Suvremeniji pristup poučavanju programiranja - rješavanje problema programiranjem nasuprot upoznavanja programskog jezika

Z. Markučič *, S. Perić **, L. Budin ***, P. Brođanac ****
* XV. gimnazija, Zagreb, Hrvatska
** II. gimnazija, Zagreb, Hrvatska
*** Sveučilište u Zagrebu, Fakultet elektrotehnike i računarstva, Zagreb, Hrvtaska
**** V. gimnazija, Zagreb, Hrvatska
{zmark@mioc.hr; smiljana.peric@zg.t-com.hr; leo.budin@fer.hr; predrag.brodjanac1@zg.t-com.hr}

Sažetak - Sadašnjim planovima i programima u osnovnim i srednjim školama iz područja informatike predviđa se usvajanje znanja i vještina iz tri područja: osnove uporabe i primjenski programi, građa i načela djelovanja računala te rješavanje problema programiranjem.

U području programiranja uočava se opterećenost tradicionalnim načinom učenja programskih jezika (njihove sintakse i semantike). Pritom, algoritmi, razrada i rješavanje problema programiranjem ostaju sekundarni ciljevi i zadaće.

Cilj ovoga rada je potaknuti nastavnike informatike, kao i sve sudionike u procesu oblikovanja nastave na drugačija promišljanja u smislu modernizacije predmeta našega doba, informatike.

Poznavanje osnovnih pravila nekoga programskoga jezika nužno je, te za savladavanje njegove sintakse treba planirati vrijeme. Uvođenjem novog suvremenoga programskoga jezika Python moguće je to vrijeme svesti na najmanju moguću mjeru te time dati prostora za usvajanje misaonih procesa iz područja primjene programiranja.

Nove i suvremenije metode podučavanja trebale bi u prvi plan staviti rješavanje problema kao osnovu učenja programiranja, a programski jezik bi u tom smislu trebao biti samo alat, a ne svrha samome sebi.

Kako bismo to ostvarili potrebno je dodatno osuvremeniti nastavu kao i podlogu za realizaciju nastave. Ponajprije su to nastavni programi koji bi trebali bili dobra podloga za realizaciju nastave pred kojom su novi izazovi. Nastavom i podučavanjem kod učenika se trebaju razvijati sposobnosti koje će mu pomoći u daljnjem cjeloživotnom obrazovanju i pripremiti ga za tržište rada. Učenika treba pripremiti na svakodnevno sagledavanje životnih zahtijeva, raščlambu i rješavanje različitih problema (od korištenja novih tehnologija do rješavanja problema konkurentnosti i održivoga razvoja).

I. UVOD

Nastavnim planovima i programima u osnovnim i srednjim školama iz područja informatike predviđa se usvajanje znanja i vještina iz tri područja: osnove uporabe i primjenski programi, građa i načela djelovanja računala te rješavanje problema programiranjem.

Danas se programiranje u obveznom osnovnoškolskom obrazovanju podučava isključivo u sklopu izbornog predmeta Informatika [4], a u srednjim školama ovisno o vrsti škole. U gimnazijskim i strukovnim programima Informatika ili Računalstvo je u pravilu obvezan predmet barem jednu godinu učenja sa sedamdeset sati godišnje. Opći, jezični i klasični gimnazijski programi realiziraju jednogodišnji obvezni program Informatike tijekom prve ili druge godine školovanja. U prirodoslovno matematičkom gimnazijskom programu Informatika se podučava kao obvezni predmet kroz sve četiri godine.

Propisani program je temelj na osnovi kojeg nastavnik izrađuje vlastiti izvedbeni program i temelji svoju nastavu. Stoga ćemo u radu analizirati područje programiranja u propisanim jednogodišnjim i četverogodišnjim gimnazijskim programima predmeta Informatika.

II. PRIKAZ POSTOJEĆEG STANJA U NASTAVNIM PROGRAMIMA ZA GIMNAZIJE

U jednogodišnjem gimnazijskom programu obvezni predmet Informatika predviđen je planom s dva nastavna sata, a u ostalim razredima moguće je predmet savladavati kroz izbornu ili fakultativnu nastavu. U prirodoslovno matematičkim gimnazijama Informatika je kao obvezni predmet predviđena planom kroz sve četiri godine s dva ili tri nastavna sata tjedno. Program je objavljen 1994. godine[1].

Jednogodišnji program za gimnazije jasno navodi ciljeve i zadaće predmeta. U sadržaju programa navedeno je devet nastavnih cjelina koje obrađuju sadržaje iz tri osnovna područja. Kod većine cjelina naveden je i broj predviđenih nastavnih sati (najmanji broj sati i najveći broj sati) za obradu pojedine cjeline te se time na neki način određuje opsežnost i dubina obrade sadržaja. U didaktičkoj uputi naznačeno je koje cjeline su obvezne, a koje se mogu ali ne moraju odraditi u obveznom programu te se upućuje da se cjeline obrade u drugoj i narednim godinama učenja kroz izbornu ili fakultativnu nastavu. Jedna od cjelina koja je obvezna i to s minimalno dvadeset nastavnih sati, što iznosi nešto manje od 30% od ukupnog broja sati, su osnove programiranja.

Sve većom uporabom računala za posao, učenje i za zabavu pojam informatičke pismenosti u javnosti se mijenjao. Time je stjecanje osnovne informatičke pismenosti, što je jedna od zadaća predmeta Informatika,

dobivalo drugi značaj te na određen način potisnulo sadržaje iz područja programiranja u drugi plan. Naravno tome u prilog išao je i odabir programskog jezika.

Za savladavanje sadržaja iz područja programiranja bilo je nužno učenika podučiti nekom programskom jeziku pomoću kojega će učenik moći savladati i same osnove programiranja. Za nastavnu cjelinu osnove programiranja programom je predviđeno obraditi: sustavni pristup rješavanju problema, pojam i razradu algoritma, ustroj programa, jednostavne tipove podataka, naredbe za upis, ispis, dodjeljivanje, uvjetnu naredbu if, složene naredbe – petlje, stil pisanja programa. Ove sadržaje je nastavnik u predviđenom fondu sati vrlo teško mogao kvalitetno približiti učenicima jer je izrazito puno vremena morao potrošiti na podučavanje sintakse odabranog programskog jezika. Posljedično, preostalo mu je nedovoljno vremena za razvoj učenikovih kompetencija nužnih za rješavanje problema, izradu programa, uviđanje i prihvaćanje misaonih procesa potrebnih za učinkovito planiranje i programiranje, ne samo u smislu pisanja programa, nego i u smislu rješavanja realnih zadataka u svakodnevnom i poslovnom životu.

Ciljevi i zadaće predmeta Informatika u prirodoslovno matematičkim gimnazijama su stjecanje informatičke pismenosti i stjecanje logičke discipline. Međutim, isto je tako naznačeno da učeniku treba omogućiti razvijanje stvaralačkih sposobnosti u odabiru i oblikovanju algoritama, odnosno razvoj logičkog procesa mišljenja i poticanja kritičke analize u algoritamskom rješavanju zadanih problema.

U didaktičkim uputama stoji da je svrha predmeta omogućiti učeniku odgovarajući razvoj misli koji će ga moći voditi tijekom cijeloga njegova radnog vijeka, bez obzira kako se sama računala budu mijenjala i razvijala [1]. Također je navedeno da je najvažnije naučiti sagledati suštinu problema te se preporučuje da se algoritmi prezentiraju pomoću nekog od strukturnih programskih jezika (Pascal ili neki drugi). U sadržajima programa navode se nastavne cjeline koje daju mogućnost nastavniku da odabere neki od strukturnih jezika iako se u razradi pojedinih cjelina isključivo navode imena pojedinih naredbi i elemenata programskog jezika Pascal. U nekim cjelinama se navode pojedini algoritmi koje učenici trebaju naučiti kao što su primjerice različita sortiranja podataka. U četvrtom razredu je predviđeno upoznavanje nestandardnih tipova podataka (pokazivači, dinamičke varijable, razne strukture).

III. OSVRT ZA PROPISANE I PREPORUČENE PROGRAMSKE JEZIKE U PROGRAMIMA STRUKOVNIH ŠKOLA

Za razliku od planova i programa za gimnazije gdje programski jezik nije izrijekom naveden u nekim programima u strukovnom obrazovanju izričito je naveden programski jezik i to u smislu savladavanja pojedinog programskog jezika, a ne pojedinih algoritama. Najčešće navedeni programski jezici su BASIC, C. Striktno propisivanje programskog jezika ograničava uvođenje modernijih programskih jezika.

IV. NOVIJI DOKUMENTI KOJI GOVORE O UČENIKOVIM KOMPETENCIJAMA

Osim propisanih planova i programa danas imamo i dva dokumenta koji govore o obrazovnim ishodima iz područja općeg obrazovanja koje je potrebno ostvariti tijekom dvanaestogodišnjeg obrazovanja učenika. To su Ispitni katalog za državnu maturu [2] i Nacionalni okvirni kurikulum za predškolski odgoj i obrazovanje te opće obvezno i srednjoškolsko obrazovanje [3]. U oba dokumenta navodi se da učenik treba savladati određene misaone procese i pojedine algoritme. Savladavanje pojedinog konkretnog programskog jezika nije od primarnog interesa i daje se mogućnost realiziranja obrazovnih ishoda kroz bilo koji programski jezik.

U Ispitnom katalogu za državnu maturu navedeni su obrazovni ishodi koji se temelje na jednogodišnjem obveznom učenju Informatike. Navodi se što sve pristupnik na državnoj maturi treba znati, odnosno moći iz područja rješavanja problema programiranjem. Između ostalog posebno su naglašeni standardni algoritmi za:

- zamjenu sadržaja dviju varijabli,
- prebrojavanje prema zadanome kriteriju,
- zbrajanje prema zadanome kriteriju,
- pretraživanje prema zadanome kriteriju,
- izračun srednje vrijednosti brojeva,
- traženje najmanjeg i najvećeg među (učitanim) brojevima,
- rad s prirodnim brojevima.

U Nacionalnom okvirnom kurikulumu navode se odgojno-obrazovni ciljevi za tehničko-informatičko područje u kojima se navodi da će učenik: biti osposobljen za

- uporabu računala, informacijske i komunikacijske tehnologije u učenju, radu i svakodnevnome životu,
- razviti algoritamski način razmišljanja, steći vještine i sposobnosti primjene računala pri rješavanju problema u različitim područjima primjene
- razviti sposobnosti tehničkoga i informatičkoga sporazumijevanja te uporabe tehničke i informatičke dokumentacije

V. PRISTUP PODUČAVANJU PROGRAMIRANJA

Prateći postojeće planove i programe u centar pažnje postavljena je nastavna jedinica, a to je najčešće u području programiranja jedna od naredbi odabranog programskog jezika. Navedimo par primjera.

A. Prvi primjer:

Nastavne cjeline prirodoslovno matematičke gimnazije, 2. razred.

Složene naredbe i tipovi: IF-THEN, IF-THEN-ELSE, petlje, skalarni tipovi i intervalni podtipovi (14 - 16 sati) i

nastavna cjelina potprogrami: strukturno programiranje, postupno profinjenje, TOP-DOWN dizajniranje programa, primjena potprograma u razradi problema, globalne, lokalne i formalne varijable (10 – 12 sati).

U izvedbenim programima to izgleda npr. ovako:

- naredbe grananja
    - if then naredba
    - if then else naredba
    - zadaci s if naredbom
- petlje
    - for petlja
    - zadaci s for petljom
    - while naredba
    - zadaci s while naredbom
    - ...

Nakon ovako pripremljenog izvedbenog programa slijedi klasični nastavni sat gdje se u centar pažnje stavlja savladavanje naredbe, a potom će nastavnik kroz zadatke odraditi određene algoritme. Program ne propisuje koji su to algoritmi. To mogu biti algoritmi s prirodnim brojevima kao npr. rastav prirodnoga broja na znamenke ili provjera je li broj prost.

Učeniku se na taj način daje poruka da su ti algoritmi tu radi pojedinih naredbi, a ne obrnuto. Također, ukoliko se radi o implementaciji pojedinih programskih jezika moguća su neka ograničenja kod primjene navedenih algoritama te ćemo učenicima dati poruku o ograničenjima računala. To je u nekom slučajevima poželjno, međutim kod navedenih algoritama koji nam daju velike mogućnosti primjene u matematičkim problemima ta ograničenja će biti do te mjere apsurdna da će biti nepotrebno usavršavati algoritme. Ovdje je odabir programskog jezika izrazito važan. Npr. Python dozvoljava korištenje vrlo velikih cijelih brojeva te je opravdano ulaganje vremena i truda u poboljšanje algoritama. U samoj konačnici učenik je napravio niz zadataka i naučio napisati određeni broj zasebnih algoritama.

Nasuprot ovom pristupu podučavanja, pojavljuju se novi pristupi i kod nas i u svijetu gdje se kroz nastavnu godinu postavljaju pojedini problemi (u pravilu više njih) koji na neki način zamjenjuju današnje nastavne cjeline te se za svaki problem navode učeničke kompetencije i znanja koja će steći nakon svladavanja tog problema.

Kada gledamo navedene dvije cjeline moguće je za njih postaviti određeni problem npr. igra koju igraju dva igrača. Svaki igrač ima pravo upisivati prirodan broj za kojeg misli da je prost dok ne upiše broj koji je barem troznamenkast. U igri se određuje pobjednik tako da pobjeđuje igrač koji je upisao prost broj koji ima više parnih znamenaka, neriješeno je ukoliko oba igrača upišu prost broj s jednako parnih znamenaka ili oba igrača upišu složeni broj.

Nakon ovako postavljenog problema uočimo što učenik mora znati, odnosno što učenik mora naučiti.

Učenik treba znati:

- koji je najmanji troznamenkasti broj
- što je prirodan broj
- svojstvo parnosti
- što je prost broj
- što je složen broj
- pojam znamenke
- učitati cjelobrojni podatak
- ispisati string
- naredbu pridruživanja
- operatore nad cjelobrojnim tipom
- pozivanje i funkciju sqrt
- logički tip podataka
- if naredbu odnosno postavljanje logičkih izraza

Učenik će:

- postupno razraditi i analizirati problem
- uvidjeti da je potrebno omogućiti višestruki upis broja dok se ne upiše troznamenkasti broj
- izdvojiti znamenke iz broja
- usvojiti algoritam za provjeru je li neka znamenka parna
- provjeriti je li broj prost
- naučiti sintaksu i način uporabe naredbe s uvjetom
- naučiti postavljati logičke uvjete
- naučiti sintaksu i način upotrebe naredbe s konačnim bojem ponavljanja
- uvidjeti da se neki algoritmi više puta koriste, ali s različitim ulaznim i izlaznim podacima
- naučiti i primijeniti potprograme u programu

Pri ovakvom rješavanju problema moramo biti svjesni da se zadatak neće moći riješiti u jednom nastavnom satu. Potrebo je razraditi dinamiku rješavanja problema te postepeno uvođenje novih znanja. Kod savladavanja novih znanja pojavit će se potreba za uvođenjem manjih zasebnih problema koji će učenika voditi cilju. Ovakvo postavljenje problema i razrada rješenja omogućuje nam korištenje alternativnih izvora znanja, suradnju s nastavnikom matematike, te u konačnici razvijanje kreativnosti kroz postavljanje novih problema i njihovo rješavanje.

B. Drugi primjer:

Nastavne cjeline prirodoslovno matematičke gimnazije, 2. ili 3. razred. Složeni tipovi podataka – string, skup: operacije i gotovi potprogrami koji se mogu primijeniti na stringu ili skupu (12 - 16 sati) i nastavne cjeline složeni tipovi podataka – polja (11 – 20 sati):

pojam indeksa, jednodimenzionalno polje, obrada niza petljom, pretraživanje ...

U izvedbenim programima to izgleda npr. ovako:

- niz znakova:
  - string
  - standardne funkcije
  - definirane nad stringom
  - zadaci sa stringom
- polja
  - jednodimenzionalni niz
  - indeksiranje elemenata
  - upis i ispis elemenata niza
  - ...

Naravno i ovo područje možemo obraditi kroz postavljanje problema koji će nas ujedno dovesti do poznavanja pojedinih algoritama, usvajanja znanja o tipovima i prikazu podataka u računalu, ali isto tako o sveopćem problemu zaštite podataka u računalnim mrežama i analizi teksta, dakle vrlo širokom i interdisciplinarnom području.

Postavljanje problema o pojmu zaštite podataka. Ovdje ne postavljamo konkretan problem već poopćen i vezan za stvarno okruženje, bankovni računi, kupovina putem Interneta, zaštita komunikacije, pojam lozinki, pinova, tanova i tako redom. Počinjemo s pojmom kriptografije, osnovnim pojmovima, povijesnim okruženjem, postojanjem stvarne potrebe za zaštitom podataka.

Ponovno postavljamo što učenik treba znati te koje će kompetencije steći nakon obrade ovog problema. Povezujemo se s predmetima Povijest, Geografija, Matematika i naravno jezici.

Učenik nakon rješavanja problema stječe razne kompetencije:

- razlikovanje kodiranja velikih i malih slova,
- uočavanje specifični slova koji su vezana za nacionalne abecede u odnosu na englesku abecedu
- razlikovanje različitih standardnih i nestandardnih kodiranja znakova
- uočavanje rada sa znakovnim tipovima podataka
- indeksiranje liste (polja)
- korištenje standardnih funkcija i metoda nad stringovima
- formatiranje ispisa stringa
- ...

VI. OSVRT NA NAJČEŠĆE KORIŠTENE PROGRAMSKE JEZIKE U PODUČAVANJU PROGRAMIRANJA I UDŽBENIČKU LITERATURU

Danas se u podučavanju učenika najčešće koriste strukturni programski jezici: LOGO u osnovnoj školi, razne inačice BASICa u osnovnoj i srednjim školama, Pascal i C u srednjim školama. Ovi jezici u određenoj mjeri mogu zadovoljiti postizanje osnovnih ciljeva i zadaća iz područja rješavanje problema programiranjem, ali ti jezici će nas vrlo često upućivati na tumačenje strukture tih programskih jezika, statičkih varijabli i orijentirati nas na razvoj logike konkretnih programskih jezika. U određenim školama kao što su prirodoslovno matematičke gimnazije podučavaju se i jezici s naprednijim konceptima u kojima se pojavljuju objektno-usmjerene paradigme. Pojavljuju objektni jezici (C++, C#, Java). Ti su jezici smišljeni za podržavanje velikih programskih projekata s naglaskom na discipliniranu izgradnju složenih programskih struktura. Ti jezici nisu prikladni za ostvarivanje manjih ili srednjih programa.

Naravno gledajući postojeće planove i programe odabiri ovih programskih jezika mogu se opravdati, posebice kada se pogleda iz koje godine (1994.) je većina propisanih planova i programa iz područja informatike [5].

Odabir programskog jezika je najčešće je vezan i za propisanu udžbeničku literaturu u kojoj se prvenstveno koriste programski jezici (LOGO, BASIC, Pascal i C)

VII. KAKO DALJE

Kako bismo i naredne generacije kvalitetno pripremali za rad i osobni probitak nužno je uvidjeti što je i gdje je informatika danas. Pojavom mobilne telefonije, brzog razvoja telekomunikacija i mrežnih struktura, razvoja računalne industrije te ugradnja mikro sklopova u većinu modernih strojeva i pomagala informatika danas dobiva novu ulogu i dimenziju. U tom svjetlu potrebno je sagledati i područje programiranja. Osim što se pred nas postavlja izazov da kod učenika razvijemo algoritamsko razmišljanje, potrebno je učenika podučavati i novim paradigmama programiranja. Taj proces će nužno zahtijevati i uvođenje Informatike kao obveznog predmeta u osnovne škole. Također je potrebno planirani broj sati za Informatiku osmisliti tako da se postignu ciljevi već spomenutog Nacionalnog okvirnog kurikuluma [3], ali i najnovije svjetske preporuke CSTA K-12 [6].

Isto tako bilo bi dobro da postoji jedinstveni programski jezik koji bi, s jedne strane, bio dovoljno jednostavan kako bi ga mogli svladati početnici u programiranju. S druge strane, trebao bi biti blizak današnjim profesionalnim jezicima s razvijenom paradigmom objektno-orijentiranog programiranja. Takav jezik, odnosno jezik najbliži takvom idealnom, je Python [5, 7, 8].

Razlozi za odabir programskog jezika Python za učenje programiranja u općeobrazovnom sustavu obrazovanja su:

- Python ima čitku sintaksu, što je jedan od najjačih razloga za njegov izbor,

- programi pisani u Pythonu lako se čitaju i imaju veliku pedagošku vrijednost,
- dodatna mu je vrijednost da se može koristiti na interaktivni način,
- u Pythonu se mogu izraziti paradigme različitih jezika,
- to je dobar prvi jezik ne samo za buduće programere u jezicima Java i C++ (objektno usmjereni jezici) već i za buduće programere u jezicima Haskell i Lisp (funkcionalni jezici),
- Python nije mrtav jezik i još se stalno razvija.
- popis tvrtki koje u svojim projektima koriste Python je već velik i brzo raste,
- Python je potpuno besplatan za sve korisnike, što znači da ga svi učenici/učenice mogu koristiti na vlastitom računalu – što znatno potpomaže njegovo savladavanje,
- svaka škola može slijediti politiku: u našoj instituciji nema nelicenciranih programa.

VIII. ZAKLJUČAK

Iz svega navedenoga vidljivo je da nije osnovna zadaća podučavanje sintakse programskih jezika, već je bolje odabrati jedan programski jezik u kojem se može implementirati algoritam na nižoj kao i na vrlo visokoj razini.

Jedan od takvih jezika je programski jezik Python koji se može implementirati u nastavni proces vrlo rano jer ne zahtjeva veliko znanje sintakse, omogućuje učeniku kao i nastavniku da sintaksu svladava postupno te algoritam postaje centar podučavanja. U trenutcima kada neke dijelove algoritma nismo u mogućnosti u potpunosti objasniti učenicima radi nedovoljno razvijenih i usvojenih kompetencija iz ostalih područja (matematika, fizika, itd) kao iz samog područja programiranja moguće je jednostavno koristiti postojeću biblioteku metoda bez ulaženja u dubinu.

Isto tako novim nastavnim planovima i programima treba omogućiti uvođenje novih paradigmi podučavanja programiranja..

LITERATURA

[1] Glasnik Ministarstva kulture i prosvjete, Zagreb, 1994

[2] Ispitni katalog za državnu maturu, Zagreb, 2011 http://www.ncvvo.hr/drzavnamatura/web/public/katalozi12

[3] Nacionalni okvirni kurikulum za predškolski odgoj i obrazovanje te opće obvezno i srednjoškolsko obrazovanje, MZOŠ, Zagreb, 2011

[4] *Hrvatski nacionalni obrazovni standard za osnovnu školu*, Ministarstvo znanosti obrazovanja i športa, Zagreb, 2005

[5] P. Brođanac, L. Budin, Z. Markučič, S. Perić, *Python – jezik za podučavanje algoritamskog pristupa rješavanju problema*, MIPRO, Opatija, 2010

[6] *A Model Curriculum for K–12 Computer Science*, Final report of the ACM K–12 Task Force Curriculum Committee, *Revised 2011* http://csta.acm.org/Curriculum/sub/CurrFiles/CSTA_K-12_CSS.pdf

[7] T.Donaldson, *Phyton as a First Programming Language for Everyone* http://www.cs.ubc.ca/wccce/Program03/papers/Toby.html

[8] V. Leping, M. Lepp, M. Niitsoo, E. Tõnisson, V. Vene, A. Villems, *Python, Prevails, International Conference on Computer Systems and Technologies*, CompSysTech'09