### Navodila

Učni cilji so lahko združevani v učne enote, vendar to združevanje ni obvezno. Tudi zaporedje usvajanja učnih ciljev ni predpisano in je prepuščeno ustvarjalcu učnega scenarija. Posebej bi poudarili, da Učni cilji po učnih enotah niso bili pisani kot zaporedje, ampak je predvideno, da učeči prilagodi vrstni red predznanju učečih se, okolju v katerem se program izvaja in ustreznim avtentičnim problemom.

Prav tako je predvideno število ur zgolj v obliki priporočila, saj se dejansko število ur razlikuje glede na konkretne učne scenarije.

Učni cilji so podani v tabeli in dodatno opremljeni z navedbo ***koncepta*** (***vsebine***) in navezavo na opis vsebine v *Okvirju računalništva in informatike od vrtca do srednje šole* (<https://redmine.lusy.fri.uni-lj.si/attachments/download/3060/Porocilo_RINOS_10_1_22.pdf>). Okvir razdeli RIN na pet področij:

1. Računalniški sistemi
2. Podatki in analiza
3. Algoritmi in programiranje
4. Omrežja in internet
5. Učinki računalništva in informatike

Vsako od področij se nato deli še na dve ali več podpodročij. Vsebina je razdeljena na starostna obdobja:

* OBDP: vrtec
* OBD1: osnovna šola, 1. triada
* OBD2: osnovna šola, 2. triada
* OBD3: osnovna šola, 3. triada
* OBD4: srednja šola.

Navezavo v tabeli zabeležimo kot: <*področje*>/<*podpodročje*>/<*starostno obdobje*>, kjer se *področje* navaja kot številka iz zgornjega spiska in *starostno obdobje* z oznako iz zgornjega seznama, medtem ko se *podpodročje* izpiše, na primer »***3/Algoritmi/OBD2***«. Med učnimi cilji se pojavljajo vsebine iz osnovne šole, ker jih v slovenskem šolskem sistemu ni in se lahko pojavijo šele v srednji šoli.

### Doseganje ciljev / preverjanje

Tako pri formativnem kot sumativnem preverjanju znanja je poseben poudarek na izdelavi takih artefaktov, kjer je možna izvedba z avtomatičnim agentom. Konkretno to pomeni, da naj bi učeči se bili sposobni ustvariti npr. program v programskem jeziku Python, s katerim bi demonstrirali svoje znanje in razumevanje koncepta zanke. Ali pa pokazati svoje poznavanje modularnosti tako, da v okolju Cubetto rešijo problem, ki predvideva izdelavo podprograma in klic le tega.

\*\*\*\* Kviz na 1ka

### Učni cilji po učnih enotah

V projekt so vključene različne srednje šole z različnimi programi. Prav tako so določeni učeči izrazili željo po večji prožnosti, s katero bi naslavljali različno predznanje, zanimanje in sposobnosti učečih se. Zato so učni cilji razdeljeni v **osnovni** in ***nadaljevalni*** del. Učne cilje in iz njih izhajajoča temeljna znanja osnovnega dela naj bi usvojili dijaki vseh srednješolskih programov. Kot je že bilo navedeno v uvodu, je zaporedje usvajanja učnih ciljev učiteljeva izbira. V pomoč mu je za vsak sklop navedeno predlagano število ur (stolpec PU).

V nekaterih programih (npr. vzgojitelj, tehnična srednja šola ipd.) bo predlagano število ur verjetno ravno pravšnje, medtem ko v je drugih programihj ocena previsoka. Prav tako je lahko v nekaterih programih del učnih ciljev pokrit že preko drugih programov (npr. v sklopu obveznega predmeta Informatika) ali preko predznanja učečih se. V tem primeru bodo učni cilji usvojeni prej kot v ocenjenem času.

Skupna količina porabljenega časa za osnovni del bo tako po programih različna - v nekaterih bo večja od predvidenih 18 ur in v nekaterih bo manjša ali celo bistveno manjša. Za preostanek časa dodajamo še učne cilje v nadaljevalnem delu, ki so razdeljeni na ***priporočene*** in ***dodatne***. Predlagamo, da se učitelji najprej odločijo za priporočene vsebin in, če jim še ostane čas, za dodatne. Ocena potrebnega časa za usvajanje ciljev nadaljevalnega dela je namenoma opuščena, ker je odvisna od konkretne učne situacije.

**Pomembno**: Ključno vlogo pri zaporedju usvajanja učnih ciljev, času za ta in naboru le-teh imajo učni scenariji. Ti bodo zelo verjetno različni glede na učni program in glede na konkretno dijaško populacijo.

#### Osnovni del

Da ponovimo, v osnovnem delu navedene učne cilje naj bi pokrivali učni scenariji v vseh različnih tipih srednjih šol in programov. Možno je, da je del učnih ciljev pokrit že preko drugih programov (npr. v sklopu obveznega predmeta Informatika) ali predznanja učečih se. Tu navedeni učni cilji ne predvidevanjo nobenega predznanja.

| **Koncept / vsebina** | **Učni cilji** | **PU** | **Okvir** |
| --- | --- | --- | --- |
| Pojem algoritma, več algoritmov za isti problem, različni načini zapisa algoritma. | Dijak razloži pojem algoritma.Dijak pokaže primer problema, kjer obstaja več algoritmov za reševanje. Dijak prikaže različne oblike zapisov algoritma. | 2 | 3 / Algoritmi / OBDP, OBD1, OBD2, OBD3 |
| Zaporedje ukazov, spremenljivke, branje/izpisovanjepodatkovni tip, pretvarjanje med tipi, nizi (ne kot sest. PS). | Dijak uporabi spremenljivko za predstavitev spreminjajočega se podatka.: - določi ustrezni tip spremenljivke oz. tip vsebine (na primer celo število, decimalno število, niz)- rokuje s spremenljivko (priredi in spremeni vrednost, prebere vrednost, uporabi jo v ustreznih operacijahSpreminja podatke enega tipa v drugega. (niz -> število, število -> niz, decimalno število -> celo število)Dijak s primerom pokaže pomembnost vrstnega reda ukazov.Dijak sledi izvajanju algoritma in napove njegov izid. | 2 | 3 / Spremenljivke / OBDP, OBD1, OBD2, OBD33 / Algoritmi / OBD3 |
| Vhod kot zaporedje podatkov, ki jih program pridobi iz okolja. Izhod kot zaporedje podatkov, ki jih program posreduje okolici. Programi, ki preberejo vhod, ga obdelajo in proizvedejo izhod. | Dijak sestavlja preproste programe, s katerimi prebere podatke, jih uporabi za novo informacijo in informacijo izpiše. | 2 | 3 / Spremenljivke / OBDP, OBD1, OBD2, OBD33 / Algoritmi / OBD3 |
| Funkcija (podprogram) kot poimenovan kos kode, posredovanje parametrov, vračanje vrednosti. | Sestavi program, kjer določen del kode izloči v podprogram/funkcijo in jo potem ustrezno kliče.Razloži razliko med izpisovanjem rezultata v funkciji/podprogramu in vračanjem rezultata. | 2 | 3 / Modularnost / OBDP, OBD1, OBD2, OBD3 |
| Pogojni stavek, »knjižnice«. | Ustvari program, ki se različno obnaša glede na vneseno vrednost podatka (npr. uporabi vejitev: če je podatek večji ali manjši od vrednosti 0, potem ...). Uporabi funkcije iz knjižnic. | 2 | 3 / Modularnost / OBD23 / Nadzor / OBD1, OBD2 |
| Naštevalne/števne zanke. | Uporabi zanko, kjer vnaprej vemo število ponovitev ukazov. | 2 | 3 / Nadzor / OOBD1, OBD2 |
| Zanke s pogojem. | Dijak prikaže primer problema, kjer ne vemo vnaprej, koliko ponovitev skupka ukazov bo potrebnih. | 2 | 3 / Nadzor / OBD1, OBD2, OBD3  |
| Načrtovanje zahtevnega programa – od problema v stvarnem svetu, preko modela, algoritma do kode. | Dijak razvije programe s postopkom, ki vključuje načrtovanje, zapisovanje programa, izvajanje, preizkušanje in odpravljanje neskladnosti z načrtom. | 2 | 3 / Razvoj prog. / OBDP, OBD1, OBD23 / Modularnost / OBD3, OBD4 |
| Načrtovanje zahtevnega programa – od problema v stvarnem svetu, preko modela, algoritma do kode. | Dijak pri načrtovanju definira problemske kriterije in omejitve ter funkcionalne specifikacije rešitve.Dijak program zapiše v skladu z načrtovanimi zahtevami. | 2 | 3 / Razvoj prog. / OBDP, OBD1, OBD23 / Modularnost / OBD3, OBD4 |

Po potrebi se število PU zmanjša/poveča.

####

#### Nadaljevalni del

Učni cilji nadaljevalnega dela nimajo navedene ocene potrebnega časa za njihovo usvajanje. V tabeli so zgolj označeni priporočeni učni cilji in dodatni cilji (glej stolpec “+”, “P” za priporočene in “D” za dodatne). Pri tem prvi predvsem nadgrajujejo koncepte osnovnega dela glede na podpodročja temeljnih znanj RINOS (glej stolpec “*Okvir*”).

| **Koncept / vsebina** | **Učni cilji** | **+** | **Okvir** |
| --- | --- | --- | --- |
| Tabela kot indeksirano zaporedje spremenljivk istega tipa. | Dijak pokaže in razloži nekaj primerov, kjer je smiselna organizacija podatkov v tabeliDijak reši problem, ki zahteva, da obdelamo vse podatke v tabeli in to razloži.  | P | 3 / **Spremenljivke** / OBD3, OBD4 |
| Gnezdene zanke/vejitveni stavki | Dijak prikaže primer problema, kjer je za rešitev problema potrebna zanka v zanki in gnezdeni vejitveni stavek. | P | 3 / **Nadzor** / OBD1, OBD2, OBD3  |
| Testiranje, dokumentiranje, primerjanje algoritmov (čas izvajanja). | Dijak program preizkuša in odpravi vsa neskladja z načrtom. Dijak demonstrira primer dveh (ali več) algoritmov za isti problem, ki nad istimi podatki zahtevata(jo) različne čase izvajanja in zna to razložiti.  | P | 3 / **Razvoj programov** / OBD33 / **Algoritmi** / OBD4 |
| Poznavanje pogostih algoritmov, poznavanje strategij razvoja algoritmov | Dijak uporablja in razloži nekaj tipičnih pristopov pri razvoju algoritmov - pregledovanje, deli in vladaj, požrešna metoda, rekurzija, sestopanje, … | D | 3 / Algoritmi / OBD4 |
| razvoj kompleksnihprogramov, knjižnice ter orodja za urejanje in obvladovanje sestavnih delov | Demonstrira uporabo orodij za razvoj programov. Dijak pokaže, da zna poiskati in uporabiti ustrezno knjižnico, ki reši kak del problema. | D | 3 / Razvoj / OBD4 |