Algoritmi za ispitivanje djeljivosti

Đurđević, Danijel

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA

Preddiplomski stručni studij Elektrotehnika, smjer Informatika

ALGORITMI ZA ISPITIVANJE DJELJIVOSTI

Završni rad

Danijel Đurđević

Osijek, 2016.

Sveučilište Josipa Jurja Strossmayera u Osijeku

ELEKTROTEHNIČKI FAKULTET OSIJEK

Obrazac Z1S: Obrazac za imenovanje Povjerenstva za obranu završnog rada na stručnom studiju

Osijek, 2016

Odboru za završne i diplomske ispite

Imenovanje Povjerenstva za obranu završnog rada na stručnom studiju

Ime i prezime studenta:	Danijel Đurđević
Studij, smjer:	Stručni studij elektrotehnike, smjer Informatika
Mat. br. studenta, godina upisa:	A4096, 2011.
Mentor:	doc.dr.sc. Tomislav Rudec
Sumentor:	-
Predsjednik Povjerenstva:	
Član Povjerenstva:	
Naslov završnog rada:	**Algoritmi za ispitivanje djeljivosti**
Primarna znanstvena grana rada:	Matematika, sekundarno računarstvo
Sekundarna znanstvena grana (ili polje) rada:	
Zadatak završnog rada	Obraditi djeljivost matematičkim alatima i izraditi aplikaciju u programskom jeziku C.
Prijedlog ocjene pismenog dijela ispita (završnog rada):	Vrlo dobar (4)
Kratko obrazloženje ocjene prema Kriterijima za ocjenjivanje završnih i diplomskih radova:	Primjena znanja stečenih na fakultetu: 2 Postignuti rezultati u odnosu na složenost zadatka: 2 Jasnoća pismenog izražavanja: 2 Razina samostalnosti: 2

Potpis sumentora:	Potpis mentora:

Dostaviti:

1. Studentska služba

U Osijeku, 2016 godine

Potpis predsjednika Odbora:

ELEKTROTEHNIČKI FAKULTET OSIJEK
Sveučilište Josipa Jurja Strossmayera u Osijeku

IZJAVA O ORIGINALNOSTI RADA

Osijek,2016

Ime i prezime studenta:	Danijel Đurđević
Studij :	Stručni studij elektrotehnike, smjer informatika
Mat. br. studenta, godina upisa:	A4096, 2011 godina

Ovom izjavom izjavljujem da je rad pod nazivom**: Algoritmi za ispitivanje djeljivosti**

izrađen pod vodstvom mentora Doc. dr. sc. Tomislava Rudeca

i sumentora

moj vlastiti rad i prema mom najboljem znanju ne sadrži prethodno objavljene ili neobjavljene pisane materijale drugih osoba, osim onih koji su izričito priznati navođenjem literature i drugih izvora informacija.
Izjavljujem da je intelektualni sadržaj navedenog rada proizvod mog vlastitog rada, osim u onom dijelu za koji mi je bila potrebna pomoć mentora, sumentora i drugih osoba, a što je izričito navedeno u radu.

Potpis studenta:

SADRŽAJ

1. UVOD

Ovaj završni ima zadatak ispitati algoritme za djeljivost. Kako bi se pojedini dijelovi i pojedina svojstva pojasnila potrebno je definirati strukturu samih algoritama kao funkcije koja nosi određene zadatke. Kako bi se pojasnio algoritam za djeljivost potrebno je definirati pojedine segmente djeljivosti potkrijepljene teorijskom matematikom. Svaki algoritam nosi matematičku strukturu koja po tome ima različite zadatke izvršavanja prema različitim redoslijedima. Ovaj rad detaljnije će pojasniti zadane strukture prema ispitivanju djeljivosti.

Rad će prikazati primjer algoritma koji se upotrebljava za ispitivanja djeljivosti. Kao okosnicu i formalni primjer koristit će se Euklidov algoritam koji predstavlja strukturalan primjer. Ovaj završni rad sastoji se od samoga uvoda koji se nastavlja poglavljem posvećenom algoritmima od samoga nastanka kroz povijest do svojstva i primjera. Slijedi teorem o djeljivosti koji postavlja uvjet funkcionalnosti samoga algoritma, te Euklidov algoritam kao primjer algoritma za ispitivanje djeljivosti. Završno poglavlje nosi naslov dokazi algoritma kojima će biti potkrijepljene pojedine tvrdnje i sadržaji cjelokupnoga rada kroz zadane primjere koji se koriste i u pojedinim programskim jezicima.

Problem nastaje kod samog pojašnjavanja algoritama, što je zapravo algoritam i koju informaciju odnosno funkciju on ima, stoga je potrebno detaljno opisati algoritme koji su vezani za određene radnje i time čine određeni naputak za izvršavanje pojedinog zadatka. Koja su svojstva algoritama i pojedini primjeri za pojašnjeno djelovanje i rad istih čine elementarni dio rada kojim je potrebno prvenstveno prikazati korištenje i svrhu istih.

Ovaj rad interpretiran je kroz nekoliko cjelina, prva i osnovna cjelina čine sami algoritmi koji su predstavljeni kroz svoje značenje , zatim slijedi poglavlje opisivanja matematičke podloge za pojašnjenje djeljivosti. Zatim slijedi poglavlje koje opisuje daleko najvažniji algoritam elementarnoga značenja odnosno Euklidov algoritam i za kraj slijedi poglavlje u kojem će biti interpretirani algoritmi za ispitivanje djeljivosti u različitim programskim jezicima.

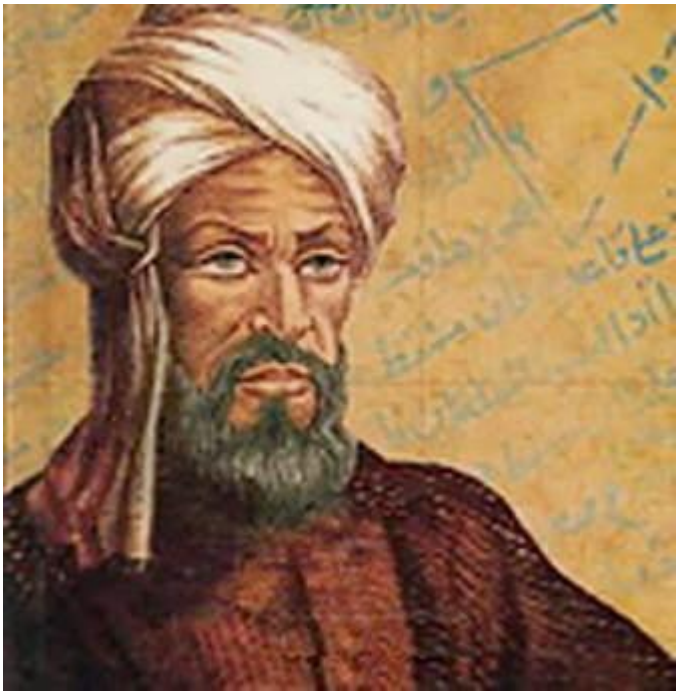
2. ALGORITMI

Algoritmi su sastavni dio svakodnevice, ali u različitim oblicima, što se interpretira kao svakodnevno rješavanje problematike odnosno svakodnevno rješavanje pojedinih operacija. Svakodnevica je time poistovjećena s različitim obavljanjem zadanih operacija u vidu poslova sa zadanom naznakom na određeni zadani redoslijed. To rješavanje pojedinih poslova kreće još od čovjekove mladosti kada dijete oponašanjem odraslog stječe određene navike i određena učenja što se svojevrsno također može predstaviti kao zadani algoritam. Kao najjednostavniji primjer kojim bi se dočaralo svojstvo algoritma, uzeo sam primjer sa prelaska ceste. Ovaj primjer se svodi na nekoliko koraka:

- Pogled na lijevo
- Pogled na desno
- Ako je vozilo na cesti u pokretu – stati i čekati
- Ako nema vozila u pokretu na cesti – pripremiti se za prelazak ceste
- Ubrzani prelazak ceste

Ovo je samo jedan od niza svakodnevnih algoritama, no pojedine upute za korištenje primjerice kućanskih aparata ili određenih drugih strojeva u poslovnom okruženju, također se mogu smatrati svojevrsnim algoritmom jer se radi o zadanom naputku s određenim redoslijedom za obavljanje određene operacije.

Algoritam kao riječ, po prvi puta se pojavljuje u perzijskom podneblju, od strane priznatog matematičara Muhamed ibn Musa al Horezmi u prijevodu s arapskog jezika to bi značilo: Muhamed sin Muse iz Horezma koji je živio u IX. stoljeću.[2][4]



Slika 1.1. Muhamed ibn Musa al Horezmi

To postojanje i prvobitno predstavljanje algoritma bilo je otkriveno tek nakon što se pojavila knjiga prevedena na latinski. Knjiga na arapskom je izgubljena, no u prijevodu knjige na latinskom iz XII. stoljeća se ispred svakog pravila navodi "Dixit Algorizmi". Muhamed ibn Musa al Horezmi je razradio i u svojoj knjizi opisivao pravila za provođenje aritmetičkih operacija s određenim zapisima u obliku brojeva unutar dekadskog oblika. "Dixit Algorizmi" u prijevodu znači "Algorizmi je govorio", time je dana naznaka da se s latinskog prijevoda koristio zadnji dio imena odnsono „al Horezmi" što je pretvoreno u Algorizmi. Ovaj prvobitni oblik latinskog smatra se kao preteča onom pravilnom latinskom, prema ostalim latinskim tekstovima i zapisima se riječ „Algorizmi" pretvorila u „Algoritmus" što je u prijevodu na hrvatskom „Algoritam". Iz svega se može zaključiti da je riječ deformirana iz koje se apsolutno ne može izdvojiti ime čovjeka koji je zaslužan za to ali i natpis prije samih pravila od čega se "Algoritam je govorio" pretvara u "Algoritam glasi". Tako je nastao pojam i od tada se koristi riječ algoritam za pravilo. No, sam početak korištenja algoritma kao riječ i kao uputa odnosno pravilo, koristi se samo kao pravilo za računanje brojeva unutar dekadskog sustava brojeva. Nakon čega se kroz različite periode unutar više stoljeća riječ algoritam počela koristiti kao naziv za pravila obavljanja raznih zadataka i operacija. Nastavak će prikazati tri primjera algoritama te svojstva istih.[4]

Primjer 1.

Jedna vrsta algoritama koji se mogu koristiti kao primjer unutar svakodnevnog života su recepti za kuhanje. Kada je u pitanju recept prema kojemu se pravi zadano jelo postoji određen broj sastojaka koji čine tzv. input. Nakon opisa sastojaka i količine istih postoji naputak prema kojemu se zadanim redoslijedom priprema svaki sastojak pojedinačno te dolazi do miješanja već izvršenih manjih postupaka te novih sastojaka u jednu krajnju cjelinu ili čak više njih. Unutar ovoga primjera svojstva algoritma čine početni sastojci koji su jasno definirani određenom zadanom količinom. Rezultat je jelo koje je napravljeno zadanim naputkom

Primjer 2.

Kao algoritam u svakodnevnom životu može biti i uputa za sijanje pojedinih poljoprivrednih kultura, gdje se neposredno prije sijanja mora pripremiti tlo te uz same inpute poput sjemenki naznačiti i ostale inpute poput gnojiva i slično, što znači da ulaznih veličina ima više. Prema zadanim uputama postoje određena pravila koja trebaju biti poštivana kako bi cjelokupni krajnji rezultat bio izvršen na najpravilniji način. Svojstva ovog algoritma čine sjemenke kao glavna ulazna veličina pa do vode i gnojiva koji su bitni faktori za izvršavanje. Rezultat čini bilje poljoprivredne kulture koje je nakon određenom vremena dalo plod.

Primjer 3.

Kao treći primjer prikaza načina algoritama postavlja se zadatak u vidu zbrajanja dvaju ili više decimalnih brojeva u dekadskom sustavu sa zapisom brojeva čiji su cijeli ili razlomljeni dijelovi razdvojeni decimalnom točkom. Ali s određenim naputkom koji predstavlja algoritam te glasi: Napisati decimalne brojeve jedan ispod drugog tako da znamenke iste težine budu u istom vertikalnom stupcu decimalne točke su pritom jedna ispod druge. Ako kod nekih brojeva lijevo ili desno nedostaju znamenke, zamisliti da su na tim mjestima zapisane nule. Zatim zbrajati znamenke u istom stupcu i pribrajati im eventualni prijenos s nižeg mjesta. Počinje se zbrajanjem zdesna brojevnim mjestom najmanje težine i smatra se da je tu prijenos jednak nuli. Postupak se zaustavlja kada se dosegne krajnji lijevi stupac. Iz ovoga primjera svojstvo algoritma predstavlja se kao decimalni broj. Rezultati su

naravno decimalni brojevi koji su dobiveni zbrajanjem, s naznakom na više rezultata odnosno više output parametara.[8]

Ovim primjerom daje se naznaka na sam algoritam koji mora biti sastavljen od konačnog broja koraka koji ukazuju na zadani slijed operacija koje treba izvršiti nad početnim objektima odnosno parametrima. Svaki korak opisan je određenom instrukcijom, te izvršavanje algoritma naziva se algoritamskim procesom. Tijekom vršenja algoritamskog procesa može doći do pojave međurezultata kao dodatna pojava. Za obavljanje algoritma potreban je izvoditelj algoritma, koji razumije algoritam i zna točno obaviti svaki korak algoritma. Trajanje algoritamskog procesa određeno je brzinom kojom izvoditelj obavlja korake algoritma. Ovisno o specijaliziranosti određeni algoritmi mogu se primijeniti samo s ovisnošću na određene početne parametre, dok pojedini dopuštaju različite vrste ulaznih parametara, sve ovisi kako je algoritam definiran od strane osobe koja piše algoritam. Kao primjer za pripremanje jela naveden je u smislu specijaliziranog algoritma dok je primjer decimalnih brojeva naveden kao algoritam u svrhu općenitog algoritma, jer dozvoljava zbrajanje općenito bilo kojih decimalnih brojeva. Algoritam je uporabljiv ako se za bilo koji član iz početne klase objekata može dobiti rezultat iz klase dozvoljenih završnih objekata u konačnom vremenu, tj. uz konačni broj koraka u algoritamskom procesu, odnosno algoritam je neuporabljiv ako se njegovo izvođenje ne može završiti u konačnom vremenu ili ako se u nekom koraku pojavi prepreka koja onemogućava dovršenje koraka.[4][8]

Treba naglasiti da se korištenje algoritama kao naziv i kao operacija sve do sredine 20. stoljeća, odnosno tijekom drugog razdoblja korištenja algoritama, koristi samo od strane matematičara. Do sredine 20. stoljeća bilo je sve povezano sa samim računskim dijelom, tada se događa revolucija u čovječanstvu odnosno pojavljuje se računalo po prvi puta. Time se pojam algoritama preslikava u računarstvu, s naglaskom na sve grane ove znanstvene discipline. No, iako je računarstvo prvo počelo upotrebljavati pojam algoritma postoje i druge grane koje su tada počeli koristiti sinonim algoritma. Tada se za pojam algoritma postavlja značenje koje glasi „Pravilo za postizanje željenog rezultata“.

Matematičari i matematika kao znanstvena disciplina i dalje koristi algoritme u svrhu izučavanja na sustavan teorijski način. Ovaj dio teorijske matematike sustavno izučava algoritme na način da se proučavaju mogući oblici algoritama te proučava svojstva istih. No, treba naznačiti da se izučavanje algoritama bavi disciplinom koja utvrđuje postavljena pravila ka nekom cilju što je i sama svrha algoritma. Izučavaju se algoritmi u cjelokupnom obliku kao skup jasno utvrđenih pravila za dostizanje određenog rezultata.

Neizostavan dio povijesti upravo je razvoj računala ili bolje rečeno nagli razvoj računala, gdje su algoritmi dosegnuli svoj vrhunac i pri tome se čovjek prilagodio rješavanju problematike i samih matematičkih zadataka uz pomoć računala. Kako bi računalo imalo svojstvo rješavanja određene problematike ili određenih potreba ka konačnom cilju potrebno je prikazati cjelokupan postupak kroz nekoliko koraka (ovaj naputak također je jedan algoritam):

- Formuliranje problematike
- Matematički oblik problematike
- Izrada algoritma
- Programiranje uz pomoć određenog programskog jezika
- Izrada testnog primjerka
- Testiranje
- Konačan rezultat i analiza

Svaki od navedenih koraka zahtjeva određena znanja te stručni kadar uz zadane tehnike rješavanja problema. Rad tako opisuje određenu problematiku kod matematičke discipline djeljivosti.
Algoritam ima i zadanu definiciju koja se koristi danas kako bi se pojasnilo što je zapravo algoritam, tako definicija glasi: „Algoritam je skup pravila koja opisuju rješavanje nekog problema u konačnom broju koraka“.[2]

Svaki algoritam ima i pojedine osobine:

- Konačnost (svaki algoritam završava nakon zadanog broja koraka)
- Definiranost (svaki korak prilikom izvršavanja algoritma mora biti precizno definiran – svaki segment tijekom izvođenja unaprijed je predefiniran i nedvosmisleno specificiran)
- Ulaz (prilikom izvršavanja algoritma mogu se unijeti zadane ulazne vrijednosti – što je bitno kod kreiranja zadanih algoritama poput algoritma za ispitivanje djeljivosti)
- Izlaz (svaki algoritam mora imati izlaznu veličinu kao rezultat – može biti i više izlaznih veličina)

- Efikasnost (svaki algoritam mora biti efikasan u konačnici te dati traženi rezultat)

Kako bi svaki algoritam imao određenu korisnost odnosno efikasnost on mora biti formiran uz pomoć zadanih načina predstavljanja algoritama.
Formiranje algoritma smatra se najkreativnijim dijelom izrade samoga algoritma jer se samim formiranjem objedinjuje mnogo struktura kojim se pokušava pronaći zadano rješenje za problematiku, formiranje algoritma izvodi se uz pomoć:

- Prirodnog jezika
- Dijagrama toka
- Pseudo koda (koji je u ovom radu najvažniji)

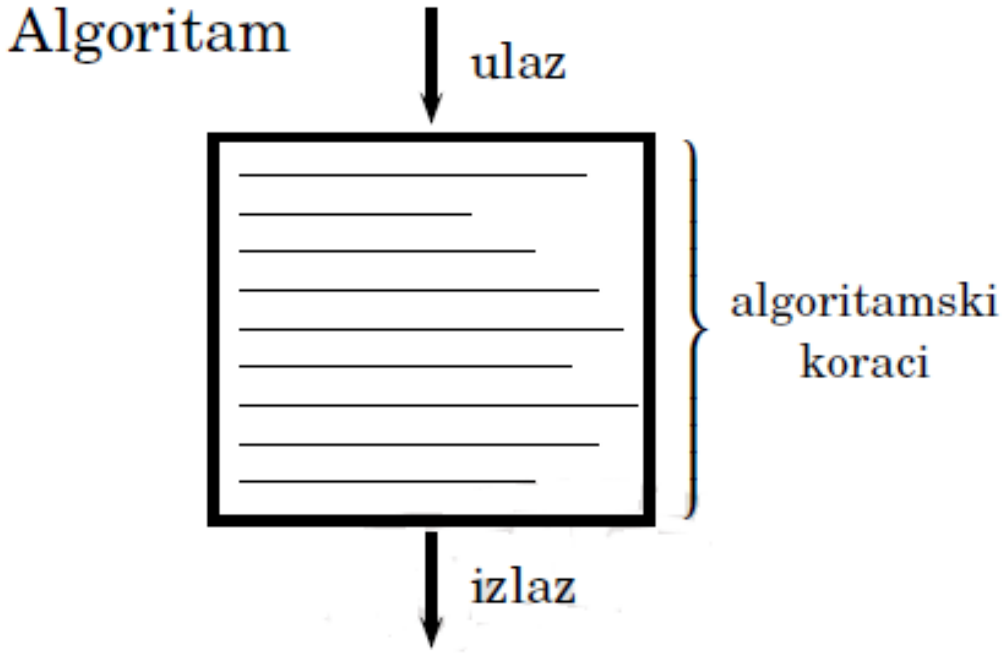
Prilikom definicije zadanih formiranja potrebno je naglasiti da u konačnici svi načini moraju dati konkretan rezultat koji dakako mora biti jednak, no sve ovisi o složenosti i disciplini problematike. Ako se primjerice radi o samoj djeljivosti zadanih brojeva **prirodni jezik** funkcionira na način da se cjelokupan proces vrši na prirodan način što dovodi računalo u probleme jer se riječima mogu definirati određeni koraci prilikom unosa, no računalo samo po sebi sintaksu prirodnog jezika ne razumije stoga je potrebna računalna sintaksa koja je preduvjet svakog programskog jezika. Za razliku od prirodnog jezika **dijagram toka** koristi određene simbole odnosno grafičke simbole koji su povezani određenim usmjerenim linijama jer se naravno radi o pojedinim koracima koji su zadani redoslijedom. Ova tehnika poprilično je popularna jer je mnogo lakša od samoga pseudo koda kojega izvršavaju sami programeri, no isto tako uz pomoć simbolike iz dijagrama toka postoje mogućnosti prenošenja dijagrama toka u programski kod što je i dalje najsigurnije izvršavati ljudskom rukom jer ako računalo izvršava ovaj postupak može doći do zadane problematike. Time dolazi u opticaj najvažniji način formiranja algoritma, a to je pseudo kod.[1][8]

Pseudo kod je tehnika za formiranje algoritma koja je slična prirodnom jeziku, no za razliku od prirodnog jezika pseudo kod koristi unaprijed preciznu sintaksu, a pošto se radi o riječima koje su unaprijed definirane i značenje je onda lakše za izvršiti, odnosno semantika je jednostavnija. Pseudo kod predstavlja poveznicu između prirodnog i programskog jezika. Poveznicu čini djelotvornost jer je jednostavan za shvaćanje s ljudske strane, ali i dovoljno jednostavan za daljnje transformacije u različite sintakse programskih jezika.

Pseudo kod funkcionira na taj način da se kroz nekoliko koraka primjerice za djeljivost predstavlja određeni niz riječi i znakova. Sljedeći primjer pojašnjava dijeljenje dvaju brojeva.

1. Korak – unos (read a, b)
2. Definiranje – c (c = a / b)
3. Konačan rezultat (display c)

Prednosti pseudo koda su jednostavnost prilikom učenja i jednostavnost za prevođenje u programski jezik zbog same sintakse.



Slika 1.2. Prikaz jednostavnog algoritma

Kada se spominje problematika, podrazumijeva se zadatak kojega je potrebno izvršiti, no u današnje vrijeme jasno je da je takav zadatak potrebno izvršiti na računalu. Problematika takve vrste definirana je preciznom specifikacijom željenog ulazno-izlaznog odnosa. Kod takve vrste definiranja potrebno je izjasniti o kakvoj se prirodi brojeva radi, jesu li ti brojevi u takvoj situaciji da mogu biti jednaki i što ako unos ne bude ispravan što algoritam onda poduzima. Stoga se koriste formalni matematički izrazi. Formalnim izrazima određena se

problematika zadaje određenim ulaznim parametrom s očekivanjima određenog izlaznog parametra, gdje oba parametra moraju zadovoljiti očekivan oblik zbog same definicije problema. Odnosno, algoritam rješava određeni problem ukoliko se na ulaz algoritma interpretira ulazni parametar za dovođenje do situacije određenog problema te na izlaz dolazi rješenje zadanog problema prema zadanim koracima. Ovim pojašnjenjem dokazuje se da je algoritam precizno naveden niz instrukcija koje računalo treba izvršiti kako bi u konačnici problem bio otklonjen ili još preciznije rečeno svrha je transformacija svake instance problema u njeno jedinstveno rješenje. Sada se već može primijetiti da se radi o jasnim pravilima i jasnom prijelazu u stupanj pojašnjenja u vidu računarstva. Stoga algoritam u vidu računarstva ima također zadane korake pojašnjenja, koji glase, ([2][4]) :

- Algoritam čini postupak od konačnog broja koraka za izvršavanje na računalu
- Algoritam je predodređen nedvosmislenim instrukcijama za izvođenje odnosno izvršavanje
- Algoritam je uvjetovan ispravnim i korektnim postupkom, što znači da se za svaku instancu problematike koja se nalazi na ulazu mora dobiti njeno odgovarajuće rješenje na izlazu, s preduvjetom da su ispunjeni svi koraci unutar algoritma.

Ranije je navedeno da pisanje algoritama dolazi od strane stručnjaka za pojedina polja, koji tijekom segmentacije određenih problema imaju na umu o kakvoj se vrsti problematike radi te su upućeni u sve segmente, čime se dotiče dio koji se naziva dizajn algoritma. Za ovaj proces ne postoji zadana formulacija već je cilj da određena osoba koja zadovoljava sve kriterije za obavljanje djela posla s pisanjem algoritma ima i određeni dašak kreativnosti i određena znanja koja vode ka rješavanju opisnog dijela instrukcija za pojedini zadani algoritam. Stoga postoje i već ranije definirane zadane metode odnosno algoritmi za rješavanje određene problematike.

Kada se govori o rješavanju problematike na računalu, uz pomoć već kreiranih algoritama ili ako se kreiraju neki jednostavniji algoritmi, postavlja se pitanje uporabe računalnih resursa. Želja korisnika je dakako smanjiti vrijeme rješavanja problematike i memorijskih zahtjeva. Time se dotiče i segment vremenske i memorijske složenosti algoritma što je dakako druga disciplina ali je potrebno navesti istu.

Prilikom uvođenja algoritama u računalni svijet u ovom radu dolazi se do najvažnije točke koja se poistovjećuje s računalnim modelom. Kada se spominje računalni model treba

naglasiti da se algoritam uvjetuje računalnim instrukcijama odnosno koracima kako je ranije navedeno. Računalni modeli čine pojednostavljeni opis kojim se izbjegava računalna složenost opisa postupka. Vrijeme i prostor odlikuju zadane parametre algoritama, no dakako treba naglasiti da se setovi instrukcija za pojedino spremanje u memoriju i tako izvršavanje ili bolje rečeno brzina izvršavanja drastično mijenja s vremenom zbog samoga napretka računalnog svijeta. Tako primjerice prilikom pojave više-jezgrenih procesora povećava se i brzina izvršavanja pojedinih instrukcijskih setova, a s povećanjem memorije povećana je i mogućnost izvršavanja kompleksnijih i većih algoritama s velikim brojem ulaznih parametara. Cilj stvaranja različitih arhitektura je i povećanje resursa za što brže izvršavanje što složenijih algoritama jer svaki unos i svaka prezentacija bilo kojeg segmenta rada na računalu interpretirana je kao određeno izvršavanje instrukcija što u konačnici dovodi do rezultata, a time cjelokupan proces izvršavanja algoritama.[8]

Kako bi se ti koraci izrazili na najefikasniji i najpreciznji način stvaraju se programski jezici koji su predefinirani iz pseudo kodova. Postupak pisanja algoritma često je uvjetovan idejama koje znaju biti komplicirane, stoga je potrebno i pojednostavljeno izraziti zadane instrukcije. Također je potrebno određene specifikacije odgovarajućeg postupka prikazati uz pomoć niza instrukcija kojima se daju naredbe računalu za izvršavanje. Tako dolazi do dijela kada se spominju algoritmi za rješavanje određenog matematičkog problema te uvjet za programsku konstrukciju tih instrukcija. Ono čini ukupnost matematičkih objekata za rješavanje problematike. Razlika je između kreiranja algoritma i programskog koda jer se kod programiranja određenog software-a pazi na svaki najmanji detalj zbog sveopće mogućnosti i funkcionalnosti programa, dok je kod algoritma izražena samo potreba za kreiranjem niza matematičkih objekata za koje sintaksa ima zadane simbole te nisu uvjetovani ostalim simbolima za pravilnu sintaksu. Zbog ovih navoda se algoritmi koriste u formatu pseudo koda koji dakako imaju elemente programskog jezika pa ih i računalo vidi kao takve. Pseudo kod najčešće sadrži elemente zadanih programskih jezika poput jezika C, Pascala ili Java programskog jezika. Stoga se koriste uobičajeni koncepti programskih jezika poput zadanih izraza i naredbi. U zadnjem poglavlju biti će prikazani zadani primjeri koji će definirati određeni algoritam za ispitivanje djeljivosti u zadanom programskom jeziku. No, prije svega treba definirati djeljivost kao matematičku disciplinu kako bi se predstavio zadani algoritam, te Euklidov algoritam kao pravi primjer algoritma.

3. TEOREM DJELJIVOSTI

Kako se svaki naputak tijekom izrade bazira na određenom shvaćanju matematike, algoritmi predstavljaju naputke u svrhu izvršavanja određenih matematičkih operacija. Ovaj rad predstavljen je kroz djeljivost, odnosno ispitivanje djeljivosti uz pomoć algoritma. No, treba naglasiti da je djeljivost dakako jedan od najjednostavnijih, ali i jedan od elementarno najvažnijih pojmova u matematici ili kako bi se reklo teoriji brojeva.

Prije svega treba napomenuti da postoje pojedina već unaprijed razumljiva opća pravila kada je u pitanju djeljivost. Treba naglasiti da su to pravila a ne definicije. Teoremi i dokazi će biti pojašnjeni u nastavku.[3][6]

Pravila glase:

a) Broj je djeljiv s 2 ako mu je posljednja znamenka 0, 2, 4, 6 ili 8;
b) Broj je djeljiv s 3 ako mu je zbroj znamenki djeljiv s 3;
c) Broj je djeljiv s 4 ako mu je dvoznamenkasti završetak djeljiv s 4;
d) Broj je djeljiv s 5 ako mu je posljednja znamenka 0 ili 5;
e) Broj je djeljiv sa 6 ako je djeljiv i s 2 i s 3;
f) Broj je djeljiv s 8 ako mu je troznamenkasti završetak djeljiv s 8;
g) Broj je djeljiv s 10 ako mu je posljednja znamenka 0;
h) Broj je djeljiv s 25 ako mu je troznamenkasti završetak djeljiv s 25;
i) Broj je djeljiv sa 100 ako su mu posljednje dvije znamenke 00;

Dijeljenje sa Z. Za broj $a \in Z$ kažemo da je djeljiv brojem $b \in Z$ i piše se $a : b$ ili $a \mid b$ (čita se na jedan od načina: a je djeljiv sa b, a je sadržalac broja b, a je umnožak ili višekratnik broja b, b dijeli a)

ako $\exists(q \in Z)a = b \cdot q$-

Teorem 1.

a) $a \mid b = a \mid bc$ $(\forall c \in Z)$

b) $a \mid b \wedge b \mid c \Rightarrow a \mid c$

c) $a \mid b \wedge a \mid c \Rightarrow a \mid (mb + nc)$ $(\forall m, n \in Z)$

d) $a \mid b \wedge b \mid a \Rightarrow a = \pm b$

e) $a \mid b \wedge a > 0 \wedge b > 0 \Rightarrow a \leq b$

Dokaz 1.

a) $a \mid b \Rightarrow \exists\, (\, q \in Z\,)\; b = a \cdot q \Rightarrow bc = a\,(\,cq\,) \Rightarrow a \mid bc$

b) $a \mid b \wedge b \mid c \Rightarrow \exists\, (\, q\,, s \in Z\,) b = aq \wedge c = bs \Rightarrow c = a\,(qs\,) \Rightarrow a \mid c$

c) $a \mid b \wedge a \mid c \Rightarrow \exists\, (\, q\,, s \in Z\,) b = aq \wedge c = as \Rightarrow bm + cn = maq + nas = a\,(mq + ns\,) \Rightarrow a \mid bm + cn$

d) $a \mid b \wedge b \mid a \Rightarrow \exists\, (\, q \in Z\,) b = aq \wedge b \mid a \Rightarrow qa \mid a \Rightarrow \exists\, (\, s \in Z\,) a = sqa \Rightarrow a\,(1 - sq) = 0$

dva slučaja:

(i) $a = 0 \Rightarrow b = 0$

(ii) $(\,(1 - sq = 0\,) \Rightarrow (\, q = 1 \wedge s = 1) \vee (\, q = -1 \wedge s = -1)\,) \Rightarrow b = a \vee b = -a$

e) $(\, a \mid b \wedge a > 0 \wedge b > 0\,) \Rightarrow \exists\, (\, 0 < q \in Z\,) b = aq \Rightarrow q \geq 1 \Rightarrow b \geq a$

Teorem 2.

Neka su $a_1, a_2, \ldots, \in k$ Z i $a_1 + a_2 + \ldots + a_k = 0$
Ako su svi pribrojnici osim jednog djeljivi brojem $p \in Z$, onda je i taj pribrojnik djeljiv brojem $p \in Z$

Dokaz 2.

Bez umanjenja općenitosti možemo pretpostaviti da je

$p \mid a_1 + a_2 + ... + a_{k-1}$

Na osnovu definicije 1. slijedi

$\exists (q_1, q_2, ..., q_{k-1} \in \mathrm{Z})\ a_1 = q_i p \quad (i = 1, 2, ..., k-1)$

Iz jednakosti $a_1 + a_2 + ... + a_k = 0$ i (*) slijedi

$a_k = -(a_1 + a_2 + ... + a_{k-1}) = -p \sum_{i=1}^{k-1} qi = p(-\sum_{i=1}^{k-1} qi)$

a ovo znači da je $p \mid a_k$, što je dokaz.

Teorem 3.

Neka su $a \in \mathrm{Z}$ i $b \in \mathrm{N}$. Tada postoje jedinstveni brojevi $q, r \in \mathrm{Z}$ koji vrijede:

$a = bq + r\ 0 \leq r < b$

Dokaz 3.

pregledom skupa $S = \{ a - bk : k \in \mathrm{Z}\}$ i skup $T \subset S$, $T = \{w \in S : w \geq 0\}$ (skup svih ne negativnih brojeva skupa S).

Skup T nije prazan, jer je $a - bk \geq 0$ ako je $k \leq \frac{b}{a}$, $k \in \mathrm{Z}$.

Kako je skup nenegativnih cijelih brojeva dobro uređen skup*, njegov svaki podskup, pa time i skup T, ima najmanji element, primjerice x, pa je tada $x = a - bk \geq 0$ (jer pripada skupu T).

Iz čega proizlazi da je $a = x + bk$.

Uzima se da je $x = r$ i $k = q$. Tada je

(*) $a = qk + r$.

Ovim putem se dokazuje egzistencija brojeva $q, r \in \mathrm{Z}$ za koje vrijedi $a = qk + r$

Dokaz je da je i $r < b$

Ako bi bilo $r \geq b$ ($r \geq 0$) tada je $r > r - b$, a kako je iz (*) $r = a - bq$ slijedi

$r > a - bq - b = a - (q + 1)b \geq 0$

Dakle, r je najmanji element skupa T, a uz pretpostavku $r \geq b$ zna se da postoji broj ($= a - (q + 1)b$) iz skupa T (jer je ne negativan a pripada skupu S) koji je manji od r, pa iz $r \geq b$ slijedi da r nije najmanji element skupa T, što je kontradikcija. Zaključak je $r < b$.

Dakle, postoji $q, r \in$ Z da je $a = bq + r \;\; 0 \leq r < b$.

Dokaz jedinstvenosti brojeva $q, r \in$ Z.

Pretpostavka $q, r \in Z$ nisu jedinstveni, što znači da postoje q_1 , $r_1 \in$ Z, $q \neq q_1$ i $r \neq r_1$ takvi da vrijedi:

$a = bq + r \;\; 0 \leq r < b$

$a = bq_1 + r_1 \;\; 0 \leq r_1 < b$

Oduzimanje jednakosti rezultira:

$0 = b\,(q - q_1) + (r - r_1) \Rightarrow r_1 - r = b\,(q - q_1) \Rightarrow b \mid r_1 - r$

Kako je $0 \leq r < b$ i $0 \leq r_1 < b$ odnosno $r - r_1 < b$ slijedi da je $r_1 - r = 0$, odnosno $r = r_1$. iz čega proizlazi $(bq + r = bq_1 + r_1 \wedge r = r_1) \Rightarrow q = q_1$ (što je kontradikcija s pretpostavkom). Dakle, brojevi $q, r \in$Z su jedinstveni.

Broj q nazivamo količnikom a broj r ostatkom pri dijeljenju broja a brojem b. $b \mid a \Leftrightarrow r = 0$

ako su $a, b \in Z$ da postoje jedinstveni brojevi $q, r \in$ Z takvi da je $0 \leq r < b$.

4. EUKLIDOV ALGORITAM

Kao jedan od najstarijih algoritama često se spominje Euklidov algoritam, odnosno spominje se još u Euklidovim elementima prema kojemu je i dobio ime. Osnovna svrha ovoga algoritma je određivanje najvećeg zajedničkog djelitelja. Treba naglasiti da Euklidov algoritam uz određivanje najvećeg zajedničkog djelitelja ima i mogućnost rješavanja diofantskih jednadžbi te razvoj racionalnog broja u verižni razlomak. Često se Euklidov algoritam poistovjećuje s teoremom o dijeljenju s ostatkom.[5][7]

Euklidov algoritam zasnovan je zadanom principu:

- Najveći zajednički djelitelj (NZD) dvaju brojeva se ne mijenja ukoliko se manji broj oduzme od većega, prema čemu se odredi najveći zajednički djelitelj novog broja koji je manji od prethodna dva.

Primjer 1.:

Broj 21 je najveći zajednički djelitelj za brojeve 252 i 105

Dokaz :

252 : 21 = 12 i 105 : 21 = 5

Kada se od 252 oduzme 105 odnosno 252 – 105 = 147, dobije se da je 21 ponovno najveći zajednički djelitelj brojeva 147 i 105, no kada se od 147 oduzme 105 odnosno 147 – 105 = 42, tada je najmanji zajednički djelitelj brojeva 105 i 42 broj 7. S obzirom da se svaki puta veći od dva polazna broja smanjuje, dolazi do ponavljanja postupka gdje se dobiju manji brojevi a time i manji zajednički djelitelji sve dok se jedan od njih ne svede na 0.

Treba naglasiti da je Euklidov algoritam iterativne prirode, što predstavlja rješavanje i dobivanje krajnjeg rezultata kroz niz koraka, prilikom čega međurezultat pojedinih koraka koristi se u svakom sljedećem, ova vrsta algoritma spomenuta je u početnom dijelu stoga je ovo konkretan primjer navedenoga algoritma s određenim među rezultatima.[7][9]

Ukoliko k predstavlja cijeli broj kojim se označava korak algoritma s početkom u 0, prvi korak predstavljen je s jednakošću jednakom 0, $k = 0$, drugom $k = 1$, itd.

Svaki korak započinje s određenim ostatkom nakon prvobitnog koraka, svaki ostatak mora biti nenegativan, odnosno r_{k-1} i r_{k-2}. Algoritam kroz svoje korake omogućava da se svaki ostatak kontinuirano smanjuje, odnosno r_{k-1} mora biti manji od svakog prethodnika r_{k-2}.

Kao rezultat k-tog koraka uzima se u obzir količnik q_k i ostatak r_k s obzirom da vrijedi jednakost:

$$r_{k-2} = q_k\, r_{k-1} + r_k$$

gde je

$$r_k < r_{k-1}$$

Odnosno umnožak manjeg broja r_{k-1} oduzima se od većega broja r_{k-2} sve dok ostatak ne bude manji od r_{k-1}.

Prvi korak koji je prezentiran kao $k = 0$, ostatak r_{-2} i r_{-1} su jednaki, a to su upravo brojevi za koje se traži najveći zajednički djelitelj.

Drugi korak $k = 1$, ostatak postaje jednak b i ostaku početnog koraka r_0...

Primjer 2.:

(248 i 92)

248 = 92 2 + 64
92 = 64 * 1 + 28
64 = 28 * 2 + 8
28 = 8 * 3 + 4

8 = 4 * 2

Iz čega se može zaključiti da je (248, 92) = 4

Dokaz

povratna supstitucija:

4 = 28 - 8 * 3 = 28 - (64 - 28 * 2) * 3 = 7 * 28 - 3 * 64
= 7 * (92 - 64) - 3 * 64 = 7 * 92 - 10 * 64

= 7 * 92 - 10 * (248 - 92 * 2) = 27 * 92 - 10 * 248.

Ovim pretpostavkama može se predstaviti algoritam uz pomoć niza jendakosti, koji glasi:

$$a = q_0\, b + r_0$$

$$b = q_1\, r_0 + r_1$$

$$r_0 = q_2\, r_1 + r_2$$

$$r_1 = q_3\, r_2 + r_3$$

…

Ako dođe do slučaja da je u prvom koraku a manji od b, algoritam ima zadatak da zamijeni brojeve, odnosno za $a < b$, početni količnik q_0 jednak je 0, a ostatak r_0 jednak je a. Na taj način, r_k je manji od svog prethodnika r_{k-1} za sve $k \geq 0$.

Prema zadanim koracima odnosno ako se kroz svaki korak ostatak smanjuje te kako ne postoji mogućnost da budu negativni, ostatak r_n u nekom trenutku mora postati jednak 0, čime se dobije konačan rezultat i time izvršen algoritam prema zadanom naputku. Posljednji ostatak r_{n-1} koji je različiti od 0 čini najvećeg zajedničkog djelitelja brojeva a i b. Treba naglasiti da n ne može biti beskonačan jer prema zakonitostima između 0 i r_0 postoji konačan broj cijelih brojeva, koji nisu negativni.

Nastavak predstavlja oblik Euklidovog teorema u obliku pseudo koda, čime se dolazi do poglavlja primjera algoritma (programski). Kao temelj tog poglavlja uzima se ovaj primjer kao konkretan pseudo kod koji će služiti kao temelj za programski oblik algoritma u zadanim programskim jezicima.[5][7]

```
function nzd(a, b)
    if a = 0
       return b
    while b ≠ 0
        if a > b
           a := a − b
        else
           b := b − a
    return a
```

5. PRIMJER ALGORITMA (PROGRAMSKI)

Ovo poglavlje predstavit će pojedine algoritme u svrhu ispitivanja djeljivosti. Primjer algoritma za ispitivanje djeljivosti (Euklidov algoritam) biti će prikazan kao programski kod u programskom jeziku C++, Python i Java.

Primjer 1. (C++)

```
#include
<iostream>
using namespace std;


float nzd(int a, int b) {
    cout<<" upiši 2 broja "<<endl;
    cout<<"a: ";
    cin>>a;
    cout<<": ";
    cin>>b;
    while ( a!= b ) {
        if (a>b)
            a -= b;
        else
            b -= a;
    }
    return a;


}
int main()
{
 int a,b;
 cout<<"Najveci zajednicki djelitelj: "<<nzd(a,b)<<endl;
 system ("pause");
 return 0;
}
```

Drugi primjer zasniva se na korištenju programskog jezika Python, te isto predstavlja Euklidov algoritam za ispitivanje djeljivosti.

Primjer 2. (Python)

```
a=int(input("Unesite
a: "))
b=int(input("Unesite b: "))

if a<b:
    temp=a
    a=b
    b=temp

while b != 0:
    ostatak=a%b
    a=b
    b=ostatak

print("Najveća zajednička mjera:",a)
```

Treći primjer predstavlja algoritam za ispitivanje djeljivosti kreiran uz pomoć programskog jezika Java.

Primjer 3. (Java)

```
public
class
Euclid
{
    // recursive implementation
    public static int gcd(int p, int q) {
        if (q == 0) return p;
        else return gcd(q, p % q);
    }

    // non-recursive implementation
    public static int gcd2(int p, int q) {
        while (q != 0) {
            int temp = q;
            q = p % q;
            p = temp;
        }
        return p;
    }

    public static void main(String[] args) {
        int p = Integer.parseInt(args[0]);
        int q = Integer.parseInt(args[1]);
        int d  = gcd(p, q);
        int d2 = gcd2(p, q);
        StdOut.println("gcd(" + p + ", " + q + ") = " + d);
        StdOut.println("gcd(" + p + ", " + q + ") = " + d2);
    }
}
```

6. ZAKLJUČAK

Prilikom izrade završnoga rada, ukazuje se na problematiku izrade samoga algoritma. Izrada algoritma može biti složen postupak ovisno o zadanoj problematici koja je potrebna za rješavanje. Algoritam je po mnogočemu zaslužan za različite svakodnevne procedure. Čovjek nije ni svjestan koliko je zapravo okružen algoritmima u različitim oblicima, no usprkos tome pravi algoritmi koji se zasnivaju na određenim matematičkim disciplinama svoju pravu korisnost iskazuju uz pomoć korištenja računala, gdje se izvršavaju algoritmi kojima je cilj rješavanje problematike zadataka.

Završni rad predstavlja algoritam te pojašnjava njegovu svrhu kao naputak za rješavanje pojedinih operacija zadanim koracima. Predstavljeno je na koji način algoritam izvršava pojedine operacije kako u pravom životu tako i u matematičkoj teoriji. Algoritam kao cjelina predstavlja sva važnija pojašnjenja te je interpretiran na način kako bi čitatelji shvatili svrhu i funkcionalnost algoritma. Kako bi sam algoritam za ispitivanje djeljivosti imao smisla, opisana je teorija djeljivosti kao matematička podloga potkrijepljena zadanim teoremima i definicijama uz prateće dokaze. Teorem kao teorem zadan je određenim preduvjetima i time je zasnovan temelj za pojašnjenje algoritma za ispitivanje djeljivosti.

Algoritam za ispitivanje djeljivosti ne bi imao smisla bez Euklidovog algoritma koji čini osnovu algoritama. Euklidov algoritam interpretiran je uz pomoć pseudo koda koji je sastavni dio jezika algoritama.Kako bi sve bilo informatički potkrijepljeno predstavljeni su i algoritmi kroz programske jezike C++, Python te Java.

Svakodnevica je pred uvjetovana algoritmima, od samoga algoritma kao osnovnog u svrhu pojašnjenja s opisom prelaska preko ceste do složenih algoritama za izvođenje niza matematičkih operacija, svaki algoritam ima zadani naputak odnosno niz instrukcija koje je potrebno izvesti kako bi se dobio konačan rezultat i cilj te samu svrhu algoritma.

LITERATURA

[1] Algoritmi, Pojam algoritma i osnovna svojstva algoritama, Predavanje, Strojarski fakultet Slavonski Brod, članak dostupan na: https://element.hr/artikli/file/1135, pristup: 10.07.2015.

[2] Algoritmi, Članak s interneta bez autora, dostupno na: http://www.gimnazijabihac.edu.ba/doc/info2/ALGORITMI_1.pdf , pristup: 11.07.2015.

[3] B. Ibrahimpašić, S. Ibrahimpašić, D. Kovačević, A. Šehanović; Pravila djeljivosti, Osječki matematički list 11 (107-112), 2011., pristup: 18.07.2015.

[4] D. Živković; Uvod u algoritme i strukture podataka, Univerzitet Singidunum, Beograd, 2010., pristup: 12.07.2015.

[5] M. Đumić, M. Jukić Bokun; Primjene Euklidovog algoritma, Osječki matematički list 13 (121-137), 2013., pristup: 23.07.2015.

[6] M. Smajilović; Teorija brojeva, Skripta sa predavanja, Univerzitet u Tuzli, Prirodno-matematički fakultet, 2004., pristup: 20.07.2015.

[7] R. Dimitrijević; Euklidov algoritam, Prirodno-matematički fakultet, Univerzitet u Nišu, Matematika i informatika, 2008., pristup: 30.07.2015.

[8] V. Krčadinec; Osnove algoritama, Sveučilište u Zagrebu, Prirodoslovno-matematički fakultet, Matematički odsjek, 2013., pristup: 10.07.2015.

[9] V. Stojanović; Euklidov algoritam za nalaženje najmanjeg zajedničkog delioca dva broja, Elektronski fakultet Niš, Katedra za Računarstvo, Algoritmi i arhitekture specijalizovanih računarskih sistema, 2010., pristup 27.07.2015.

SAŽETAK

Završni rad predstavlja opis algoritma u svrhu ispitivanja djeljivosti. Koncipiran je na način da opisuje što su to algoritmi te opisuju svojstva te funkcionalnosti algoritama. Algoritam je predstavljen kao naputak za izvršavanje pojedinih operacija kroz zadani niz koraka. Matematička podloga opisuje definicije i teoreme djeljivosti što čini temelj za uporabu algoritama u svrhu ispitivanja djeljivosti. Završni rad popraćen je detaljnim opisom algoritma i same strukture algoritm i Euklidovog algoritma za ispitivanje djeljivosti, koji je interpretiran kao pseudo kod. Uz prikaz samoga pseudo koda za shvaćanje funkcionalnosti algoritma za ispitivanje djeljivosti, prikazan je algoritam s informatičke perspektive kao zadani program napisan u programskim jezicima. Kao primjer koristili su se programski jezici C++, Python te Java. Rad je interpretiran kroz nekoliko poglavlja s opisno definiranim segmentima radi lakšega obrazloženja i shvaćanja samih algoritama i matematičke teorije djeljivosti kao elementarne. Prikazane su i operacije kroz određene korake prilikom izvršavanja općenitog algoritma te time obuhvaćen niz pravila koje je potrebno poštivati kako bi se kreirao algoritam koji bi u konačnici riješio problematiku kroz niz zadanih instrukcija odnosno koraka.

ABSTRACT

This final work is a description of the algorithm for the purpose of divisibility tests. The final work is designed in a way that describes what are a algorithm and describe the properties and functionality of algorithms. The algorithm is presented as a guide to the performance of certain operations through the default number of steps. Mathematical foundation describes definitions and theorems of divisibility which forms are the basis for the use of algorithms in order to divisibility tests. The final work is accompanied by a detailed description of the algorithm and the structure of the algorithm, but also the Euclidean algorithm for testing divisibility, which is interpreted as a pseudo code. With the display of the pseudo code for understanding the functionality algorithm for testing divisibility, the algorithm are shown from the computing perspective as the default algorithm written in programming languages. As an example of using the programming languages C ++, Python and Java. The work was interpreted through several chapters with descriptive defined segments for easy explanation and understanding of algorithms and mathematical theory of divisibility as elementary. Showing and operations through a number of steps during the execution of a general algorithm and thus comprised a set of rules that must be observed in order to create an algorithm that would ultimately resolve the problem through a series of instruction sets by steps.

ŽIVOTOPIS:

Danijel Đurđević rođen je 29. lipnja 1992. u Bjelovaru. Završio je osnovnu školu u Grubišnom Polju, te srednju školu Bartol Kašić smjer Tehničar za računalstvo. Upisao je Elektrotehnički fakultet u Osijeku 2011.

Tijekom studija je bio zaposlen u prodajnom centru Studio moderna i telekomunikacijskoj firmi Hrvatski Telekom . Student ima vrlo dobro poznavanje engleskog i njemačkog jezika.