Prijedlog optimalnog korištenja gradiva matematike za usavršavanje vještina za programiranje

AUTOR 1: Aleksander Radovan, HUJAK

AUTOR 2: Branko Mihaljević, HUJAK

Sažetak

Jedna od najvećih dilema kurikularne reforme koja se planira objaviti 2016. godine i provesti narednih godina se sastoji u tome da se odluči je li informatika obvezan predmet ili ne. Tržište rada u Hrvatskoj i inozemstvu vapi za velikom količinom obrazovanih IT stručnjaka [1] i već se sada može predvidjeti da će do 2020. godine u Evropi nedostajati 900.000 stručnjaka ovog profila [2]. Kako bi se što kvalitetnije pripremilo mlade stručnjake za zanimanja iz tih područja, njihovo usavršavanje mora započeti već u osnovnoj školi kroz razne module postojećih predmeta kao što je matematika, osim same informatike koja bi za rješavanje navedenih problema morala biti apsolutni imperativ.

Pomno odabranim poglavljima i područjima matematike učenicima je moguće već u ranim fazama obrazovanja dati uvid u najpotrebnija znanja za razvijanje „programerskog razmišljanja" te analize i rješavanja problema koji se pojavljuju njihovom u svakodnevnom radu. S dobrim temeljima znanja najbitnijih matematičkih koncepata učenicima je puno lakše percipirati, savladavati i usvajati potrebne tehnike, principe i algoritme kako bi optimalno razvili svoje vještine potrebne za stvaranje uspješne karijere u području programiranja.

U ovom radu autori će predložiti područja matematike koja bi trebala biti zastupljenija u nastavi osnovnoškolskog i srednjoškolskog obrazovanja, uzevši u obzir najaktualnije trendove na globalnom tržištu.

Uvod

Programiranje u raznim programskim jezicima se dugo smatralo kao vještina koju posjeduje samo nekolicina vrlo talentiranih pojedinaca i koja se teško razvija. Povećana potreba za programerima i skraćivanjem vremena potrebnog za isporuku softverskih rješenja je rezultirala razvojem alata koji drastično pojednostavljuju sam proces pisanja programskog koda, bez obzira na programski jezik i tehnologiju koja se koristi. Bez obzira na to, u Hrvatskoj se proteklih godina dogodio pad upisnih kvota za studije računarstva [3], što će uz pad nataliteta u Hrvatskoj dugoročno predstavljati veliki problem. Maturantima će često privlačnije upisivati društveno-humanističke studije koji rezultiraju profesijama koje nisu toliko tražene na tržištu rada [3], a jedan od uzroka tog obrasca ponašanja leži u predznanju koje se zahtijeva za tu vrstu studija. Uvođenje informatike od najranijih početaka formalnog obrazovanja kao i naglašavanje određenih gradiva matematike može djelovati pozitivno na rješavanje tih problema.

Odabir odgovarajućeg programskog jezika za ulazak u svijet programiranja

U srednjim školama u Hrvatskoj su proteklih godina bili najzastupljeniji programski jezici Pascal i C, međutim globalni trendovi i napredak tehnologije zahtijevaju promjene i na tom području [4][5]. Programski jezik poput Pythona omogućava olakšano korištenje metoda kao što su objektno orijentirano programiranje i funkcijsko programiranje, koje su krucijalne za korištenje najzastupljenijih programskih jezika kao što je programski jezik Java, C# i JavaScript [6].

Autori iz osobnog iskustva znaju da studentima najveći problem predstavlja ispravljanje pogrešaka u programskom kodu i postizanje stanja ispravnog rada programa, a fleksibilnost programske sintakse kod programskog jezika Python dopušta korištenje ključnih riječi na intuitivan način poput pisanja rečenica u engleskom jeziku, npr. „if vrijednost not in skupVrijednosti“. Također, Python omogućava promjenu vrijednosti dviju varijabli unutar jedne naredbe: „a,b = b,a“, za što je u drugim jezicima potrebno koristiti tri naredbe, čime se dodatno potvrđuje jednostavnost i prikladnost tog jezika za početak učenja programiranja. Rješavanjem problema vezanim uz sintaksu programiranja, uz korištenje odgovarajućih besplatnih alata koji dodatno olakšavanju pisanje programskog koda i pronalaženje pogrešaka u radu programa, učenici na lakši način mogu vidjeti rezultate svog programiranja kroz pokretanje ispravnog programa, što na njih djeluje motivirajuće za nastavak učenja i usavršavanja.

Ključna gradiva matematike za najvažnije vještine programiranja

Autori su kroz dugogodišnje iskustvo u programiranju kompleksnih informacijskih sustava i radu u visokoškolskim ustanovama došli do zaključka da su studentima potrebna dodatna predznanja iz dijelova gradiva matematike opisanih u nastavku s kojima bi se budući programeri trebali „susresti“ u što ranijim fazama svog obrazovanja. Vrlo važan segment programiranja je i kreativno razmišljanje te rješavanje istog problema na različite načine, što također treba ostati sačuvano kao jedna od sloboda u kreiranju vlastitih rješenja bez obzira na nužnost poštivanja zajedničkih koncepata i najboljih praksi. Gradiva matematike koja su važna za razvoj optimalnih programerskih vještina su sljedeća:

Logička algebra

U programiranju se vrlo često moraju konstruirati vro složeni upiti i provjere koje omogućavaju ugrađivanje poslovne logike specifične za domenske probleme koji se nastoje riješiti uvođenjem informacijskih sustava. Razlaganje složenog problema na više jednostavnijih problema i povezivanje tih dijelova na više različitih mjesta kako bi se dobio efekt ponovnog iskorištavanja je vrlo često prisutno kod programiranja kako bi se spriječilo kopiranje jednog te istog koda na više mjesta. Korištenjem logičkih tablica (ili često nazivanih „tablicama istine“) učenici mogu steći osnovno logičko razmišljanje po pitanju definiranja i uvođenja vlastitih složenih logičkih funkcija koje se sastoje od osnovnih „I“ i „ILI“ logičkih funkcija i koje se mogu višestruko iskorištavati u različitim scenarijima.

Statistički proračuni

Poslovne aplikacije gotovo uvijek podrazumijevaju uključivanje modula koji omogućavaju generiranje statističkih izvještaja pa je nužno poznavanje matematičke pozadine koja to omogućava. Odabir najprikladnijeg algoritma po pitanju kompleksnosti i performansi također utječe na uspješnost programa koji ih koristi te zadovoljstvo korisnika sustava koji koristi dobivene podatke. Grupiranje rezultata istog tipa radi što bolje čitljivosti pomaže u odabiru podatkovne strukture ugrađene u programski jezik koji se koristi kako bi se sa što manje linija koda dobio što efektniji prikaz.

Kombinatorika

U osmišljavanju vlastitih algoritama za rješavanje najčešćih problema vrlo veliku ulogu ima kombinatorika. Ona omogućava stjecanje potrebnih znanja za dizajniranje programskih petlji koje prolaze po cijelom skupu zadanih vrijednosti i unutar njega traže sve moguće kombinacije traženog rješenja, samo dio kombinacija ili samo jedno optimalno rješenje koje je potrebno producirati. Razmišljanje o načinu „pobrojavanja“ i planiranja petlje može drastično smanjiti kompleksnost algoritma koji se koristi, a samim time i vrijeme potrebno za dobivanje rezultata iz velike količine podataka. Kod potrebe korištenja tehnike rekurzivnih poziva funkcija je također potrebno na ispravan način detektirati rubne slučajeve u kojima rekurzija mora završiti, kako bi se izbjeglo stvaranje beskonačne petlje koja nikad ne završava.

U svijetu umjetne inteligencije vrlo često je potrebno koristiti nekonvencionalne algoritme koji po uzoru na procese u prirodi pronalaze rješenja praktičnih problema. Jedan od takvih problema problem trgovačkog putnika [7] koji se može riješiti korištenjem genetskih

algoritama. Korištenjem posebno optimiziranog algoritma moguće je riješiti problem bez nužnog korištenja svih mogućih kombinacija rješenja te umjesto toga koristiti principe prirodne evolucije na koji su implementirani pomoću genetskih algoritama [8].

Kartezijev višedimenzionalni koordinatni sustav i vektori

U teoriji igara veliku ulogu igraju koordinatni sustavi kojima je moguće definirati poziciju objekta u nekom prostoru, a kretanje je moguće opisati vektorima koji osim smjera označavaju i brzinu. Obavljanje raznih transformacija kao što su rotacija i translacija objekta koji se kreće, određivanje pozicije „oka kamere“ koja prikazuje neku scenu, određivanje vidnog polja neprijatelja od kojeg se glavni lik treba sakriti ili detekcija kolizije 3D objekata samo su neki od često korištenih scenarija u programiranju igara i simulacija u kojima je od krucijalne važnosti snalaženje u dvodimenzionalnom i trodimenzionalnom prostoru. Dobro snalaženje u Kartezijevim koordinatnim sustavima kao i povezivanje s koordinatnim sustavom na osnovni kojeg funkcioniraju računalni zasloni može značajno smanjiti vrijeme potrebno za implementaciju određenih scenarija ili scena u takvoj vrsti aplikacija ili igara.

Rad s funkcijama

Programska paradigma funkcijskog programiranja, koja je vrlo popularna u svim modernim programskim jezicima temelji se na tome da računalni programi koriste matematičke funkcije temeljene na vraćanju izlaznih vrijednosti koje ovisne o ulaznim vrijednostima na način da ne mijenjaju ulazne vrijednosti. Na toj paradigmi počivaju koncepti poput lambda izraza, koji su vrlo popularni jer smanjuju količinu koda koji je potrebno napisati da bi se ostvarila ekvivalentna funkcionalnost i iskoristio potencijal višejezgrenih procesora bez nužnosti poznavanja internih detalja njihove arhitekture. Prethodni rad s matematičkim funkcijama pospješuje razumijevanje koncepata i ubrzava proces učenja paradigme funkcijskog programiranja.

Logičko razdvajanje programskog koda u zasebne cjeline koje imaju jasno definirane ulazne i izlazne parametre i koje se višestruko iskorištavaju na više mjesta je također vrlina kvalitetnih programera, ali i nužnost kod detekcije zajedničkih funkcionalnosti aplikacije koje se mogu izložiti vanjskim sustavima pomoću tzv. REST API (engl. *Representational State Transfer Application Programming Interface*) sučelja.

Traženje ekstrema u funkcijama

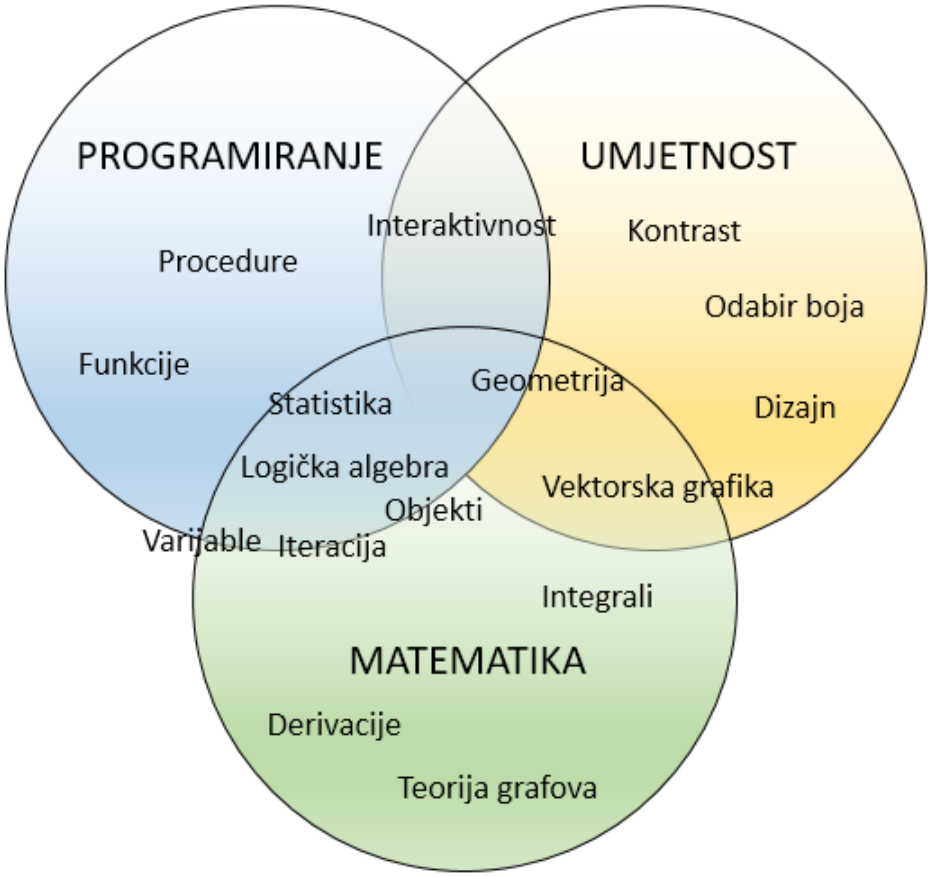
U korištenju naprednih algoritama koji su često vezanu uz umjetnu inteligenciju, genetske algoritme i korištenje umjetnih neuronskih mreža često se pojavljuje pojam „lokalnog ekstrema“ zbog kojeg program ne vrati najbolji mogući rezultat. Promjenom ulaznih parametara i optimizacijom algoritma pretraživanja moguće je efikasnije izbjegavati ulazak u lokalne ekstreme i uštediti dosta vremena ako programer razumije matematičke koncepte rada s ekstremima.

Ograničenja programera u slučaju nedostatka potrebnih matematičkih znanja

Iako su matematička znanja od vrlo velikog značaja da bi se postalo dobrim programerom, u praksi se često dogodi da i stručnjaci kojima matematičke vještine nisu jača strana, ostvare solidnu karijeru kao programeri. Međutim, zbog nedostatka širine znanja prilikom prvog susreta sa specifičnim problemom, potrebna je pomoć od kolega koji raspolažu tim znanjima. Nesamostalnost u takvim situacijama predstavlja ograničenje i kod napredovanja u karijeri. Repetitivno ponavljanje nekih usvojenih tehnika ne predstavlja problem, ali nailaskom na neki od problema ili izazova koji se ne javljaju vrlo često rezultira „zastojem“ kod kojeg trebaju intervenirati iskusniji članovi tima.

Današnji programski okviri (engl. *framework*) na kojima se temelji moderno programiranje i koji drastično ubrzavaju vrijeme potrebno za dovršetak implementacije, često nude unaprijed pripremljene funkcionalnosti koje se konfiguriraju i ugrađuju u vlastita rješenja uz minimalno truda i „sakrivaju“ nepotrebne detalje implementacije. Time naizgled eliminiraju potrebu za razumijevanjem složenih koncepata koji se koriste, ali čim nastanu problemi zbog koji je potrebno pokazati razumijevanje internih detalja, to se pokazuje netočnim.

Bez obzira na to, praksa iskusnih voditelja razvojnih timova također pokazuje da je produktivno unutar istog tima ima više različitih „vrsta“ programera koji posjeduju različita znanja, jer svaki projekt razvoja informacijskog sustava uključuje zadatke koji zahtijevaju samostalna istraživanja i korištenja naprednih tehnika i praksi, ali osim toga ima i pregršt repetitivnih zadataka za koje je potrebno koristiti samo prijašnje iskustvo i automatizirano ga koristiti.



Slika 1. Preklapanje elemenata programiranja, matematike i umjetnosti [9]

Zaključak

Za ublažavanje problema nedostatka kvalitetnog kadra iz kojeg će se razviti programeri konkurentni na globalnom i lokalnom tržištu potrebno je djelovati u što ranijim fazama školovanja. Znanja iz područja matematike usko su povezana s programerskim vještinama i svaki ambiciozan programer mora biti svjestan da je za postizanje potrebne samostalnosti za napredovanje u karijeri nužno usavršavanje na svim relevantnim područjima. Sposobnost matematičkog razmišljanja je krucijalna u područjima poput kod dizajniranja računalne grafike, obrade signala ili kriptografije, što je vrlo često prisutno kod razvoja raznih programa i aplikacija. Osim matematike, kod programiranja i oblikovanja korisničkog grafičkog sučelja često je potrebna i razina „umjetnosti“ za odabir kompatibilnih boja, kontrasta, oblika i elemenata za komunikaciju korisnika s računalnim programom. Na slici 1. prikazan je graf koji oslikava međusobnu ovisnost triju disciplina: programiranja, matematike i umjetnosti. Za postizanje optimalne samostalnosti programeri također moraju znati dovoljno o drugim dijelovima informacijskog sustava kao što su baze podataka. Neprestana potreba za usavršavanjem i praćenjem aktualnih tehnologija je obveza svakog modernog programera, jer

napredovanjem hardverskih mogućnosti informacijskih sustava rastu i mogućnosti koje je mogu softverski implementirati te se time ubrzano razvijaju nove tehnologije, a istom brzinom zastarijevaju one koje više nisu aktualne.

S obzirom na povećanu potražnju za IT kadrom s vještinama programiranja, trenutno dostupni resursi na tržištu rada ne pokrivaju te potrebe pa se mladi stručnjaci zapošljavaju i prije nego što završe sa studiranjem. Zato je još važnije tijekom srednje škole steći dovoljna formalna predznanja iz predmeta povezanih s matematikom kako bi bili što uspješniji u svom poslu i svoje znanje i iskustva prenosili na mlađe generacije.

Literatura

[1] eSkills inicijativa, Zagreb 2014. Dostupno na <http://eskills.hr/ako-zelimo-povecati-konkurentnost-ukupnog-gospodarstva-nuzno-je-prilagoditi-sustav-obrazovanja-novim-trzisnim-zahtjevima-ili-ce-nas-konkurencija-prestici/>, 27.05.2016.

[2] Nedostatak IT kadrova, Zagreb, 2015. Dostupno na <http://eskills.hr/stanje-na-trzistu-rada-71-posto-ict-poslodavaca-otvarat-ce-nova-radna-mjesta-a-najtrazeniji-su-programeri/>, 27.05.2016.

[3] Rezultati analize interesa za STEM studije, Zagreb, 2014., Dostupno na <http://eskills.hr/rezultati-analize-interesa-za-stem-studije/>, 27.05.2016.

[4] P. Guo: Popularnost Pythona kao programskog jezika za početnike, S.A.D, 2014., Dostupno na <http://cacm.acm.org/blogs/blog-cacm/176450-python-is-now-the-most-popular-introductory-teaching-language-at-top-us-universities/fulltext>, 27.05.2016.

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

[6] PYPL Popularity of Programming Language, Dostupno na <http://pypl.github.io/PYPL.html>, 27.05.2016.

[7] Problem trgovačkog putnika, Dostupno na <http://mathworld.wolfram.com/TravelingSalesmanProblem.html>, 27.05.2016.

[8] V. Bradvica: Genetski algoritmi, Zagreb, 2007. Dostupno na <http://www.zemris.fer.hr/~golub/ga/studenti/seminari/2007_bradvica/>, 27.05.2016.

[9] Kako postati dobar programer, 2014., Dostupno na <http://www.freejupiter.com/how-to-be-a-good-programmer/>, 27.05.2016.