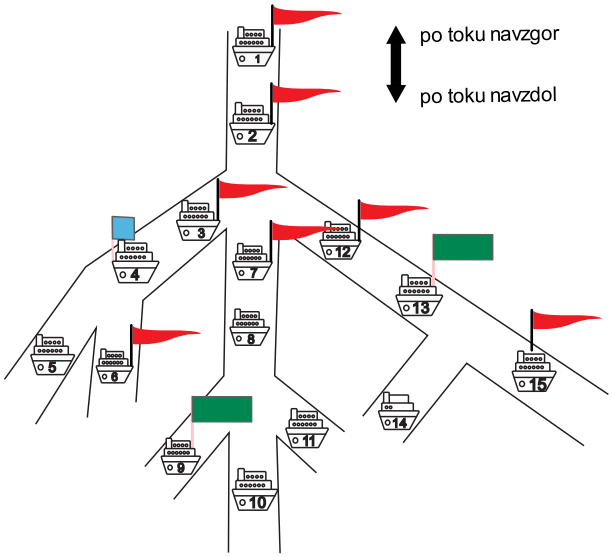


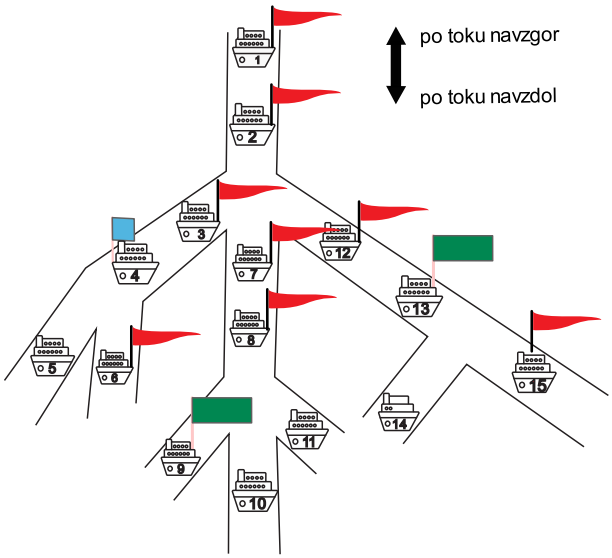
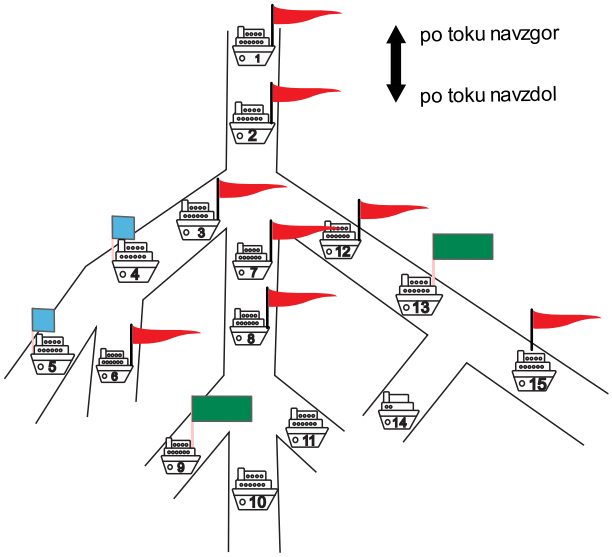
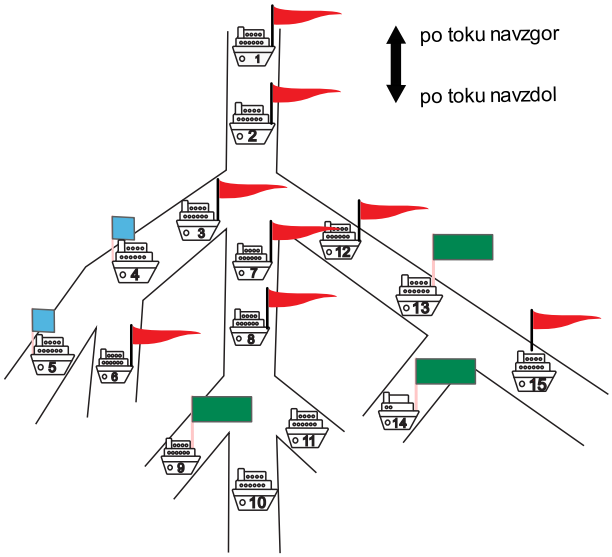
Mestni svet deli zastavice še neokrašenim ladjam, pri čemer uporablja naslednje pravilo: barva zastavice za neokrašeno ladjo mora biti enaka barvi zastavice, ki jo ima najbližja sosednja ladja **po toku navzgor (»zgornja« ladja)**. To pravilo uporabijo za vse še neokrašene ladje, dokler niso okrašene vse ladje v delti.

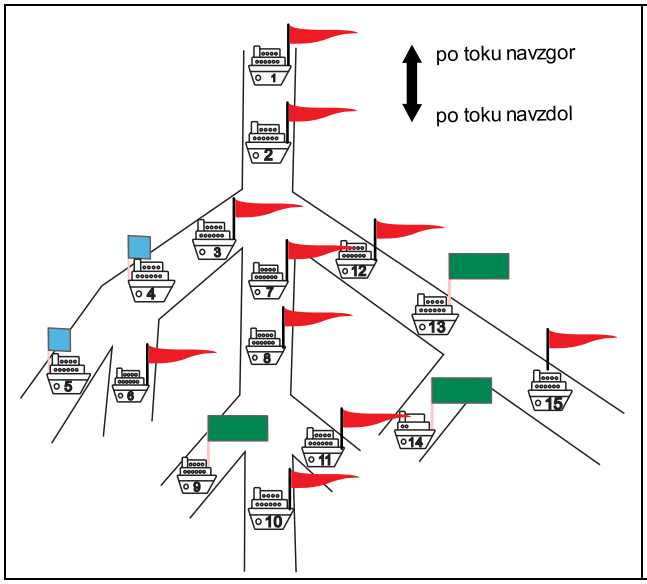
Koliko ladij na sliki bo okrašenih z rdečimi zastavicami?

Rešitev

Pravilen odgovor je 10.

	Najprej oštevilčimo ladje po toku navzdol, začenši z 1 pri ladji, ki je najbližje izviru.
	Najbližja soseda po toku navzgor od ladje številka 2 je ladja številka 1, ki ima rdečo zastavico, zato bo tudi na ladji 2 rdeča zastavica.
	Ladje 3, 7 in 12 imajo po toku navzgor za najbližjo sosedo ladjo 2, ki je dobila rdečo zastavico, zato tudi te tri ladje dobijo rdečo zastavico.

	Ladja 8 ima po toku navzgor za najbližjo sosedo ladjo 7, ki je dobila rdečo zastavico, zato tudi ladja 8 dobi rdečo zastavico.
	Ladja 5 ima za najbližjo sosedo ladjo 4, ki je okrašena z modro zastavico, zato tudi ladja 5 dobi modro zastavico.
	Ladja 14 ima po toku navzgor za najbližjo sosedo ladjo 13, ki je okrašena z zeleno zastavico, zato tudi ladja 14 dobi zeleno zastavico.



Ladji 10 in 11 imata po toku navzgor za najbližjo sosedo ladjo 8, ki je okrašena z rdečo zastavico, zato tudi ti dve ladji dobita rdečo zastavico.

Z rdečimi zastavicami bodo tako okrašene ladje 1, 2, 3, 6, 7, 8, 10, 11, 12 in 15. To je 10 ladij.

Računalniško ozadje

V tej nalogi je prikazan koncept *dedovanja*. Na rečno delto lahko gledamo kot na drevo in za ladjo, ki je najvišje po toku navzgor (v glavnem rokavu reke), bi lahko rekli, da je koren tega drevesa (to je ladja 1). Vse ostale ladje po toku navzdol so vozlišča tega drevesa. Ladje, ki se nahajajo najnižje po toku navzdol (torej nižje od njih ni nobene druge ladje), so listi drevesa. Vsaka ladja, ki je še brez zastavice, prevzame barvo zastavice ladje nad njo, torej neposredno po toku navzgor. Torej lahko rečemo, da ladja *podeduje* barvo zastavice od ladje/vozlišča, ki je neposredno nad njo v drevesu (od očeta).

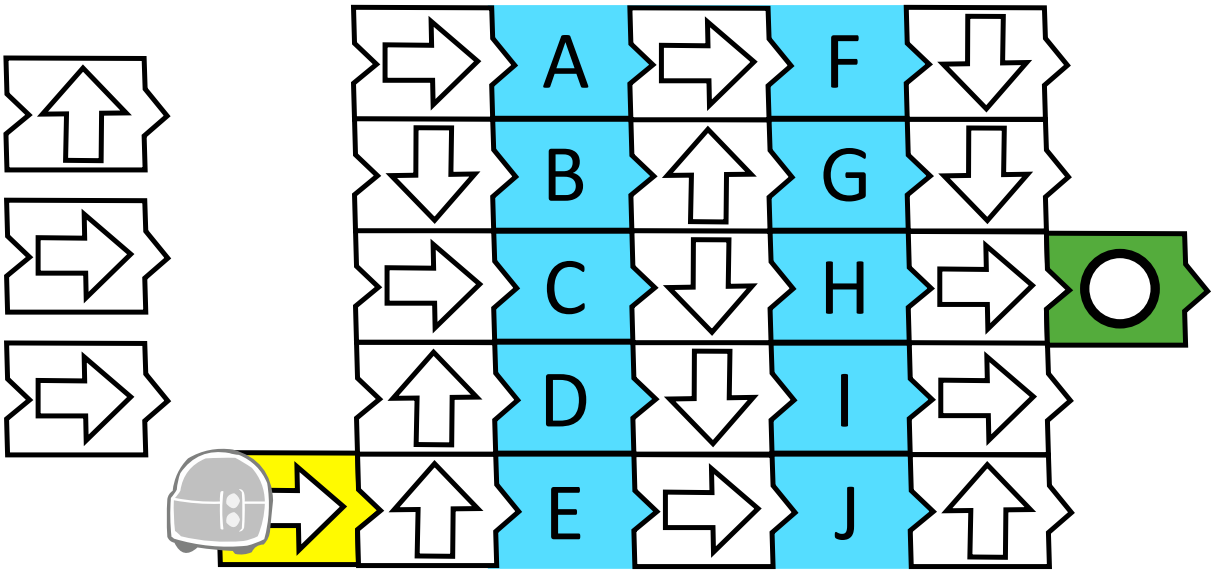
Pot za robota

6. do 9. razred

Tomov robot se premika po ploščicah, pri čemer sledi naslednjim pravilom:

- Robot začne na rumeni ploščici.
- Robot se premika v smeri puščice, ki je narisana na ploščici, na kateri je.
- Če robot zaide s ploščic, se ustavi.

Tom je sestavil progo iz ploščic, kot kaže slika. Ostale so mu še tri ploščice (dve ploščici s puščico desno in ena s puščico gor), ki jih mora položiti na označena mesta na modrem področju tako, da bo robot z rumene ploščice prišel do cilja na zeleni ploščici s krogom.



Na katero mesto mora Tom položiti ploščico s puščico gor, da bo robot prišel do cilja?

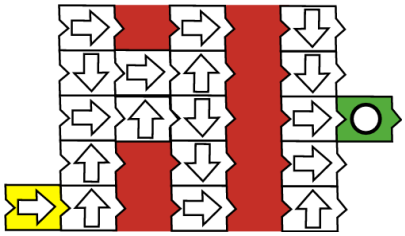
Rešitev

Tom mora ploščico s puščico gor položiti na mesto, označeno s črko C.

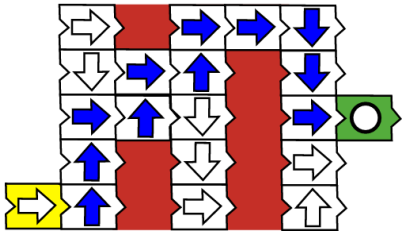
Če sledimo že postavljenim puščicam, hitro ugotovimo, da robot zaide na prazen prostor na mestu C.

Tom ima na voljo le ploščice s puščico gor in puščico desno, zato lahko na mesto C postavi eno od teh dveh ploščic:

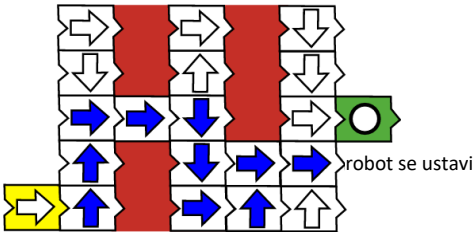
A) Če izbere ploščico s puščico gor, se robot za tem premakne na mesto B, kjer lahko Tom položi le ploščico s puščico desno.

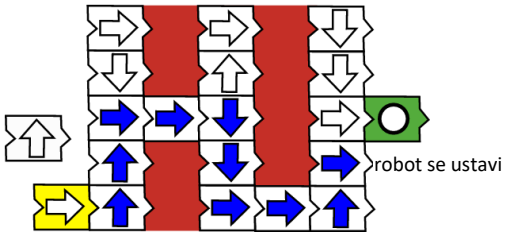


Ko robot sledi naslednjim ukazom, se premakne na prazen prostor na mestu F, kamor lahko Tom položi edino preostalo ploščico, to je ploščico s puščico desno. Sledeč nadaljnjim ukazom, robot pride do želenega cilja.



B) Če izbere ploščico s puščico desno, v nadaljevanju sledi puščicam do spodnje vrstice, kjer pride na prazen prostor na mestu J. Ne glede na to, ali na mesto J postavi puščico desno ali pa na mesto J postavi puščico gor in na mesto I puščico desno, bo robot zašel s ploščic tik pod ciljem in se ustavil. Oba primera lahko vidimo na naslednjih dveh slikah:





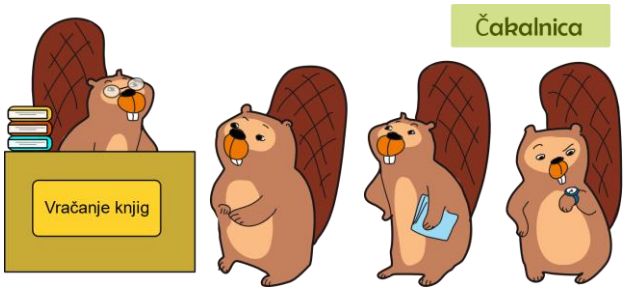
Računalniško ozadje

V tej nalogi robot ukaze bere s ploščic, poiskati pa smo morali pot, ki bo robota pripeljala od starta do cilja. Pri iskanju rešitve smo morali upoštevati podane *omejitve*: uporabili smo lahko le tri dodatne ploščice; imeli smo dve ploščini s puščico desno in eno ploščico s puščico gor; ploščice smo lahko postavili le na modra polja. Brez omejitev bi ta problem lahko imel več rešitev in bi ga bilo v splošnem težje rešiti, saj bi morali razmišljati o številnih možnih izidih.

Vračanje knjig

6. do 9. razred

V Trdolesju bobri obožujejo branje knjig. V knjižnici je običajno dolga vrsta čakajočih na vračilo knjig. Ker so vsi bobri prijazni, se je vodja knjižnice odločila, da uvede novo pravilo glede vračanja knjig, in sicer:



»Bober z najmanj knjigami je prvi na vrsti.«

Bobri prihajajo v knjižnico ob različnih časih in ne glede na to, kdaj pride bober v knjižnico, bo knjige najprej vrnil tisti bober v knjižnici, ki želi vrniti najmanj knjig. Knjižničar lahko v eni minuti obdela eno vrnjeno knjigo. Ko obdela vse knjige, ki jih je vrnil bober, pride na vrsto čakajoči bober, ki želi vrniti najmanj knjig.

Neko jutro je v knjižnico prišlo 5 bobrov, ki so želeli vrniti izposojene knjige. Njihovi časi prihoda in število knjig, ki jih želijo vrniti, so prikazani v spodnji tabeli:

Ime	**Čas prihoda**	**Število knjig za vračilo**
Ana	9.00	4
Beti	9.02	6
Cene	9.03	2
Darja	9.05	4
Emil	9.11	1

Ana je prišla takoj ob odprtju knjižnice, zato je knjižničar takoj začel s postopkom vračanja njenih knjig.

V kakšnem vrstnem redu bodo bobri vrnili knjige knjižničarju?

A) Ana, Beti, Cene, Darja, Emil
B) Ana, Cene, Beti, Darja, Emil
C) Ana, Cene, Darja, Beti, Emil
D) Emil, Cene, Ana, Darja, Beti

Rešitev

Pravilni odgovor je C: Ana, Cene, Darja, Beti, Emil.

Ana je prva prišla v knjižnico. Ker je edina v vrsti, lahko knjige vrne takoj. Ker vrne 4 knjige, z vračilom konča ob 9.04.

Ob 9.04 sta v čakalni vrsti dva bobra: Beti je prišla ob 9.02 s 6 knjigami, Cene pa ob 9.03 z 2 knjigama. Ker ima Cene le dve knjigi, je naslednji, ki ju bo vrnil. Cene konča vračanje knjig ob 9.06.

Ob 9.06 sta v čakalni vrsti še dva bobra: Beti s 6 knjigami in Darja, ki je prišla ob 9.05 s 4 knjigami. Ker ima Darja manj knjig za vrnitev kot Beti, je naslednja na vrsti. Darja konča z vračanjem knjig ob 9.10.

Ob 9.10 je v vrsti samo še Beti, zato vrne vse knjige ob 9.16. Ob 9.16 je v vrsti samo Emil, ki je prišel ob 9.11, in ima eno knjigo za vračilo, zato konča z vračanjem ob 9.17.

Odgovor A bi bil pravilen, če bi bil podlaga za obdelavo le čas prihoda.

Odgovor B je napačen, ker v tem vrstnem redu Beti vrne knjige pred Darjo, čeprav je Darja prišla v knjižnico z manj knjigami kot Beti, ko je bil prvi bober v vrsti poklican h knjižničarki.

Odgovor D bi bil pravilen, če bi upoštevali le število knjig.

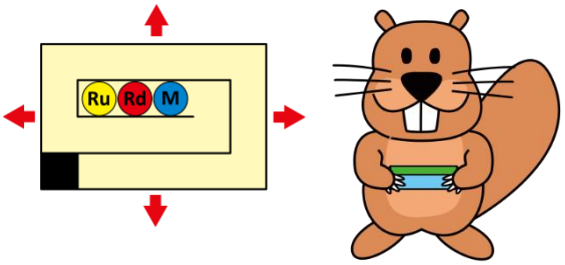
Računalniško ozadje

Naloga prikazuje delovanje razvrščevalnika. Razvrščevalnik je program, ki določa vrstni red dostopa procesov do centralne procesne enote (CPE). Razvrščevalnik določa, kateri procesi se izvajajo, kdaj in kako dolgo. Obstaja več različnih konceptov razvrščevalnika, kot so npr. *prvi pride, prvi melje* (angl. *First Come First Serve*), pri katerem se prvi izvede proces, ki je prvi v čakalni vrsti, vsak proces pa se izvede v celoti; *razvrščanje po prioriteti*, pri katerem se prvi izvede proces z najvišjo prioriteto, tudi tu se vsak proces izvede v celoti; *krožno razvrščanje* (angl. *Round Robin*), kjer se procesi izmenično izvajajo določen čas in če nek proces še ni v celoti izveden, se vrne v čakalno vrsto, izvajanje pa nadaljuje naslednji proces iz čakalne vrste; *najkrajši naprej* (angl. *Shortest Job First*), pri katerem se prvi izvede proces z najkrajšim predvidenim časom trajanja, vsak proces pa se izvede v celoti. V naši nalogi je knjižničar deloval po algoritmu *najkrajši najprej*.

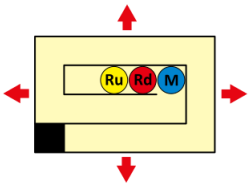
Bomboni

6. do 9. razred, srednja šola

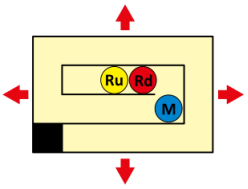
V škatlici so trije bomboni. V spodnjem levem kotu škatlice je odprtina, skozi katero lahko pade bombon. Bober Bitaro drži škatlico vodoravno. Nato naredi eno potezo: škatlico nagne v eno smer, da se bomboni premaknejo, in jo nazaj poravna. Poteze ponavlja, dokler iz škatlice ne dobi bombona.



Z nagibanjem zgornje škatlice v desno ➡ se bomboni premaknejo takole:



Nato se z nagibanjem škatlice navzdol ⬇ bomboni premaknejo takole:

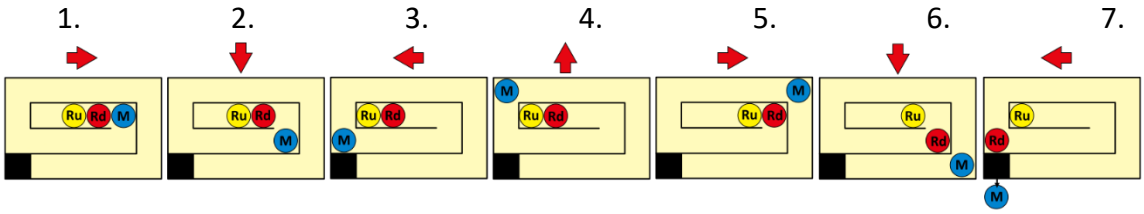


Najmanj koliko potez mora narediti Bitaro, da dobi iz škatlice rdeči bombon?

Rešitev

Da dobi rdeči bombon, mora Bitaro narediti najmanj 11 potez.

Da pride do rdečega bombona, mora najprej iz škatlice spraviti modri bombon. To naredi v sedmih potezah, ki so prikazane na sliki:



Hkrati z modrim bombonom se premikata tudi rdeči in rumeni bombon (tudi to je prikazano na zgornji sliki). Pri četrti potezi je rdeči bombon na svoji začetni poziciji. Zato mora po četrti potezi narediti še 7 enakih ponovitev potez, da spravi do odprtine še rdeči bombon. Poteze si sledijo po vrsti: desno ➡, dol ⬇, levo ⬅ in gor ⬆.

Skupaj torej potrebuje 11 potez.

Računalniško ozadje

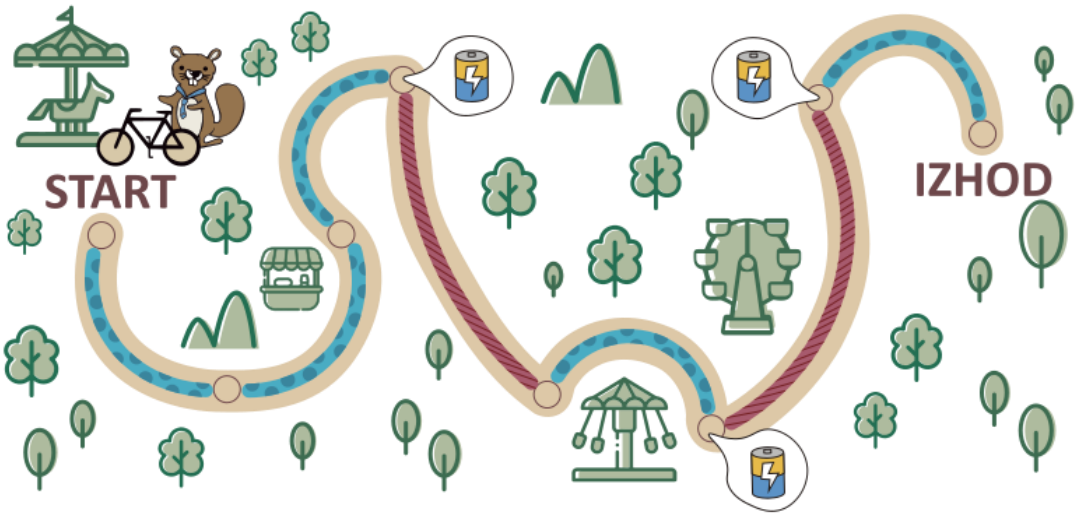
Z nagibanjem škatlice se trije bomboni premaknejo v smeri nagiba. Nagibanje škatlice si lahko zamislimo tudi kot ukaze robotom in na problem pogledamo kot na dajanje enakega ukaza trem različnim robotom (trem bombonom). Pri tem opazujemo, kako se roboti premikajo. A postavljene stene določajo, kako daleč se lahko posamezen robot premakne oziroma ali se sploh lahko premakne. Poleg tega lahko robot vpliva na premikanje drugega robota. Zato je programiranje robotov v praksi polno izzivov.

Električno kolo

6. do 9. razred, srednja šola

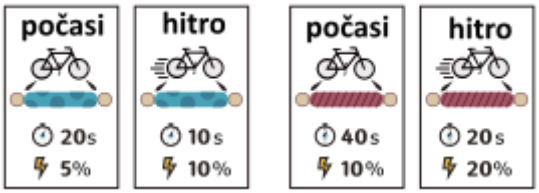
Ker se je zabaviščni park že zapiral, je bobrček Dejan vzel svoje električno kolo in pohitel proti izhodu.

Spodaj je prikazan zemljevid zabaviščnega parka. V njem sta dve vrsti poti: modri odseki poti in rdeči odseki poti. Na nekaterih mestih ob poti lahko Dejan zamenja baterijo na kolesu in s tem takoj napolni kolo ( ) za 20 odstotkov.



Dejanovo kolo ima dve hitrosti vožnje: počasi in hitro. Med vožnjo po posameznem odseku poti ne more zamenjati hitrosti; to lahko stori šele na koncu posameznega odseka poti.

Spodnja tabela prikazuje čas in odstotek energije (baterije), ki ju porabi za vožnjo preko posameznega odseka glede na hitrost vožnje:



Dejanovo kolo je na začetku (START) napolnjeno na 20 %, do izhoda pa mora priti, preden mu zmanjka energije.

Najmanj koliko časa potrebuje Dejan, da doseže izhod?

Rešitev

Do izhoda potrebuje Dejan najmanj 130 sekund.

Če želi Dejan priti do izhoda v najkrajšem možnem času, mora voziti z največjo možno hitrostjo. Pri tem pa mora seveda tudi paziti, da mu med potjo ne zmanjka energije. Zato mora porabo energije načrtovati glede na postaje, kjer lahko zamenja baterijo. Tako lahko pot do izhoda razdelimo na štiri dele (ker imamo na poti tri postaje za zamenjavo baterije), kot prikazuje tabela:

Del poti do izhoda	**Hitrost vožnje**	**Poraba energije**	**Preostanek energije**	**Porabljen čas**
od START do prve postaje	1 modri odsek, hitro	10	10	10
	2 morda odseka, počasi	2 x 5 = 10	0	2 x 20 = 40
od prve do druge postaje	1 rdeči odsek, počasi	10	10	40
	1 modri odsek, hitro	10	0	10
od druge do tretje postaje	1 rdeči odsek, hitro	20	0	20
od tretje postaje do IZHOD	1 modri odsek, hitro	10	10	10

Torej potrebujemo najmanj 10 + 40 + 40 + 10 + 20 + 10 = 130 sekund, da pridemo do izhoda.

Na prvem delu poti ne moremo voziti hitro preko vseh treh modrih odsekov, ker nimamo dovolj energije. Zato lahko hitro prepeljemo le enega od treh odsekov, druga dva pa moramo prepeljati počasi.

Na drugem delu poti ne moremo voziti hitro po rdečem odseku poti, saj bi s tem izpraznili baterijo, preden bi prišli do postaje za zamenjavo baterije.

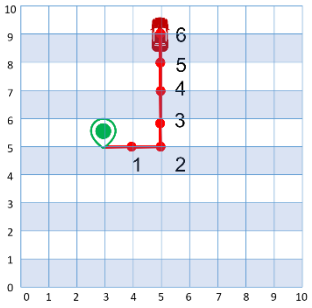
Računalniško ozadje

Tudi v tej nalogi smo reševali problem optimizacije, saj smo iskali najboljšo rešitev podanega problema. Pri podanih pogojih smo poiskali optimalno rešitev s pomočjo računanja ali iskanja. V primeru električnega kolesa iz naloge smo imeli vnaprej določene poti in tudi možen čas in porabo energije na vsakem odseku poti. Na podlagi danih pogojev smo poiskali najkrajši čas, v katerem lahko pridemo do izhoda, kar je pravzaprav iskanje najboljše (optimalne) rešitve.

Iskanje avta

6. do 9. razred, srednja šola

V mestu so ceste zgrajene v obliki mreže. Bobi se ne more spomniti, kje je parkiral svoj avto, ve pa, da ga je pustil na križišču dveh cest. Da bi poiskal svoj avto, hodi od enega križišča do drugega v smeri severa, juga, vzhoda ali zahoda. Bobi uporablja aplikacijo, ki mu na vsakem križišču pove, koliko križišč je oddaljen od svojega avta. Na primeru desno vidimo, da je Bobi (zelena točka na sliki) od avta (rdeč simbol avta) oddaljen 6 križišč.

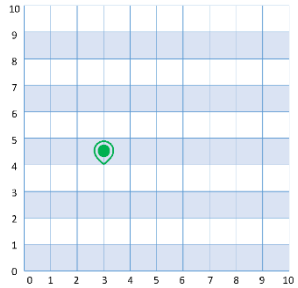


Na vsakem križišču se mora Bobi odločiti, v katero smer se bo odpravil v naslednjem koraku. Najprej je poskušal avto poiskati z naključno izbiro smeri v vsakem križišču, a je ugotovil, da tava v krogih, zato se je odločil, da se bo iskanja lotil bolj premišljeno.

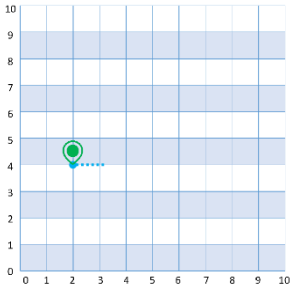
Ko Bobi obišče križišče prvič, uporabi naslednje pravilo: Če sem se premaknil dlje stran od avtomobila, se vrnem v prejšnje križišče, sicer nadaljujem v isti smeri.

Ko se Bobi vrne v križišče, ki ga je že obiskal, uporabi naslednje pravilo: nova smer, ki jo izbere, je naslednja v nasprotni smeri urinega kazalca.

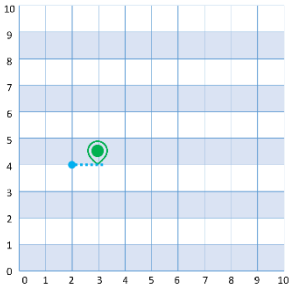
Tu je primer:

① 

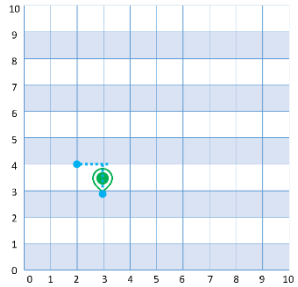
Izhodišče
5 križišč stran od avta

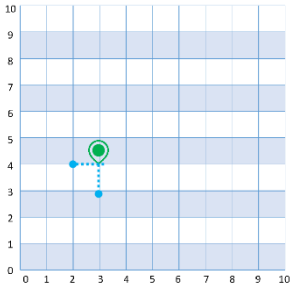
② 

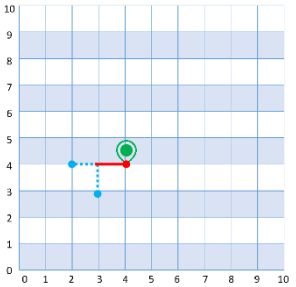
Naključno izbere premik na zahod
6 križišč stran od avta

③ 

Vrne se v prejšnje križišče
5 križišč stran od avta

④ 

⑤ 

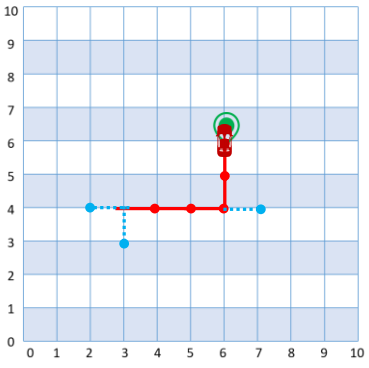
⑥ 

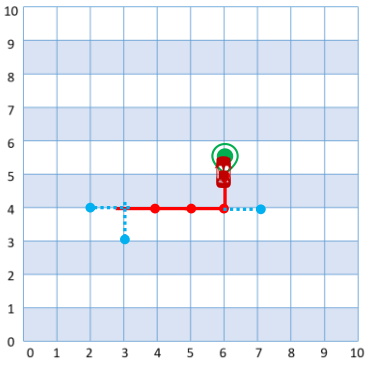
Premakne se proti jugu 6 križišč stran od avta	Vrne se v prejšnje križišče 5 križišč stran od avta	Premakne se proti vzhodu 4 križišča stran od avta, zato bo v naslednjem koraku nadaljeval proti vzhodu

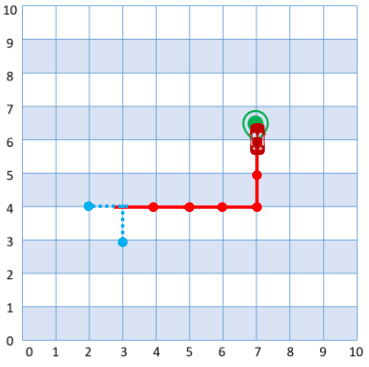
Vključno z izhodiščem in končnim križiščem, ki ga bober obišče, aplikacija poda naslednje podatke:

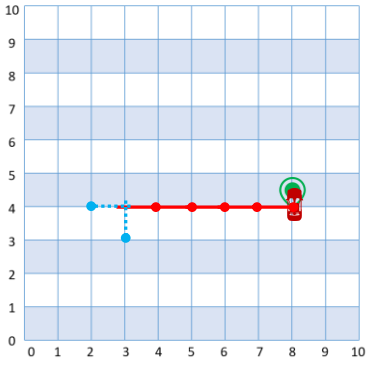
	①	②	③	④	⑤	⑥	⑦	⑧	⑨	⑩	⑪	⑫
Število križišč od avta	5	6	5	6	5	4	3	2	3	2	1	0

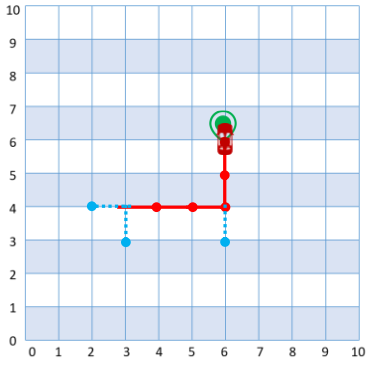
Katera od spodnjih slik pravilno prikazuje Bobijev premike?

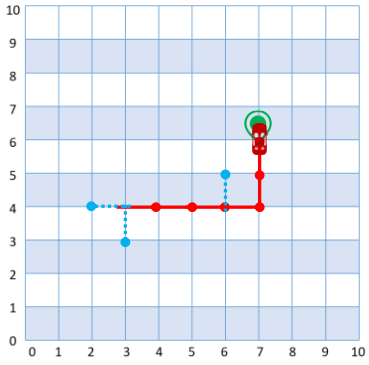
A) 

B) 

C) 

D) 

E) 

F) 

Rešitev

Pravilen odgovor je A.

Bobi prejme povratno informacijo o oddaljenosti avtomobila po vsakem premiku v naslednje križišče. Najprej naključno izbere smer zahod, vendar se, ker je oddaljenost pri ② naraščala, po premiku vrne v izhodišče ③. Nato se obrne v nasprotni smeri urinega kazalca proti jugu in se premakne za eno križišče ④. Ker je oddaljenost od avta zopet narasla, se vrne v izhodišče ⑤. Obrne se v nasprotni smeri urinega kazalca tako, da je obrnjen proti vzhodu in se premakne za eno križišče ⑥. Ker se oddaljenost od avta manjša, Bobi nadaljuje v smeri vzhoda, dokler se oddaljenost zopet ne poveča, to se zgodi v koraku ⑨. Bobi se vrne v križišče, v katerem je bil v koraku ⑧. Ko je v tem križišču ⑩, se obrne v nasprotni smeri urinega kazalca v smer severa in naredi korak naprej ⑪. Ker se je oddaljenost od avta zmanjšala, naredi še en korak in v koraku ⑫ najde svoj avto, saj je oddaljenost od njega 0. Avto se tako nahaja v točki (6,6).

Odgovor B ni pravilen, saj Bobi konča svoje korake eno križišče prezgodaj.

Odgovor C ni pravilen, ker se pri iskanju avta brez upoštevanja pravil obrne levo in ne upošteva, da se v koraku ⑨ razdalja poveča.

Odgovor D ni pravilen, ker ne upošteva, da se v koraku ⑨ oddaljenost od avta poveča in se mora Bobi vrniti v prejšnje križišče, pot pa nadaljevati v drugi smeri.

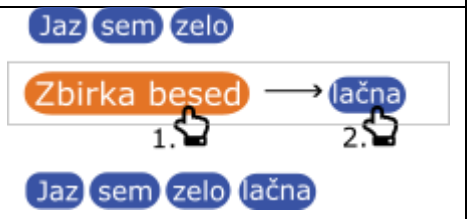
Odgovor E ni pravilen, ker v koraku ⑨ sicer upošteva, da se razdalja do avta poveča in se vrne v predhodno križišče, vendar se pred tem (v koraku ⑧) obrne desno, kar ni skladno s pravili.

Odgovor F ni pravilen, saj se v koraku ⑧ obrne levo na sever, čeprav bi moral nadaljevati naravnost, ker se razdalja do avta na tem koraku ni povečala. Nato se sicer vrne, kot zahtevajo pravila, a ponovno naredi napačen obrat in nadaljuje proti vzhodu (čeprav bi naslednja smer morala biti proti zahodu).

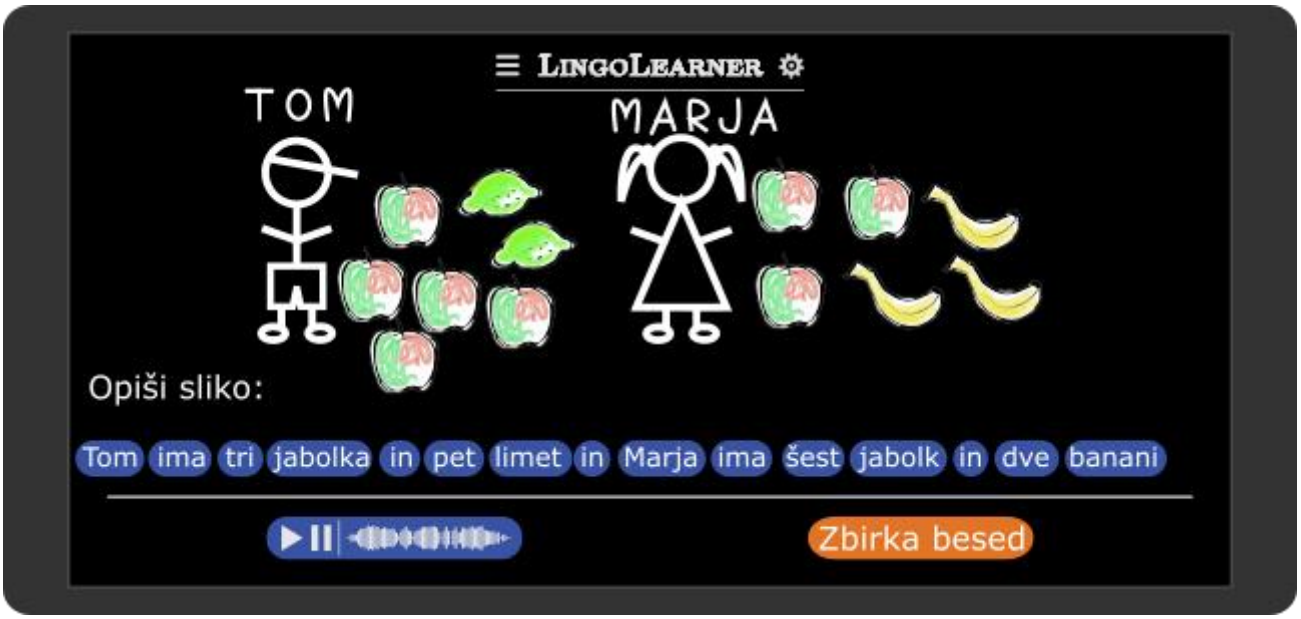
Računalniško ozadje

V nalogi je Bobi za iskanje avtomobila uporabil *požrešni algoritem* (angl. *greedy algorithm*). Na vsakem koraku je izbral najboljšo ali najugodnejšo možnost in tako poskušal priti do najboljšega ali najbolj optimalnega rezultata na splošno.

Liza se z uporabo aplikacije LingoLearner uči slovenščino. V aplikaciji stavke sestavlja iz besed, pri čemer ima na voljo dve možnosti urejanja odgovora:

Sprememba A: po kliku na katero koli besedo v stavku, ta beseda izgine, ostale besede pa se pomaknejo levo.	
Sprememba B: po kliku na gumb Zbirka besed lahko uporabnik izbere katero koli besedo iz te zbirke in jo doda na konec stavka.	

V prikazani aktivnosti mora Liza opisati sliko. Ko je končala svoj opis, je ugotovila, da je zamešala števila v povedi, kot kaže slika:



Najmanj koliko sprememb mora narediti Liza, da popravi poved?

Rešitev

Pravilna poved je taka:

Tom ima pet jabolk in dve limeti in Marja ima tri jabolka in tri banane.

Da bi uporabili kar najmanj potez, moramo iz povedi odstraniti napačne besede na tak način, da v povedi ostane čim več besed.

Začenši z leve proti desni strani, v povedi najdemo 6 pravilnih besed, ki smo jih odebelili in označili z zeleno:

Tom ima tri jabolka in **pet** limet in Marja ima šest **jabolk in dve** banani

Te lahko ostanejo v povedi, ostale besede pa izbrišemo s klikom na vsako od njih. Izbrisati moramo 9 besed.

Tom ima ~~tri~~ ~~jabolka~~ ~~in~~ **pet** ~~limet~~ ~~in~~ ~~Marja~~ ~~ima~~ ~~šest~~ **jabolk in dve** ~~banani~~

Pravilna poved ima enako število besed kot tista, ki jo je sestavila Liza, kar pomeni, da moramo stavku dodati 9 besed. To pomeni, da moramo za pravilno sestavljeno poved uporabiti 9 potez A in 9 potez B, to je skupaj 18 sprememb.

Računalniško ozadje

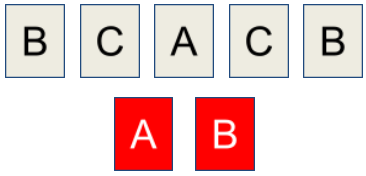
Najbolj preprosta rešitev, ki jo vidimo, je, da enostavno izbrišemo vse besede iz povedi in poved napišemo na novo. Ampak v računalništvu pogosto iščemo načine, kako optimizirati rešitve problema. Tudi v tej nalogi smo morali optimizirati (to je zmanjšati) število potez, ki so bile potrebne za popravek besedila.

Najvišji rezultat

6. do 9. razred, srednja šola

Imamo zaporedje kart z oznakami A, B in C. Za to zaporedje kart izračunamo rezultat tako, da vsako skupino zaporednih črk točkujemo s toliko točkami, kot je dolgo zaporedje enakih črk. Tako vsaka skupina dveh zaporednih enakih črk prejme 2 točki, vsaka skupina treh zaporednih enakih črk prejme 3 točke in tako naprej. Rezultat je vsota vseh prejetih točk.

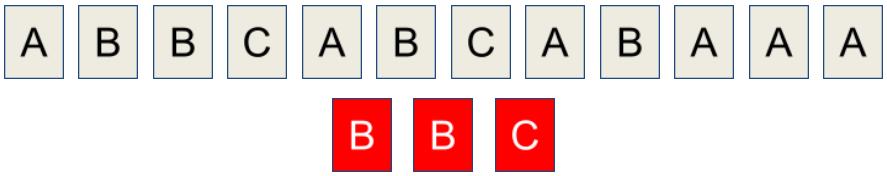
Na primer, spodnje zaporedje petih sivih kart ima rezultat 0 točk, ker nima nobenega zaporedja enakih črk.



Vendar imamo še dve rdeči karti, s katerima lahko prekrijemo poljubno sivo karto iz zaporedja tako, da izboljšamo rezultat zaporedja. Tako lahko prvo črko C prekrijemo z rdečo karto A, drugo črko C pa z rdečo karto B:

Tako dobimo dve zaporedji enakih črk dolžine 2, torej je rezultat tega zaporedja kart 2 + 2 = 4 točke.

Poglejmo novo zaporedje 12 sivih kart in še dodatne 3 rdeče karte:



Z rdečimi kartami B, B in C v zaporedju sivih kart prekrijemo katere koli tri sive karte tako, da dobimo najvišji možni rezultat. Kam postavimo rdeče karte?

Rešitev

Pravilen odgovor je:

To zaporedje kart ima rezultat 2 + 2 + 4 + 3 = 11 točk.

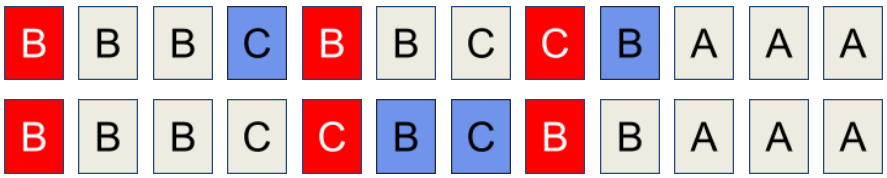
Najprej lahko ugotovimo, da velikost skupine, ki ji pripada črka, pri rezultatu ne igra nobene vloge, saj vsaka posamezna črka v skupini prinese natanko 1 točko. Torej je vseeno, koliko

črk strnemo v eno skupino, v zaporedju se moramo izogniti le posameznim črkam (ki niso v skupini), saj te ne prinesejo točk. Take posamezne črke se nahajajo na mestih 1, 4, 5, 6, 7, 8 in 9. Le tri od njih lahko prekrijemo z rdečo karto.

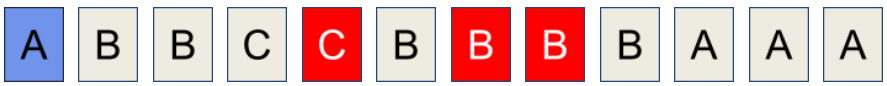
1	2	3	4	5	6	7	8	9	10	11	12
A	B	B	C	A	B	C	A	B	A	A	A

Druga ugotovitev je, da samostojne črke A (te so na mestih 1, 5 in 8) ne moremo postaviti v skupino, saj nimamo nobene rdeče karte A. Torej poskusimo te črke prekriti z rdečimi kartami B ali C, da ustvarimo skupine s sosednjimi črkami. Vseeno je, kako velike skupine ustvarimo (glede na prvo ugotovitev).

Recimo, da z rdečimi kartami prekrijemo vse tri samostojne črke A: prvo prekrijemo z B, da jo pridružimo skupini B-jev, drugo (na mestu 5) lahko prekrijemo ali z B ali s C (združimo v skupino na desni ali na levi), tretjo (na mestu 8) pa podobno lahko prekrijemo ali z B ali s C (združimo v skupino na desni ali na levi). V obeh primerih prekrivamo z le eno rdečo karto podzaporedje C A B, kar pomeni, da nam na koncu vedno ostaneta še dve samostojni karti, ne glede na to, s katero rdečo karto (B ali C) prekrivamo. Torej lahko s takim prekrivanjem dosežemo največ 10 točk (če bi bile vse karte zaporedja v skupini, bi dosegli 12 točk, vsaka samostojna karta, ki ni del skupine, pa ta rezultat zmanjša za eno točko). Na slikah smo z modro označili posamezne karte, ki niso del skupine.



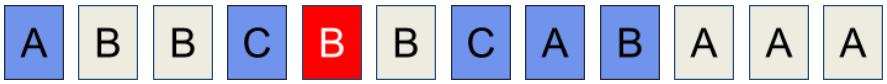
Poglejmo še drugo možnost, ko prve samostojne karte A (na mestu 1) ne vključimo v skupino, vse ostale samostojne karte pa poskusimo vključiti v neko skupino. Preostale samostojne karte so na zaporednih mestih 4, 5, 6, 7, 8 in 9 ter tvorijo podzaporedje C A B C A B. Če prvi A tega podzaporedja prekrijemo z rdečo karto C, dobimo skupino dveh kart C (podzaporedje se spremeni v C C B C A B). Ostaneta nam še dve rdeči karti B. S tema prekrijemo karti C in A na 7. in 8. mestu ter ju s tem združimo v skupino s kartama B, ki sta pred in za kartama C in A. Na ta način smo vse samostojne karte na mestih od 4 do 9 uspeli združiti v skupine. Torej v celem zaporedju ostane le ena samostojna karta, to je karta A na prvem mestu, ki je ne moremo vključiti v nobeno skupino. Rezultat je torej 11 točk.



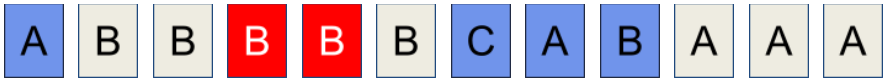
Tako smo pokazali, da je 11 točk najvišji možni rezultat. Poglejmo še, da je to res edina rešitev, ki da najvišji rezultat.

Ugotovili smo že, da najvišji rezultat lahko dobimo le tako, da prve karte A ne prekrijemo (in ta ostane edina karta, ki ni v skupini). Torej moramo prekriti preostali samostojni karti A (na

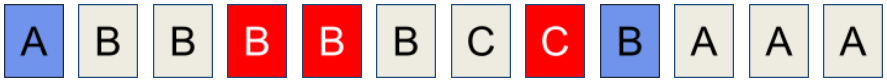
mestu 5 in 8). Rešitev, ko karto A na 5. mestu prekrijemo z rdečo karto C, smo preverili že zgoraj. Druga možnost je, da to karto prekrijemo z rdečo karto B.



V tem primeru ostane karta C na mestu 4 izven skupine in če te karte ne prekrijemo, ne moremo dobiti najvišjega rezultata, saj bi imeli izven skupin vsaj dve karti (A na prvem mestu in C na četrtem mestu). Torej jo moramo prekriti z rdečo karto B.



Ostane nam le še ena rdeča karta, to je karta C. Z njo lahko prekrijemo karto A na 8. mestu, da ustvarimo skupino dveh kart C.



Vendar pa ostane v zaporedju (poleg prve karte A) še ena samostojna karta, to je karta B na 9. mestu. Te ne moremo prekriti, saj nam je zmanjkalo rdečih kart. Torej tudi v tem primeru lahko dobimo rezultat le 10 točk, kar ni najvišji možni.

Računalniško ozadje

Z vidika računalništva smo se v nalogi ukvarjali s problemom optimizacije. Optimizacija pomeni, da naredimo nekaj tako učinkoviti ali tako funkcionalno, kot je le mogoče z upoštevanjem podanih omejitev. Optimizacija torej pomeni, da poiščemo najboljšo izmed vseh možnih rešitev.

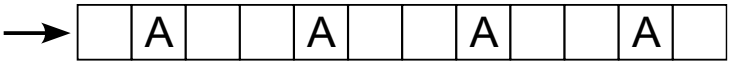
Stroj za zapis nizov

6. do 9. razred, srednja šola

Stroj za zapis nizov po navodilih zapisuje nize dolžine do 12 znakov. Stroj prebere ukaz, sestavljen iz treh vrednosti:

- prva vrednost je znak, ki se bo ponavljal v nizu,
- druga vrednost pove, na katerem mestu v nizu se bo znak prvič pojavil in
- tretja vrednost pove, koliko razmaka je med ponovitvami znaka.

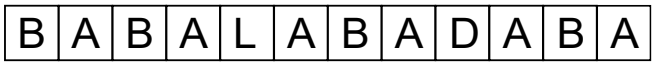
Primer: ko stroj prejme ukaz »A 2 3«, zapiše znak »A« na 2., 5., 8. in 11. mesto v 12-mestnem nizu:



Stroj za zapis nizov lahko izvaja več zaporednih ukazov. Ob vsakem novem ukazu začne stroj ukaze izvajati na začetku niza. Če je na nekam mestu že zapisan znak, ga ne prepiše.

Primer: ko stroj prejme ukaza »A 2 3« in »B 3 2« zapiše naslednji niz:

Kakšno zaporedje ukazov mora prejeti stroj za zapis nizov, da sestavi naslednji niz:



A) A 2 2, L 5 4, D 9 5, B 1 2

B) D 9 5, L 5 4, B 1 2, A 2 2

C) A 2 2, B 3 4, D 9 5, L 5 4

D) B 1 2, A 2 2, L 5 4, D 9 5

E) L 5 10, D 9 3, A 2 2, B 1 2

Rešitev

Pravilen odgovor je B. Poglejmo, kaj se zgodi, ko stroj za zapis nizov izvede posamezen ukaz:

Ukaz	1	2	3	4	5	6	7	8	9	10	11	12
D 9 5									D			
L 5 4					L				D			
B 1 2	B		B		L		B		D		B	
A 2 2	B	A	B	A	L	A	B	A	D	A	B	A

Z zaporedjem ukazov pri odgovoru A bi dobili naslednji rezultat:

Ukaz	1	2	3	4	5	6	7	8	9	10	11	12
A 2 2		A		A		A		A		A		A
L 5 4		A		A	L	A		A	L	A		A
D 9 5		A		A	L	A		A	L	A		A
B 1 2	B	A	B	A	L	A	B	A	L	A	B	A

Z zaporedjem ukazov pri odgovoru C bi dobili naslednji rezultat:

Ukaz	1	2	3	4	5	6	7	8	9	10	11	12
A 2 2		A		A		A		A		A		A
B 3 4		A	B	A		A	B	A		A	B	A
D 9 5		A	B	A		A	B	A	D	A	B	A
L 5 4		A	B	A	L	A	B	A	D	A	B	A

Z zaporedjem ukazov pri odgovoru D bi dobili naslednji rezultat:

Ukaz	1	2	3	4	5	6	7	8	9	10	11	12
B 1 2	B		B		B		B		B		B	
A 2 2	B	A	B	A	B	A	B	A	B	A	B	A
L 5 4	B	A	B	A	B	A	B	A	B	A	B	A
D 9 5	B	A	B	A	B	A	B	A	B	A	B	A

Z zaporedjem ukazov pri odgovoru E bi dobili naslednji rezultat:

Ukaz	1	2	3	4	5	6	7	8	9	10	11	12
L 5 10					L							
D 9 3					L				D			D
A 2 2		A		A	L	A		A	D	A		D
B 1 2	B	A	B	A	L	A	B	A	D	A	B	D

Računalniško ozadje

Tudi računalnik je stroj, ki izvaja zaporedje ukazov. Zaporedja računalniku razumljivih ukazov predstavljajo računalniške programe.

Skrivna sporočila

8. in 9. razred

Nekaj bobrov mora zašifrirati besedilo. Za ta namen uporabljajo spodnjo šifrirno tabelo. Vsak znak zašifrirajo s tremi števili. Prvo število je zaporedna številka stolpca in drugo zaporedna številka vrstice, v katerih se znak nahaja. Tretje število predstavlja pozicijo znaka v celici.

Tabela izgleda takole:

AB	CČD	EF	GH
IJ	KL	MN	OP
RSŠ	TU	VZ	Ž_

Besedo »KARO« zakodirajo kot: 221-111-131-421. Znak »_«, ki je v tabeli za znakom Ž, predstavlja presledek med besedami.

Kako bi bobri zakodirali »LEP DAN«?

A) 222-311-422-432-213-111-322
B) 222-131-242-342-123-111-232
C) 222-113-224-234-312-111-223
D) 222-311-422-423-231-111-322

Rešitev

Pravilen odgovor je A. Besedno zvezo LEP DAN zašifrirajo tako:

ZNAK	Številka stolpca	Številka vrstice	Pozicija znaka v celici	KODA
L	2	2	2	222
E	3	1	1	311
P	4	2	2	422
_	4	3	2	432
D	2	1	3	213
A	1	1	1	111
N	3	2	2	322

V odgovoru B so črke zašifrirane tako, da je prvo število številka vrstice in drugo število številka stolpca.

V odgovoru C je zamenjan vrstni red števil tako, da prvo število predstavlja zaporedno število znaka v celici, drugo število številko vrstice, tretje število pa številko stolpca, v katerem se nahaja znak.

V odgovoru D je zamenjan vrstni red števil tako, da prvo število predstavlja številko stolpca, drugo število zaporedno številko znaka v celici in tretje število številko vrstice, v katerem se nahaja znak.

Računalniško ozadje

Šifriranje je postopek preoblikovanja podatkov z uporabo algoritmov, tako da poslanih podatkov ne more prebrati nihče razen tistih, ki imajo ključ. Bobri kot ključ za šifriranje uporabljajo stolpec, vrstico in pozicijo črke v celici tabele. Brez tabele (ključa) bi nekdo težko dešifriral (obrnil šifriranje) in dobil izvirno sporočilo.

Šifriranje se uporablja za shranjevanje in pošiljanje vseh vrst občutljivih podatkov. Gesla ali vaši osebni podatki so na primer pri shranjevanju običajno šifrirani.

Kamen, škarje, papir

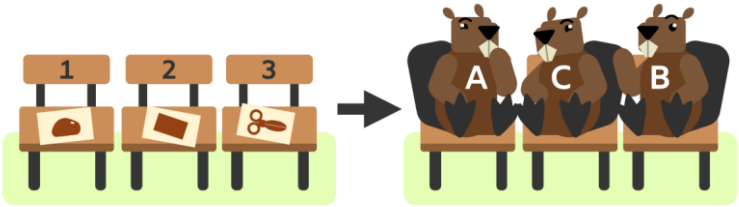
8. in 9. razred, srednja šola

Anja, Breda in Cene igrajo različico igre kamen, škarje, papir. Začne se s tremi oštevilčenimi stoli. Na vsakem od stolov je eden od listkov, na katerem je narisan kamen, škarje ali papir, kot prikazuje spodnja slika.

Ne pozabite:

- Kamen premaga škarje.
- Škarje premagajo papir.
- Papir premaga kamen.

Anja, Breda in Cene se usedejo na stole, kot kaže slika.



Zgodijo se naslednje akcije:

1. Najprej vsak od prijateljev listek s stola, na katerem sedi, zapakira v pisemsko ovojnico, tako da drugi ne morejo videti, kaj je narisano na listku.
2. Potem dva prijatelja zamenjata stola in ovojnici.
3. Nato dva prijatelja zamenjata ovojnici, a ne stolov.
4. Nazadnje dva prijatelja zamenjata stola, a obdržita ovojnici.

Po vseh teh menjavah prijatelji odprejo pisemske ovojnice, ki jih imajo v rokah in primerjajo sličice na papirju, ki je v njih.

Katera od naslednjih izjav drži?

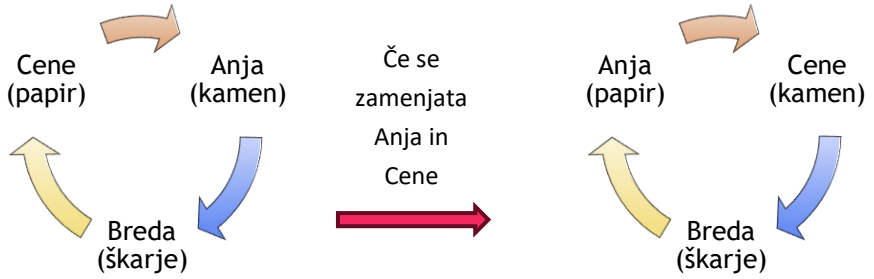
A) Cene bo premagal Bredo.
B) Breda bo premagala Anjo.
C) Cene bo premagal Anjo.
D) Za noben par ni mogoče določiti, kdo bo zmagal.

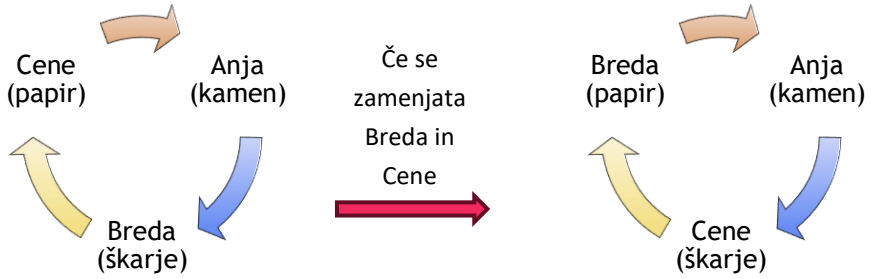
Rešitev

Pravilen odgovor je C, Cene bo premagal Anjo.

Če pogledamo začetno stanje: Anja premaga Berto, Berta premaga Ceneta in Cene premaga Anjo.

Ko dva prijatelja zamenjata mesti, ne pa tudi pisemskih ovojnic, je vseeno, kdo se zamenja, saj v vsakem primeru dobimo obraten vrstni red zmag. Na primer, če se zamenjata Anja in Cene, potem zdaj Cene premaga Bredo, Breda Anjo in Anja Ceneta. Na spodnjih slikah puščice kažejo od zmagovalca k poraženca.





Kot vidimo s slik, se vrstni red zmag v vsakem primeru zamenja, razlika je le, kdo ima v roki kateri listek. Iz tega razloga vemo, da je po prvi zamenjavi (2. korak) rezultat takšen: Anja premaga Ceneta, Cene premaga Bredo in Breda premaga Anjo.

V naslednjem koraku dve osebi ponovno zamenjata kuverti, ne pa tudi stolov, zato se tako kot prej stanje zmag ponovno obrne: Anja premaga Bredo, Breda premaga Ceneta in Cene premaga Anjo.

V zadnjem koraku dva prijatelja zamenjata mesti, a obdržita isti ovojnici, zato v tem primeru ne pride do spremembe v zmagah. Končni rezultat je, da Anja premaga Bredo, Breda premaga Ceneta in Cene premaga Anjo. Ker je edina od trditev, ki drži, da Cene premaga Anjo, je pravilen odgovor C.

Računalniško ozadje

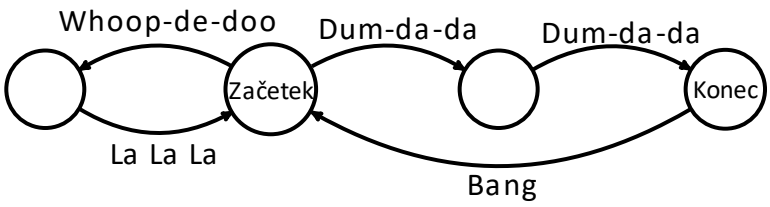
Pri tej nalogi analiziramo, kaj se zgodi v določeni situaciji, ko se izvedejo različne spremembe. Zanimivo je, da se vse trditve "X premaga Y" spremenijo v "Y premaga X", kadar dve osebi zamenjata ovojnici. Po drugi strani pa se ne zgodi nič, kadar igralca zamenjata sedeža (ne pa tudi ovojnic), saj sedež ni pomemben za izid igre kamen, škarje, papir. Naloga se tako ukvarja z *invariantami*, to je iskanjem stvari, ki se v določenih okoliščinah ne spremenijo, ali prepoznavanjem okoliščin, v katerih so stvari enake kot prej.

Invariante so ključnega pomena za matematiko in računalništvo, saj nam, če jih najdemo, omogočijo zmanjšanje števila primerov, ki jih je treba analizirati, kar olajša upravljanje in reševanje problema.

Lizine pesmice

8. in 9. razred, srednja šola

Liza si rada izmišljuje nove pesmice, ampak ne kakršnih koli. Njene pesmice morajo upoštevati pravila, ki so prikazana na sliki:



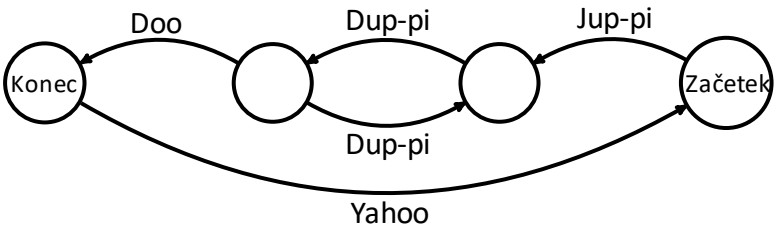
Da bi zapela eno od svojih pesmic, mora začeti pri oznaki Začetek in slediti puščicam v ustreznem vrstnem redu, zaključiti pa pri oznaki Konec. Ko pride do oznake Konec, ni nujno, da takoj preneha s petjem. Svojo pesmico lahko tudi nadaljuje v smeri puščic, a mora vedno končati pri oznaki Konec.

Tu sta dva primera njenih pesmic:

1) Whoop-de-doo La La La Whoop-de-doo La La La
   Dum-da-da Dum-da-da Bang Dum-da-da Dum-da-da

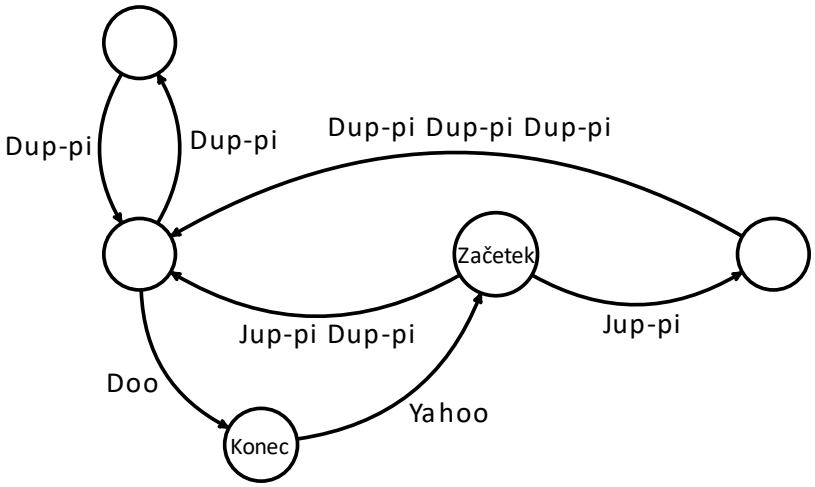
2) Dum-da-da Dum-da-da Bang Whoop-de-doo La La La
   Dum-da-da Dum-da-da Bang Whoop-de-doo La La La
   Dum-da-da Dum-da-da Bang Dum-da-da Dum-da-da

Liza pa se včasih odloči in zapoje tudi kakšno alternativno pesmico. Takrat uporabi naslednje pravilo:

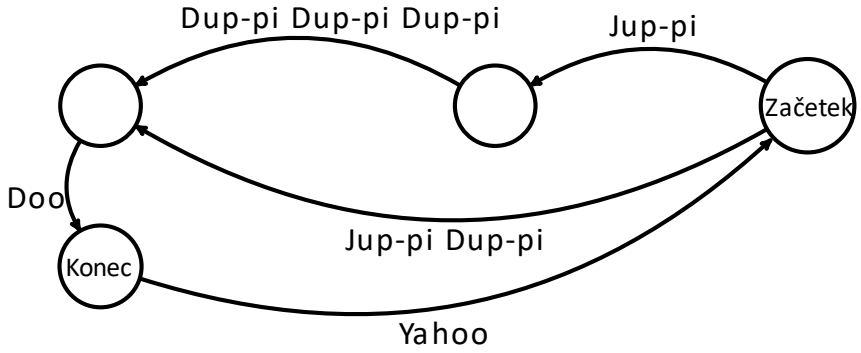


Kateri od naslednjih diagramov prikazuje navodilo za Lizine alternativne pesmice?

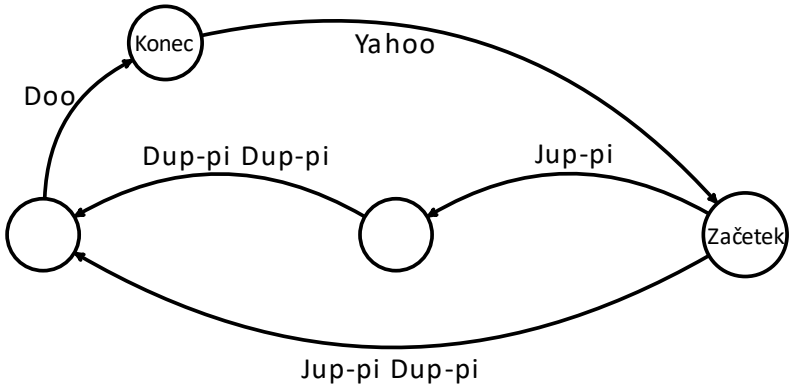
A)


Dup-pi
Dup-pi
Dup-pi Dup-pi Dup-pi
Začetek
Jup-pi Dup-pi
Jup-pi
Doo
Yahoo
Konec

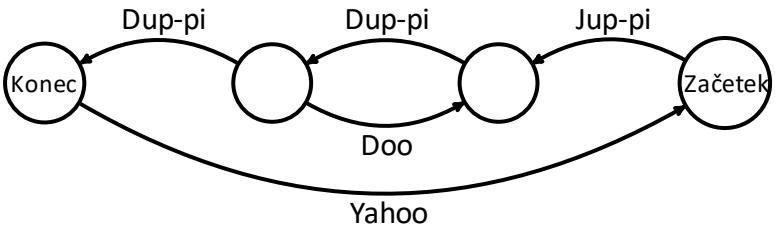
B)


Dup-pi Dup-pi Dup-pi
Jup-pi
Začetek
Doo
Jup-pi Dup-pi
Konec
Yahoo

C)


Konec
Yahoo
Doo
Dup-pi Dup-pi
Jup-pi
Začetek
Jup-pi Dup-pi

D)



Rešitev

Pravilen odgovor je A.

Odgovor A je pravilen, saj lahko na podlagi tega navodila sestavimo enake pesmice, kot je alternativna. Pri tem se, podobno kot v podanem pravilu za alternativno pesem, pesem vedno začne z Jup-pi, besede Dup-pi pa se v pesmi vedno ponavljajo lihokrat zapored, preden jim sledi beseda Doo.

Odgovor B ni pravilen, saj se v pesmici za Jup-pi beseda Dup-pi lahko ponovi le enkrat ali trikrat, ne pa večkrat (npr. tudi petkrat, sedemkrat …), kot to predvideva prvotni načrt alternativne pesmi.

Odgovor C ni pravilen, saj se za Jup-pi beseda Dup-pi v posameznem delu pesmi (med začetkom in koncem) pojavi le enkrat ali dvakrat.

Odgovor D ni pravilen, saj se pesem ne zaključi z besedo Doo, ampak z besedo Dup-pi.

Računalniško ozadje

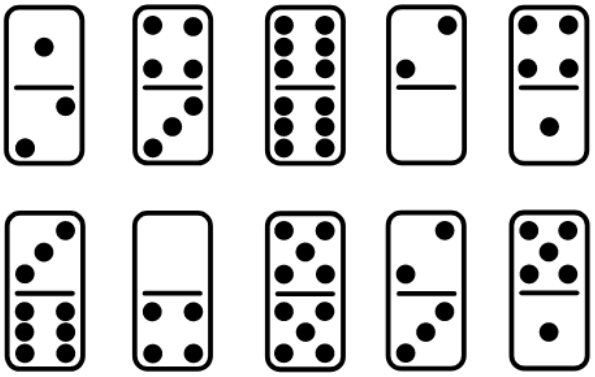
Računalništvo se v veliki meri ukvarja s prepoznavanjem in uporabo struktur v podatkih. V tej nalogi obravnavamo strukturirana besedila (pesmi), ki so oblikovana po strogem sistemu pravil. Mehanizmi za generiranje (diagrami) se imenujejo končni avtomati. Ti imajo ključno vlogo pri oblikovanju programskih jezikov, tj. jezikov, ki jih lahko „razumejo“ računalniki.

Pri reševanju te naloge je bilo ključnega pomena prepoznati vzorec ali podobne značilnosti problema, kar nam je pomagalo pri razčlenitvi problema in iskanju rešitve.

Ugani domino

8. in 9. razred, srednja šola

Alica in Bob igrata igro ugibanja domin. Na mizi imata naslednjih 10 domin:



Bob izbere eno od domin. Alica mora z vprašanji, na katera Bob odgovarja le z da ali ne, ugotoviti, katero domino je izbral.

Alica mora svoja vprašanja oblikovati tako, da bo ne glede na Bobov odgovor v kar najmanj korakih odkrila, katero domino je izbral Bob.

Katero vprašanje naj Alica postavi kot prvo?

A) Ali je vsota pik na domini večja ali enaka 8?
B) Ali je na tisti strani domine, kjer je več pik, število pik večje ali enako 4?
C) Ali je na tisti strani domine, kjer je manj pik, število pik večje ali enako 2?
D) Ali imata obe strani domine enako število pik?
E) Vseeno je, katero od navedenih vprašanj Alica postavi kot prvo.

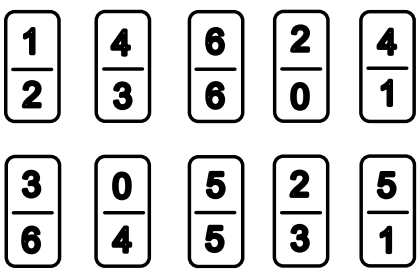
Rešitev

Pravilen odgovor je E.

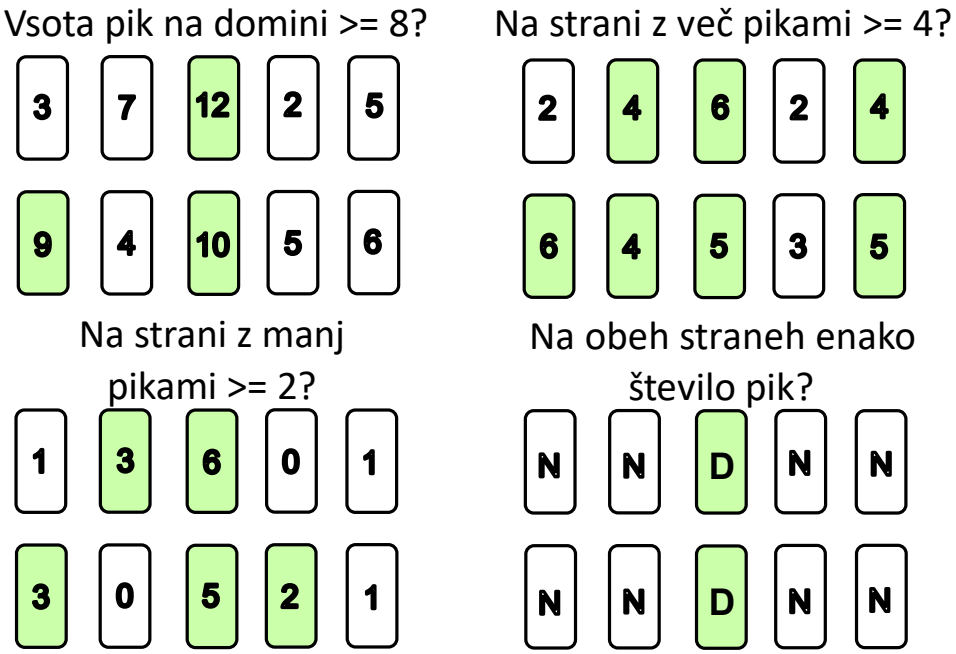
Za vsakega od možnih odgovorov preverimo, koliko vprašanj moramo postaviti v najslabšem primeru, da ugotovimo iskano domino.

V igri Alica in Bob uporabljata naslednje domine:

10 domin na mizi



Naslednja slika za vsako možno vprašanje prikazuje, katere domine bi ostale, če bi Bob odgovoril z da (zeleno) ali z ne (belo). Na vsaki domini so prikazani le podatki, ki jih upoštevamo pri odgovarjanju na zapisano vprašanje.



Vidimo, da pri vseh štirih vprašanjih razdelimo 10 domin na dva dela, kjer en del vsebuje 5, 7 ali 8 domin, drugi pa 5, 3 ali 2 domini.

Če nam ostaneta dve domini, lahko z enim dodatnim vprašanjem ugotovimo, katero domino iščemo.

Če ostanejo tri domine, lahko z naslednjim vprašanjem te domine razdelimo na dva dela, po 1 in 2 domini, a v najslabšem primeru (če nam ostaneta dve domini) potrebujemo še dodatno vprašanje, da uganemo iskano domino. Torej potrebujemo skupaj 3 vprašanja.

Če ostane pet domin, jih lahko z drugim vprašanjem razdelimo v skupini po 2 in 3 domine. In nato v najslabšem primeru potrebujemo še dve vprašanji (glej primer v prejšnjem odstavku), da uganemo domino. Torej skupaj potrebujemo 4 vprašanja.

Če ostane 7 domin, lahko z naslednjim vprašanjem razdelimo skupino domin na dva dela, to je po 4 in 3 domine. Za štiri domine potrebujemo še dve dodatni vprašanji: prvo razdeli na dve skupini po 2 domini, drugo pa določi iskano domino izmed preostalih dveh domin. Torej v tem primeru potrebujemo skupaj 4 vprašanja. Tudi v primeru, če nam ostanejo 3 domine, potrebujemo še dve dodatni vprašanji, torej skupaj 4 (glej prejšnje primere).

Poglejmo še primer, če nam ostane 8 domin. Tu spet uporabimo bisekcijo (razpolavljanje) in skupino 8 domin z dodatnimi vprašanji oklestimo na 4 domine, nato na 2 domini in z zadnjim

vprašanjem ugotovimo iskano domino. Torej tudi v tem primeru potrebujemo skupaj 4 vprašanja v najslabšem primeru.

V splošnem torej velja, da pri vseh štirih podanih vprašanjih potrebujemo v najslabšem primeru še tri dodatna vprašanja, da lahko določimo iskano domino. Torej je vseeno, katero od navedenih vprašanj zastavimo kot prvo.

Verjetno se boste vprašali, zakaj najboljše prvo vprašanje ni tisto, ki nam domine razdeli na dva enaka dela. Odgovor se skriva v dejstvu, da imamo na mizi 10 domin. Število 10 ni potenca števila 2. Če bi imeli na mizi 8 domin ($8 = 2^3$), bi z razpolavljanjem (metodo bisekcije) lahko uganili domino s 3 vprašanji. Če postavimo 4 vprašanja, lahko z metodo bisekcije uganemo domino izmed 16 domin ($16 = 2^4$). Ker pa imamo 10 domin, v najslabšem primeru ne zadostujejo 3 vprašanja, ampak potrebujemo 4. In če prvo vprašanje razdeli naših 10 domin na taka dva dela, da noben del ni večji od 8 (tako kot vsa štiri ponujena vprašanja pri odgovorih), potem potrebujemo v najslabšem primeru še 3 dodatna vprašanja, da uganemo domino. Ker imamo na začetku 10 domin, ni potrebno, da pri prvem vprašanju domine razdelimo enakomerno na dva dela. Le pri vprašanju, ki bi domine razdelilo na dela velikosti 9 in 1, ne bi zadostovala štiri vprašanja, ampak bi jih v najslabšem primeru morali postaviti pet.

Računalniško ozadje

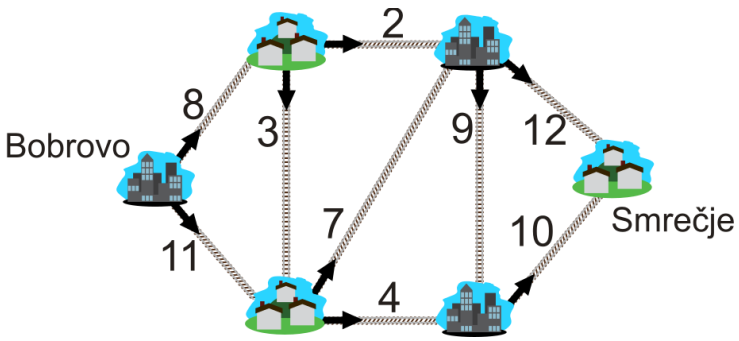
Računalniki hranijo veliko podatkov in potrebujejo hitre načine za iskanje določenih podatkov. Če so podatki urejeni, lahko za iskanje podatkov uporabimo na primer metodo *binarno iskanje*. Deluje takole: začnemo s preverjanjem srednjega elementa. Če je tisto, kar iščemo, večje ali manjše, lahko zanemarimo polovico podatkov in se osredotočimo le na preostale. To počnemo še naprej in vsakič zmanjšamo količino podatkov na polovico. Če imamo torej 1 milijon elementov, lahko tisto, kar iščemo, najdemo v samo 20 korakih!

V tej nalogi imamo samo vprašanja z odgovorom da/ne, zato je smiselno domino poiskati tako, da z vsakim vprašanjem množico potencialnih rešitev razdelimo na dva dela (podobno kot koraki pri binarnem iskanju). Tak algoritem imenujemo *bisekcija* ali *razpolavljanje*.

Železniško omrežje

8. in 9. razred, srednja šola

V Bobrovini imajo razvejano železniško omrežje, a lahko po vsaki progi med dvema krajema pelje na dan le omejeno število vlakov (številka ob progi na spodnjem diagramu). Vlak lahko pelje po progi le v smeri, ki jo nakazuje puščica, hkrati pa je lahko na progi več vlakov (če ne presežejo podane omejitve).



Z vlaki morajo prepeljati blago iz Bobrovega v Smrečje. Največ koliko vlakov lahko dnevno odpelje iz Bobrovega in prispe v Smrečje?

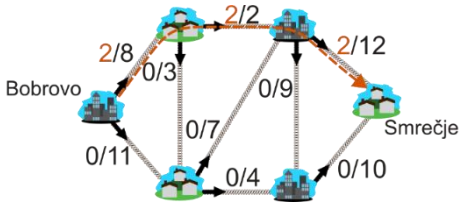
Rešitev

Iz Bobrovega v Smrečje lahko vsak dan odpelje največ 13 vlakov.

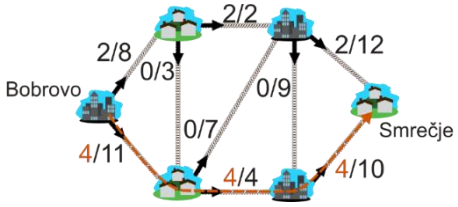
Najprej ugotovimo, da iz Bobrovega v enem dnevu ne more odpeljati več kot 8 + 11 = 19 vlakov.

Nalogo lahko rešimo tako, da poiščemo najkrajšo možno pot od Bobrovega do Smrečja in ugotovimo največje možno število vlakov na tej poti. Nato postopek ponovimo še na preostanku omrežja.

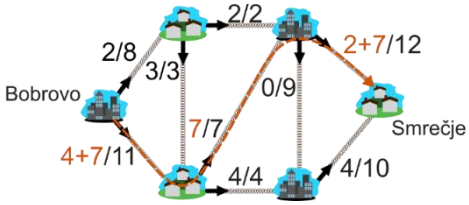
Spodnje slike prikazujejo ta postopek. Najkrajša pot od Bobrovega do Smrečja je dolžine tri. Po prvi poti lahko peljeta le dva vlaka na dan.



Tudi druga pot ima dolžino tri in po njej lahko peljejo štirje vlaki.



Tretja pot pa omogoča prehod 7 vlakom.



Našli smo torej tri poti, njihova skupna zmogljivost pa je 2 + 4 + 7 =13 vlakov. Seveda lahko 13 vlakov od Bobrovega do Smrečja pelje tudi po drugih, morda tudi daljših, poteh.

Preverimo še, da je 13 res največje možno število vlakov. Na sredini omrežja so tri proge, ki imajo kapacitete 2, 7 in 4. Ker gredo vse poti od Bobrovega do Smrečja preko ene od teh treh prog, največje število vlakov ne more biti večje od 2 + 7 + 4 = 13.

Računalniško ozadje

V nalogi smo reševali problem največjih pretokov. Za rešitev takega problema obstaja več algoritmov, kot so Ford-Fulkersonov, Edmonds-Karpov, Dinitzov algoritem in še mnogi drugi. Ti algoritmi se uporabljajo pri optimizaciji prometnega toka, v distribucijskih sistemih (npr. vode ali elektrike), za dodeljevanje virov, računalniška omrežja in podobno.

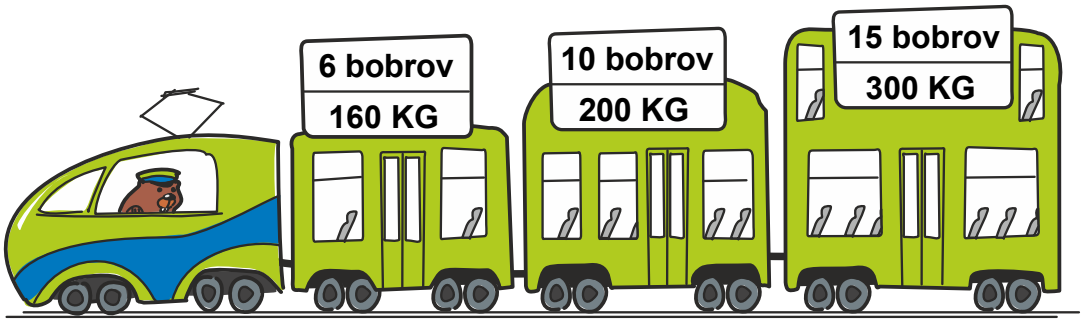
Izlet z vlakom

srednja šola

Osem družin bi rado šlo z vlakom na izlet. Vlak ima v vsakem vagonu omejeno število sedežev in lahko v njem prevaža omejeno količino prtljage. Podatki o družinah so predstavljeni v naslednji tabeli:

Družina	**Število članov**	**Teža prtljage [kg]**
Avsec	3	50
Božič	4	80
Cankar	5	110
Dolenc	4	80
Erjavec	2	40
Furlan	3	70
Gregorič	6	130
Hrovat	5	100

Železnice imajo na voljo naslednjo vlakovno kompozicijo – na sliki je na vsakem vagonu napisano število sedežev in koliko prtljage lahko peljejo v posameznem vagonu:



Koliko družin gre lahko na izlet, če:

- morajo vsi člani iste družine potovati v istem vagonu,
- mora vsak član sedeti na svojem sedežu,
- mora biti prtljaga v istem vagonu kot njeni lastniki in
- mora teža prtljage ustrezati največji dovoljeni teži prtljage v vsakem vagonu?

Rešitev

Za lažjo razlago rešitve označimo družine s prvimi črkami njihovih priimkov.

V vseh družinah skupaj je 32 članov, na vlaku pa je le 31 sedežev, zato zagotovo ne more potovati vseh 8 družin.

Preveriti moramo, ali je možno, da naenkrat potuje 7 družin. Ugotovimo lahko, da je to mogoče. Eden od možnih načinov je naslednji:

- V 1. vagonu potuje družina G, v kateri je 6 članov in imajo 130 kg prtljage.
- V 2. vagonu potujejo družine A, C in E, ki imajo skupaj 10 članov in 200 kg prtljage.
- V 3. vagonu potujejo družine B, D in H, ki imajo skupaj 13 članov in 260 kg prtljage.

Pravilen odgovor je zato 7 družin.

Računalniško ozadje

Naloga se ukvarja z optimizacijo in primerno izbiro, glede na dane omejitve. Izhaja iz teorije kompleksnosti in kriptografije. V širšem kontekstu je tak tip naloge znan kot *problem nahrbtnika* (angl. *knapsack problem*), za katerega vemo, da ne obstaja primeren algoritem, ki bi dani problem rešil v nekem zglednem času.

Napaka v programu

srednja šola

Robot risar pozna dva osnovna ukaza za premik:

N pomeni, da se premakne naprej za 1 enoto in

D pomeni, da se obrne v desno za 90 stopinj.

Poleg tega robot omogoča, da novo (še neuporabljeno) črko določimo kot bližnjico za bolj zapleteno zaporedje premikov.

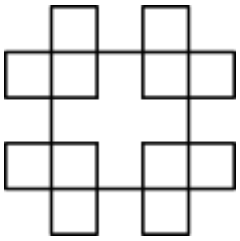
Na primer:

A = **NNN** je bližnjica za premik naprej za 3 enote.

B = **ADA** je bližnjica za uporabo **A** (**NNN**), nato obrat v desno in ponovno uporabo **A**.

Program lahko vsebuje poljubno število vrstic kode za definiranje različnih bližnjic. Zadnja vrstica programa pa mora vsebovati ukaz `Nariši`, ki ustvari končno risbo.

Jani želi narisati naslednjo sliko, kjer je vsak kratek odsek črte dolžine 1 enote:



Jani je napisal naslednjih pet vrstic kode:

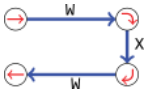
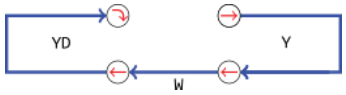
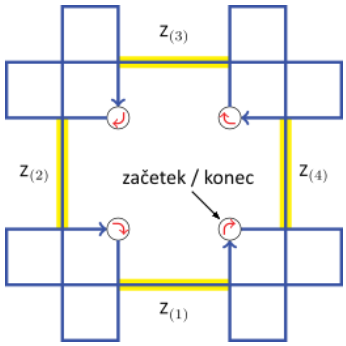
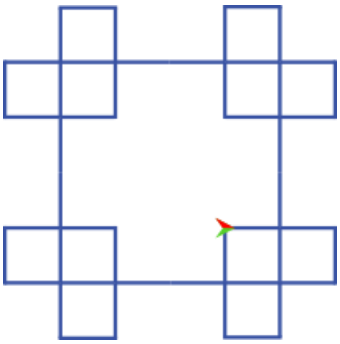
```
Vrstica 1: W = NN
Vrstica 2: X = DND
Vrstica 3: Y = WXW
Vrstica 4: Z = YWYD
Vrstica 5: Nariši ZZZZ
```

Vendar robot ne ustvari pravilne slike, saj je v programu tipkarska napaka. Natanko ena vrstica kode vsebuje natanko eno napačno črko. Katero vrstico kode mora popraviti, da bo program izrisal želeno sliko?

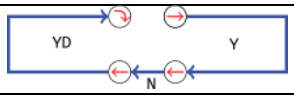
Rešitev

Spremeniti mora vrstico 4.

Recimo, da je robot na začetku obrnjen proti desni. V spodnji tabeli so v stolpcu »Akcija« prikazani premiki, in sicer modra puščica predstavlja premik naprej, krogec pa morebitni obrat na desno (ukrivljena puščica). V stolpcu »Učinek« pa je za vsak ukaz prikazan končni rezultat akcije, kjer sta začetni pozicija in orientacija robota označeni z zeleno puščico, končni pa z rdečo.

Ukaz	Akcija	Učinek
`Vrstica 1: W = NN`		
`Vrstica 2: X = DND`		
`Vrstica 3: Y = WXW`		
`Vrstica 4: Z = YWYD`		
`Vrstica 5: Nariši ZZZZ`		

Odseki pri črki **Z** (označeni rumeno) so dolgi 2 enoti, na želeni sliki pa so ti segmenti dolgi le 1 enoto. Torej moramo popraviti vrstico 4 tako, da namesto ukaza **W** (ki je **NN**) uporabimo ukaz **N** (premik za eno samo enoto):

`Vrstica 4: Z = YNYD`		

Morda bi kdo pomislil, da bi lahko kodo enostavno popravil tako, da popravi prvo vrstico v **W** = **N**. Vendar taka rešitev ni pravilna, saj tak popravek vpliva tudi na ukaz **Y** (vrstica 3), ki uporablja ukaz **W**, in posledično tudi za ukaz **Z** (vrstica 4). Zato bi s takim popravkom narisali drugačno sliko.

Računalniško ozadje

V nalogi smo uporabili programski jezik, ki ukaze izvaja zaporedno po vrsti, a omogoča tudi definiranje novih procedur kot skupino ukazov. Procedure in funkcije se v programu pogosto uporabljajo, saj z njimi dobimo boljšo strukturo programa in posledično lažje berljiv (in razumljiv) program. Vendar pa slednje ne velja za program v naši nalogi: s poimenovanjem procedur smo dobili program, ki je bolj nejasen in zato težje razumljiv.

Opečnat zid

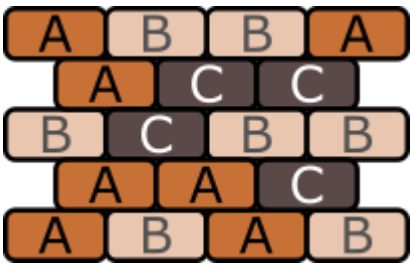
srednja šola

Bojan je zgradil zid iz treh vrst opek: A, B in C (slika na desni). Potem pa je ugotovil, da zid stoji na napačnem mestu, in ga mora prestaviti. Zid prestavi opeko za opeko na naslednji način:

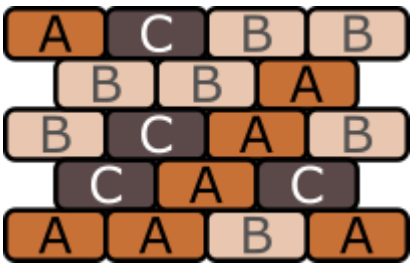
- Iz prvotnega zidu odstrani katero koli opeko, ki neposredno nad sabo nima nobene opeke.
- Odstranjeno opeko postavi neposredno v novi zid, bodisi na tla bodisi na drugo opeko, vendar ne pod katero koli obstoječo opeko v novem zidu.

Bojanu je vseeno, v kakšnem vrstnem redu prestavi različne vrste opek, da le sledi zgornjima dvema praviloma. Kateri zid **ne** more biti Bojanov novi zid?

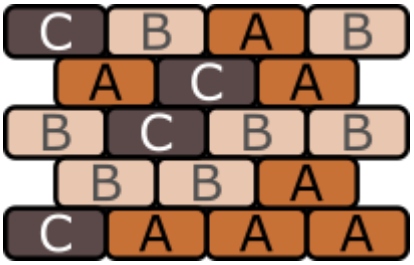
A)



B)



C)



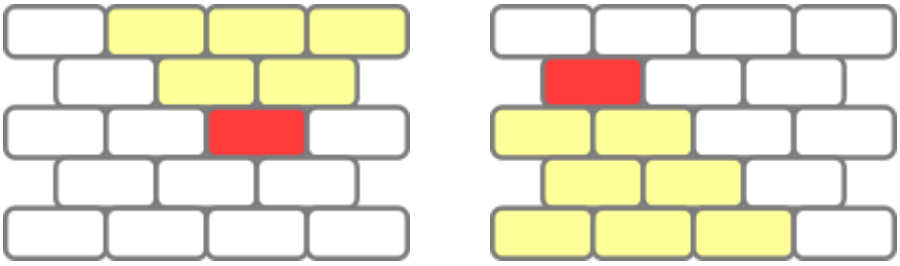
D)

Rešitev

Pravilen odgovor je B.

Bojan lahko zgradi zidove A, C in D, ne more pa zgraditi zidu B.

Pri iskanju rešitve te naloge najprej razmislimo o obeh pravilih, ki se ju mora držati Bojan pri prestavljanju zidu. Če pogledamo rdečo opeko na spodnji levi sliki prvotnega zidu, vidimo, da moramo najprej iz zidu odstraniti vse rumene opeke v trikotniku nad njo, preden lahko odstranimo rdečo opeko. Podobno velja za rdečo opeko na desni sliki novega zidu: preden lahko v zid postavimo rdečo opeko, moramo postaviti vse rumene opeke v trikotniku pod njo (na tej sliki del trikotnika manjka, saj je tam rob zidu).

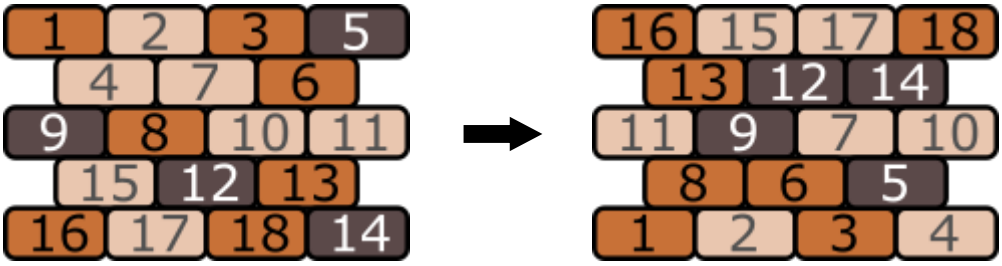


Z upoštevanjem te ugotovitve lahko preverimo, katere zidove lahko zgradimo. Na spodnjih slikah so opeke oštevilčene po vrsti, kot jih jemljemo iz prvotnega zidu (na levi sliki) in jih dodajamo v novi zid (na desni sliki). Upoštevanje obeh pravil lahko enostavno preverimo, saj mora veljati naslednje:

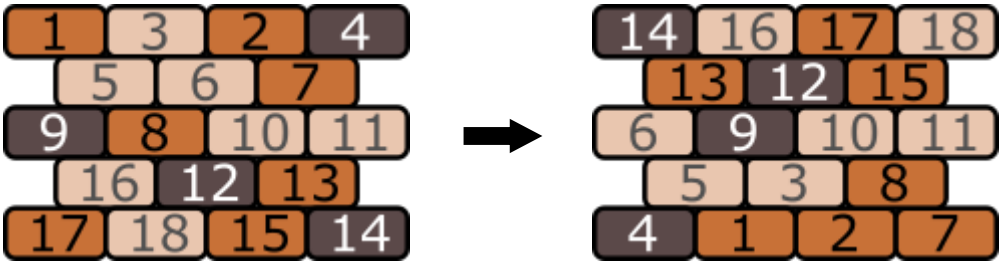
1. Za vsako opeko v prvotnem zidu (na levi) mora veljati, da imajo vse opeke v trikotniku nad njo manjše številke (to pomeni, da smo najprej odstranili vse opeke nad to opeko).
2. Za vsako opeko v novem zidu (na desni) mora veljati, da imajo vse opeke v trikotniku pod njo manjše številke (to pomeni, da smo najprej dodali vse opeke pod to opeko).

Za zidove pod A, C in D lahko najdemo rešitev (prikazana je le ena izmed možnih rešitev).

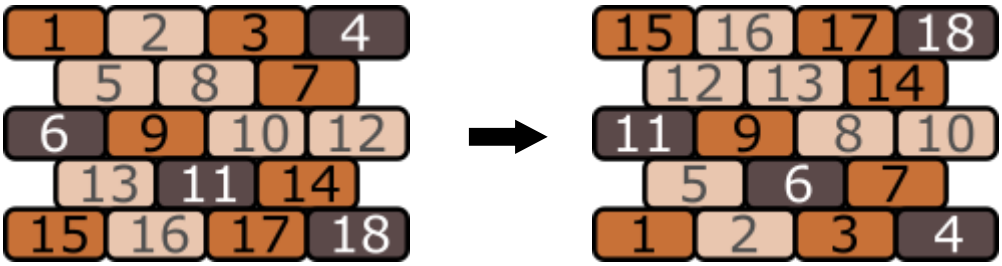
Gradnja zidu A:



Gradnja zidu C:



Gradnja zidu D:



Razmislimo še o zidu pri odgovoru B). Tik preden četrtič odstranimo opeko vrste B iz prvotnega zidu (to opeko bomo poimenovali četrta B-opeka), nobena opeka v spodnjih dveh vrstah ni na voljo. Torej smo do tega trenutka lahko premaknili največ 2 C-opeki, in sicer tisti iz zgornjih treh vrst. Sedaj odstranimo četrto B-opeko in jo postavimo v novi zid (zid pri odgovoru B). Da to opeko lahko postavimo v zid (opeka lahko pride le v drugo vrstico od zgoraj), bi morali pred tem v novi zid že prestaviti vsaj 3 C-opeke, kar pa nasprotuje naši prejšnji ugotovitvi. Torej zidu B ni mogoče zgraditi.

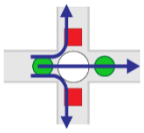
Računalniško ozadje

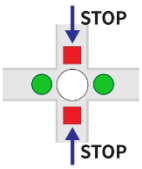
Naloga se nanaša na analizo odvisnosti med objekti, kar je klasičen problem v računalništvu. Odvisnosti med objekti (v naši nalogi so to opeke) so običajno izražene v obliki grafa odvisnosti. Veljaven vrstni red obdelave (v naši nalogi je to vrstni red, v katerem je mogoče opeke odstraniti in postaviti) ustreza topološkemu razvrščanju objektov.

Semaforji

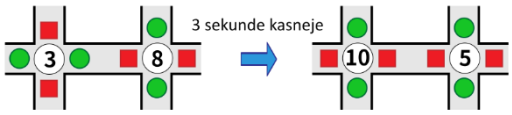
srednja šola

V Bobrovem imajo na vsakem križišču semafor, ki usmerja promet po naslednjih pravilih:



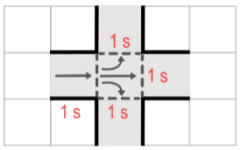


Ko sveti zelena luč, lahko promet teče naravnost ali pa zavije levo ali desno.
Ko zasveti rdeča luč, se mora promet ustaviti in počakati na zeleno luč.

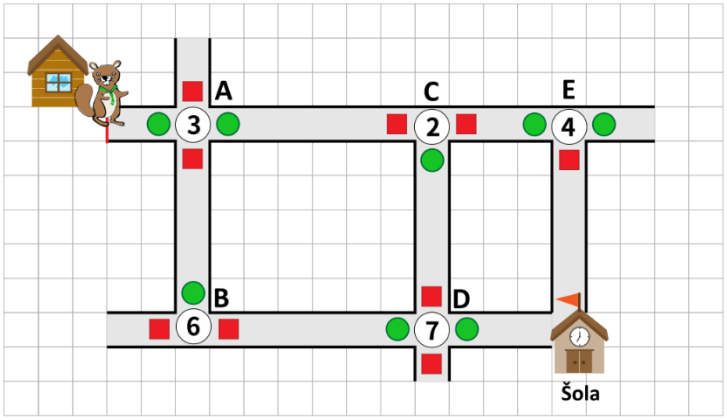


Semafor izmenično preklaplja luči med rdečo in zeleno vsakih 10 sekund. Vsak semafor ima časovnik, ki odšteva od 10 do 0. Ko doseže 0, semafor preklopi luči in ponovno začne odštevati. Trenutno stanje časovnika je prikazano na vsakem semaforju.

Bobrček Boris potrebuje eno sekundo, da prehodi en kvadratek poti. Prav tako potrebuje eno sekundo, da prečka cesto ali da v križišču zavije levo ali desno.



Boris se odpravi v šolo. Spodnji zemljevid prikazuje ceste do šole in stanje časovnikov na semaforjih v trenutku, ko Boris starta z rdeče črte pred svojim domom:



Boris želi priti do šole, ne da bi se moral ustaviti na rdeči luči. Po kateri poti naj gre?

Rešitev

Boris mora po poti A → C → D → ŠOLA, da se izogne čakanju na rdeči luči.

Do rešitve pridemo tako, da za posamezen semafor ugotovimo, ali na njem sveti rdeča ali zelena luč, ko Boris pride do tega semaforja. Tako na primer na začetku časovnik na križišču A kaže 3 sekunde do rdeče luči. Boris do križišča A potrebuje 2 sekundi (prehodi 2 kvadratka). Torej še lahko ujame zeleno luč in lahko prečka cesto ali pa zavije.

Do križišča B potrebuje Boris 8 sekund. Semafor na tem križišču je na začetku zelen, a se po 6 sekundah preklopi na rdečega. Torej je na semaforju rdeča luč, ko do njega prispe Boris. Zato pot od križišča A do B ni prava, saj bi moral na križišču B čakati na rdeči luči celih 8 sekund.

Do križišča C potrebuje Boris 9 sekund. Na začetku na tem križišču sveti rdeča luč, a se po 2 sekundah spremeni v zeleno, ki sveti naslednjih 10 sekund. Torej Boris pride na križišče na zeleno luč. Njegova pot mora biti A → C.

Poglejmo naprej križišče E. Do njega potrebuje Boris 13 sekund. Na začetku sveti zelena luč, a se po 4 sekundah preklopi na rdečo, ki sveti še naslednjih 10 sekund. Ko pride Boris do Križišča E, na njem še vedno sveti rdeča luč in mora tako na zeleno počakati še 1 sekundo. Torej pot skozi križišče E ni prava.

Preverimo še križišče D. Do njega potrebuje Boris 15 sekund. Na začetku na tem križišču sveti rdeča luč, ki se po 7 sekundah spremeni v zeleno. Zelena nato sveti še 10 sekund. V tem času lahko do križišča pride tudi Boris in tako ujame zeleno luč. Torej je edina možna pot do šole, da se izogne rdeči luči, preko križišč A, C in D: A → C → D → ŠOLA.

Računalniško ozadje

Planiranje je pomembna naloga tudi v našem vsakdanjem življenju, saj omogoča, da naloge izvedemo bolj učinkovito. Včasih moramo splanirati najboljšo pot (kot v tej nalogi), drugič pa morda stvari, ki jih shranimo v nahrbtnik in vzamemo na izlet. Pri reševanju takih nalog planiranja si lahko pomagamo z dinamičnim programiranjem.

Skrivanje slike

srednja šola

Slika v računalniku je pravzaprav pravokotnik, razdeljen na vrstice in stolpce kvadratnih celic – pikslov, kjer vsak piksel hrani podatek o njegovi barvi.

Leo se je spomnil novega načina za šifriranje slik z uporabo **V** (vodoravnih) in **N** (navpičnih) operacij.

Pri vsaki uporabi operacije **V** naredi naslednje:

- Vsak piksel v 1. vrstici ostane na svojem mestu (se ne premakne).
- Vsak piksel v 2. vrstici se premakne za 1 mesto v desno.
- Vsak piksel v 3. vrstici se premakne za 2 mesti v desno.
- …
- Vsak piksel v *n*-ti vrstici se premakne za $n - 1$ mest v desno.

Pri tem velja, da vsi piksli, ki se premaknejo preko desnega roba slike, kot skupina v istem vrstnem redu zavzamejo prazen prostor na levi strani vrstice (ciklični premik, glej spodnji primer).

Podobno deluje tudi operacija **N**:

- Vsak piksel v *n*-tem stolpcu se premakne za $n - 1$ mest navzdol, piksli, ki se premaknejo preko spodnjega roba slike, pa se prestavijo na vrh.

Primer operacij **V** in **N** na sliki velikosti 3 x 3, kjer so piksli označeni s števili od 1 do 9:

1	2	3
4	5	6
7	8	9

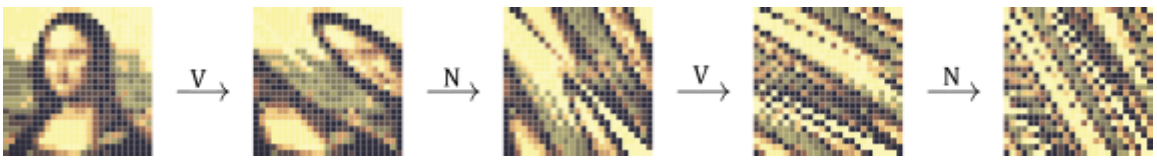
$\xrightarrow{V}$

1	2	3
6	4	5
8	9	7

$\xrightarrow{N}$

1	9	5
6	2	7
8	4	3

Sliko Mone Lize velikosti 25 x 25 bi z zaporedjem **VNVN** zašifrirali takole:



Leo zašifrira spodnjo dvobarvno sliko velikosti 1000 x 1000 pikslov tako, da najprej uporabi operacijo **N** in nato še operacijo **V**.

Katera slika najbolje prikazuje dobljeni rezultat?

A)  B)  C)  D) 

E)  F)  G)  H) 

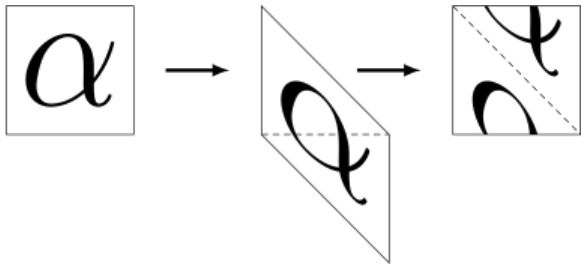
Rešitev

Pravilen odgovor je E.

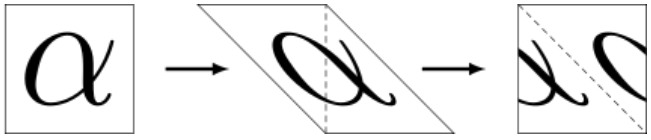
Slika ima visoko resolucijo, zato lahko premike posameznih pikslov dobro aproksimiramo z geometričnimi transformacijami likov na sliki.

Na primeru 3 x 3 slike vidimo, da pri operaciji **V** postane prvi stolpec (1, 4, 7) diagonala od zgoraj levo proti spodaj desno, medtem ko diagonala od zgoraj desno proti spodaj levo (3, 5, 7) postane zadnji stolpec. To nakazuje, da gre za strižno deformacijo, kar jasno kaže tudi prva slika v primeru Mone Lize. Geometrijske učinke obeh operacij, **N** in **V**, bomo analizirali podrobneje, začenši z **N**, saj je uporabljena prva.

Učinek operacije **N** lahko razdelimo na dva dela. Najprej pravokotnik strižemo navpično proti spodnjemu desnemu oglišču, da dobimo paralelogram z izkrivljeno sliko. Nato spodnji previsni trikotnik premaknemo navzgor, da dobimo novo pravokotno sliko prvotne velikosti:

N : 

Podobno je učinek operacije **V** vodoravni strig, ki mu sledi delni vodoravni premik:

V : 

Z uporabo teh korakov na dani sliki dobimo naslednje zaporedje:

Rezultat je najbolj podoben sliki E.

Računalniško ozadje

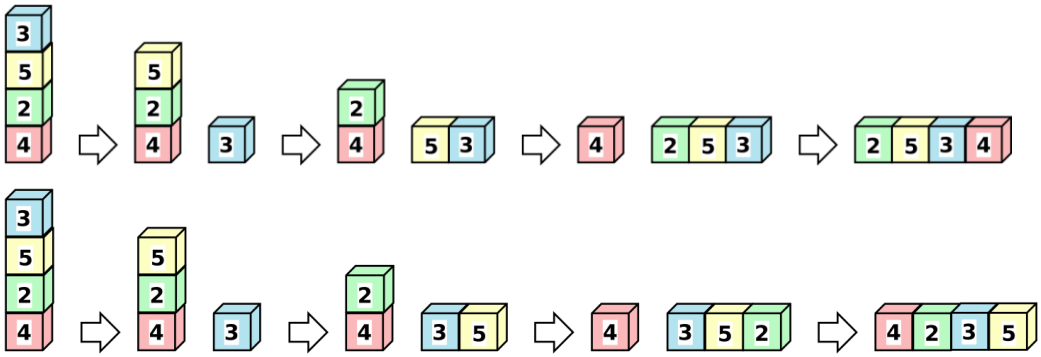
Pri prenosu podatkov moramo le-te šifrirati, če jih želimo skriti pred nepooblaščenimi pogledi morebitnih vsiljivcev, ki bi te podatke lahko prestregli. V tej nalogi je bil podan šifrirni algoritem, ki je preprost in enostaven tudi za dešifriranje, vendar kljub temu ustvari novo sliko, ki je zelo drugačna od izvirnika (in na kateri le stežka prepoznamo izvirno sliko). Pravzaprav bi šifrirani sliki lahko pripisali zelo veliko možnih izvornih slik, če ne poznamo točnega šifrirnega mehanizma. Za tako šifriranje pa ne potrebujemo nobenega skrivnega ključa – vse, kar moramo vedeti, je le šifrirni algoritem in s tem lahko dešifriramo katero koli sliko, šifrirano na ta način.

Števila

srednja šola

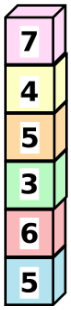
Olga se igra s kockami. Na vsaki kocki je zapisana ena števka. Olga iz kock sestavi visok stolp in nato z vrha stolpa jemlje kocke po vrsti eno po eno ter iz njih sestavi število. Vsakič, ko s stolpa vzame kocko, jo lahko postavi ali na levo ali na desno stran števila, ki ga sestavlja.

Spodnje slike prikazujejo stolp iz štirih kock in dve števili, ki ju lahko sestavimo (2534 in 4235):



Seveda bi iz tega stolpa lahko sestavili tudi druga števila (npr. 3524).

Olga je zgradila še višji stolp, iz šestih kock (na desni sliki). Katero je najmanjše število, ki ga lahko sestavi ob upoštevanju podanih pravil?



Rešitev

Najmanjše število, ki ga lahko ob upoštevanju podanih pravil sestavi iz kock v stolpu, je 347565.

Najmanjše število se mora začeti na najmanjšo števko, ki je v stolpu, torej s 3. Vse števke, ki so v stolpu pod kocko s števko 3, moramo dodati na desno stran števila, ki ga sestavljamo. Torej se mora naše število končati na 65. Nad kocko s števko 3 imamo manjši stolp 547, iz katerega moramo sestaviti najmanjše število in ga vstaviti med 3 in 65. Tudi za ta manjši stolp sklepamo podobno: število se mora začeti na najmanjšo števko v stolpu, to je 4, na desno stran pa moramo dodati 5 – dobimo število 475. Končno število iz šestih števk je 3, ki ji sledi 475 in na koncu še 65, torej 347565.

Računalniško ozadje

Iskanje najboljše rešitve izmed vseh možnih rešitev (kot je na primer najmanjše število v naši nalogi) je pogost računalniški problem – imenuje se optimizacija. Rešitev lahko poiščemo tako, da preverimo vse možne rešitve. A je to v praksi precej potratno (v času in potrebnem

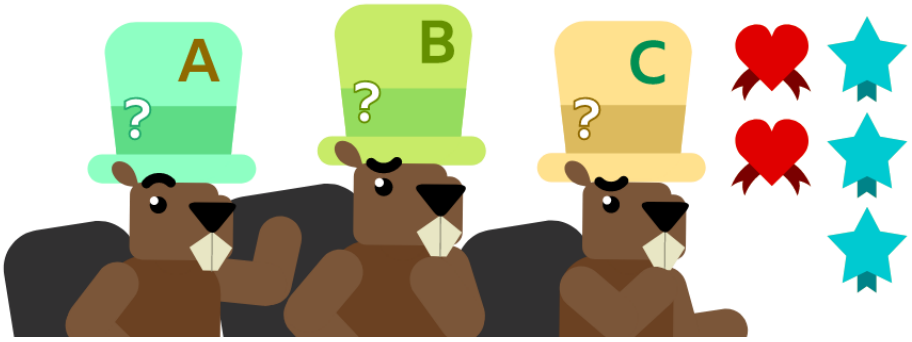
delu), zato pogosto uporabimo dodatne tehnike, s pomočjo katerih zmanjšamo množico možnih rešitev in tako pohitrimo iskanje optimalne rešitve. V tej nalogi smo uporabili *požrešno strategijo* (angl. *greedy strategy*), da smo določili najbolj levo števko, in pristop *deli in vladaj* (angl. *divide-and-conquer*), da smo poiskali rešitev še za stolp kock nad kocko z najmanjšo števko (enak problem, le z manj kockami).

Trakovi

srednja šola

Bobri imajo pet klobukov. Dva klobuka imata trak s srcem in trije klobuki trak z zvezdo.

Trije bobri stojijo v vrsti in vsak od njih dobi en klobuk. Vsak lahko vidi, kakšen trak je na klobuku vseh bobrov pred njim, ne vidi pa traku na svojem klobuku ali na klobuku katerega koli bobra za njim.



Bober A pravi, da ne more ugotoviti, kakšen trak je na njegovem klobuku.

Bober B odgovori, da tudi on ne more ugotoviti, kakšen trak je na njegovem klobuku.

Kakšen trak je na klobuku bobra C?

A) Trak z zvezdo
B) Trak s srcem
C) Ni mogoče določiti

Rešitev

Le na dveh klobukih je lahko trak s srcem. Če bi bil tako na klobuku bobra B in bobra C trak s srcem, bi bober A vedel, da je na njegovem klobuku zagotovo trak z zvezdo, saj vidi oba klobuka pred njim. Iz tega vemo, da je trak z zvezdo vsaj na enem od klobukov bobra B ali bobra C.

Če bi bil trak s srcem na klobuku bobra C, bi lahko bober B, ki ta klobuk vidi, iz izjave bobra A izvedel, da ima on na klobuku trak z zvezdo. Vendar iz izjave bobra B izvemo, da ne more ugotoviti, kakšen trak ima na klobuku. Iz tega sklepamo, da je na klobuku bobra C trak z zvezdo.

Računalniško ozadje

Naloga od nas zahteva logično sklepanje. Logika igra ključno vlogo na številnih področjih računalništva: podatkovne baze, programski jeziki, umetna inteligenca, načrtovanje in

preverjanje strojene in programske opreme računalnika … Logika je eden od temeljnih konceptov, na katerih temelji računalniško mišljenje.

Pregled nalog

Državno tekmovanje

		6.	7.	8.	9.	SŠ
Igra simbolov	Urugvaj	●	●			
Kocke	Avstralija	●	●			
Smučarski Raj	Poljska	●	●			
Ustvarjanje vzorca	Irska	●	●			
Iskanje zaklada	Madžarska	●	●	●	●	
Labirint	Tajvan	●	●	●	●	
Ladje	Črna Gora	●	●	●	●	
Pot za robota	Poljska	●	●	●	●	
Vračanje knjig	Slovenija	●	●	●	●	
Bomboni	Japonska	●	●	●	●	●
Električno kolo	Tajvan	●	●	●	●	●
Iskanje avta	Kitajska	●	●	●	●	●
LingoLearner	Avstralija	●	●	●	●	●
Najvišji rezultat	Slovaška	●	●	●	●	●
Stroj za zapis nizov	Latvija	●	●	●	●	●
Skrivna sporočila	Severna Makedonija			●	●	
Kamen, škarje, papir	ZDA			●	●	●
Lizine pesmice	Švica			●	●	●
Ugani domino	Portugalska			●	●	●
Železniško omrežje	Pakistan			●	●	●
Izlet z vlakom	Slovenija					●
Napaka v programu	Avstralija					●
Opečnat zid	Finska					●
Semaforji	Tajvan					●
Skrivanje slike	Avstralija					●

		6.	7.	8.	9.	SŠ
Števila	Portugalska 					●
Trakovi	ZDA 					●

Avtorji nalog

Milla Aavakare, Finska

Esraa Almajhad, Saudova Arabija

Masiar Babazadeh, Švica

Leonardo Barichello, Brazilija

Javier Bilbao, Španija

Anne-Marie Bøgelund Kristiansen, Danska

Zainab Bohulaigh, Saudova Arabija

Eugenio Bravo, Španija

Lucia Budinská, Slovaška

Marta J. Burzanska, Poljska

Leonard Busuttil, Malta

Maria Cepeda, Mehika

Špela Cerar, Slovenija

Marios Omar Choudary, Pakistan

Lucía Crivelli, Urugvaj

María Eugenia Curi, Urugvaj

Andrew Csizmadia, Združeno kraljestvo

Mária Čujdíková, Slovaška

Susanne Datzko, Švica

Justina Dauksaite, Nizozemska

Zhang Dongshuang, Kitajska

Nora A. Escherle, Švica

Gerald Futschek, Avstrija

Bence Gaál, Madžarska

Simon Garrad, Irska

Anaclara Gerosa, Urugvaj

Maria Ines Gomez, Kostarika

Adam Grodeck, Avstralija

Ya-Chun Hsu, Tajvan

Heikki Hyyrö, Finska

Yukio Idosaka, Japonska

Thomas Ioannou, Ciper

Mile Jovanov, Severna Makedonija

Ungyeol Jung, Južna Koreja

Heidi Kaarto, Finska

Filiz Kalelioğlu, Turčija

Alenka Kavčič, Slovenija

Vaidotas Kinčius, Litva

Dongyoon Kim, Južna Koreja

Injoo Kim, Južna Koreja

Vaidotas Kinčius, Litva

Jia-ling Koh, Tajvan

Sophie Koh, Singapur

Victor Koleszar, Urugvaj

Dennis Komm, Švica

Omran Kouba, Sirija

Greg Lee, Tajvan

Taina Lehtimäki, Irska

Marielle Léonard, Francija

HsinTing Lin, Tajvan

Judith Lin, Tajvan

Ke-Chun Lin, Tajvan

Nadia Martignone, Urugvaj

Yoshiaki Matsuzawa, Japonska

Anja Mejač, Slovenija

Jelena Milojković, Črna Gora

Kwangsik Moon, Južna Koreja

Madhavan Mukund, Indija

Trajche Najdovski, Severna Makedonija

Tom Naughton, Irska

Martins Opmanis, Latvija

Graciela Oyhenard, Urugvaj

Özgür Özdemir, Turčija

Dejan Ozebek, Slovenija

Alex Parry, Združeno kraljestvo

Marika Parviainen, Finska

Praphan Pavarangkoon, Tajska

Jean-Philippe Pellet, Švica

Margot Phillipps, Nova Zelandija

Romina Pons, Urugvaj

John-Paul Pretti, Kanada

Jelena Radičević, Črna Gora

Pedro Ribeiro, Portugalska

Maria del Rosario Schunk, Urugvaj

Nežka Rugelj, Slovenija

Eljakim Schrijvers, ZDA

Humberto Sermeño, Salvador

Vipul Shah, Indija

Maiko Shimabuku, Japonska

Răzvan-Alexandru Smădu, Pakistan

Emil Stankov, Severna Makedonija

Alieke Stijf, Nizozemska

Nikolaos Stratis, Ciper

Maciej M. Sysło, Poljska

Goran Šuković, Črna Gora

Seiichi Tani, Japonska

Ezra Templonuevo, Filipini

Monika Tomcsányiová, Slovaška

Meng-ting Tsai, Tajvan

Christine Vender, Kanada

Michael Weigend, Nemčija

Chris Wetherell, Avstralija

Kyra Willekes, Nizozemska

Jiangang Yao, Hong Kong

Hongjin Yeh, Južna Koreja

Yi Shan Yeh, Tajvan

Vaiva Zukauskaite, Litva