UNIVERZITET U NIŠU

ELEKTRONSKI FAKULTET

**ZAVRŠNI RAD**

**Implementacija i evaluacija algoritama za testiranje prostih brojeva**

**Nikola Stevanović**

Niš, februar 2018. god.

UNIVERZITET U NIŠU

ELEKTRONSKI FAKULTET

**ZAVRŠNI RAD**

**Implementacija i evaluacija algoritama za testiranje prostih brojeva**

**Zadatak:** Proučiti algoritme za testiranje prostih brojeva. Uočiti razliku izmedju determinističkih i nedeterminističkih algoritama. Implementirati sledeće algoritme za testiranje prostih brojeva:

* Deterministički
  + Naive approach
  + Elliptic curve primality proving – Atkin’s ECPP
* Nedeterministički
  + Miller-Rabin
  + Solovay-Strassen

Izvršiti evaluaciju rada algoritama nad 32-bitnim i 64-bitnim brojevima.

#### Mentor: Kandidat:

Prof. dr Vladimir Ćirić Nikola Stevanović

broj indeksa: 14525

Komisija za ocenu i odbranu završnog rada:

1. Datum prijave završnog rada:
2. Datum predaje završnog rada:
3. Datum odbrane završnog rada:

**Sadržaj**

[1. Uvod 4](#_Toc505993571)

[2. Prosti brojevi 6](#_Toc505993572)

[**2.1.** **Prosti brojevi i osnovni stav aritmetike** 6](#_Toc505993573)

[**2.2.** **Pseudoprosti brojevi** 7](#_Toc505993574)

[**2.3.** **Gustoća prostih brojeva** 7](#_Toc505993575)

[**2.4.** **Značaj prostih brojeva za asimetričnu kriptografiju** 8](#_Toc505993576)

[3. Testovi za ispitivanje prostih brojeva 9](#_Toc505993577)

[**3.1** **Vremenska složenost i klase složenosti** 10](#_Toc505993578)

[**3.2** **Nedeterministički testovi** 10](#_Toc505993579)

[**3.3** **Deterministički testovi** 11](#_Toc505993580)

# **Uvod**

“Matematičari su do današnjih dana uzaludno pokušavali da otkriju neki red u nizu prostih brojeva i imamo razloga da verujemo da je to tajna u koju um nikada neće prodreti” – Ojler, (Havil 2003, str. 163).

Prosti brojevi su fascinirali ljude veoma dugo. Iako se čini da je pojam prostog broja prilično jednostavan i lako razumljiv, prosti brojevi su zadavali mnogo muke matematičarima u prošlosti.

Prost broj je pozitivan ceo broj p > 1 koji nema pozitivne cele delitelje osim 1 i samog broja p. Pozitivni celi brojevi, osim broja 1[[1]](#footnote-1), koji nisu prosti se zovu složenim brojevima. Možemo da uocimo da je 2 jedini paran prost broj.

Drevni Egipćani su počeli da se pitaju kako da pronađu proste brojeve. Jedan od najintuitivnijih načina za pronalaženje prostih brojeva je isključivanje svih brojeva koji ne mogu biti prosti. Ovaj metod je poznat kao Eratostenovo sito, koji je dobio ime po starogrčkom matematičaru Eratostenu. On jednostavno isključuje sve brojeve koji su deljivi sa unapred poznatim prostim brojevima ( 2,3,5,7... √*p*), gde je *p* granica do koje želimo testiramo brojeve. Ovaj metod sa sigurnošću daje rezultat da li je broj prost ili ne, ali je takođe potrebno puno vremena za izračunavanje kada je *p* veliki broj.

Mnogo godina kasnije, napravljena su važna otkrića o identitetima prostih brojeva. Jedno od otkrića poznato je i kao mala Fermaova teorema: , gde je *a* pozitivan ceo broj i *p* prost broj. Zbog činjenice da ova teorema važi za svaki prost broj, može se koristiti da isključi brojeve iz skupa potencijalnih prostih brojeva. Ovaj test ne može sa sigurnošću reći da li je broj prost. Postoji više nedeterminističkih testova, kao što je Miller-Rabin. Ovaj test se takođe oslanja na identitet koji važi za proste brojeve, ali ovaj identitet takođe važi i za neke složene brojeve poznatije kao pseudoprosti brojevi (poglavlje 2.2). To znači da je tačnost testa može biti velika, ali nikada 100%. Isto važi i za Solovay-Strassen test, koji se zasniva na teoremi koju je dokazao Euler.

U 20-om veku su se pojavilo nekoliko atraktivnih rešenja. 30-ih godina prošlog veka D.H.Lehmer je poboljšao test koji je E.Lucas osmislio 1878. Tako je nastao Lucas-Lehmer test. Iako je ovaj test deterministički, on funkcioniše za malu grupu brojeva tzv. Mersenovi brojevi Mp = , gde je *p* poznat prost broj.

50-ak godina kasnije pojavio se algoritam zasnovan na eliptičnim krivama. Iako u početku test nije bio deterministički, bio je veoma često korišćen za potvrdu da li je broj prost ili ne. Kasnije je poboljšan, tako da može da potvrdi da li je broj prost. Međutim to je znatno usporilo algoritam i zbog toga je postao manje atraktivan za praktičnu primenu.

Međutim u 21. veku došlo je do velikog pomaka. 2002 je objavljen document “Primes is in P” u kojem je predstavljen novi algoritam za testiranje prostih brojeva. Algoritam je nazvan po njegovim stvaraocima: Agrawal, Kayal and Saxena. Za razliku od testova pomenutih ranije u tekstu, ovaj algoritam nam sa sigurnošću moze reći da li je broj *p* prost.

Danas algoritmi za testiranje prostih brojeva imaju široku primenu u računarstvu, tačnije u kriptografiji. RSA sistem je jedan od podsticaja zbog čega je brzo i efikasno pronalaženje prostih brojeva bitno u kriptografiji. RSA bira dva različita velika prosta broja (preko 100 decimalnih cifara) *p* i *q* i izračunava njihov prizvod *n = pq*. Brojevi *p* i *q* se čuvaju u tajnosti, dok je *n* deo javnog ključa.

Cilj ovog rada je proučavanje algoritama za testiranje prostih brojeva i njihova implementacija. U radu će najpre biti opisani prosti brojevi, nakon toga ce biti reči o podeli algoritama za testiranje prostih brojeva na dve grupe:

* Nedeterminističke
* Determinističke

Zatim će biti opisana po dva predstavnika obe grupe algoritama i biće data njihova implementacija. Na kraju ćemo izvršiti evaluaciju i poređenje algoritama što je značajno za odabir algoritma koji će kasnije biti integrisan sa kriptografskim sistemima.

Nakon uvoda, u drugom poglavlju, biće objašnjen pojam prostog broja i biće date njihove karakteristike, kao i njihov značaj u kriptografiji.

U trećem poglavlju biće reči o algoritmima za testiranje prostih brojeva i biće data njihova podela u dve grupe kao i opis sveke od grupa.

U sledećem poglavlju, četvrtom, biće detaljno opisana po dva algoritma iz obe grupe. Za nedeterminističke to će biti Miller-Rabin i Solovay-Strassen, a za determinističke Naivni algoritam tj. algoritam koji prati definiciju prostih brojeva i Atkinov algoritam zasnovan na eliptičnim krivama. U ovom poglavlju prikazaćemo i implementaciju svakog algoritma u jeziku C#.

U petom, pretposlednjem poglavlju, biće data evaluacija svakog algoritama i njihovo poređenje.

Zaključak će biti dat u poslednjem, šestom poglavlju.

# **Prosti brojevi**

## **Prosti brojevi i osnovni stav aritmetike**

Definiciju prostog broja uveo je grčki matematičar Euklid, koji je živeo na prelazu iz 4. u 3. vek pre nove ere. Posmatrajmo broj *a* ∈ N, *a* > 1. Svaki prirodni broj a uvek ima dva delitelja: 1 i samog sebe. Te delioce zovemo trivijalnim deljiteljima.

**Definicija 2.1.:** Za svaki prirodni broj *p* > 1 kažemo da je prost broj ukoliko ima samo trivijalne delitelje.

Delitelje broja *a* ∈ N zovemo još i faktorima broja *a*. Faktorizaciju prirodnog broja *a* prikazujemo u obliku *a = bc*, gde su *b,c* ∈ N. Ako je delitelj *b* prost broj, zovemo ga jos i prostim faktorom broja *a*. U svrhu dokazivanja osnovne teoreme aritmetike razmotrićemo nekoliko teorema koje ćemo iskoristiti kasnije.

**Teorema 2.1.1.** Neka je *a* prirodan broj, *a* > 1 ili je prost ili je proizvod prostih brojeva.

**Teorema 2.1.2.** Ako prost broj *p* nije deljiv sa *a* onda je (*a,p*) = 1.

**Teorema 2.1.3.** Ako *p* deli *ab*, onda *p* deli *a* ili *p* deli *b*. Opštije ako *p* deli *a1a2a3...aN*, tada *p* deli barem jedan od brojeva *a1,a2,a3,...,aN*.

**Definicija 2.2.:** (Osnovni stav aritmetike) Svaki prirodan broj *a* > 1 može se predstaviti kao proizvod prostih brojeva na jedinstven način.

Ova teorema je ključni stav tzv. Multiplikativne teorije brojeva. U faktorizaciji broja *a* neki prost broj se može pojaviti kao faktor više puta, recimo 24 = 2 x 2 x 2 x 3. Ako su *p*1,...,*p*K svi različiti prosti faktori broja *a*, onda se *a* može predstaviti kao

*a* =

gde su α1, α2,..., αk jednoznačno određeni prirodni brojevi. Ovo se naziva kanonska reprezentacija prirodnog broja *a*, i veoma se često koristi u teoriji brojeva.

Odgovor na pitanje koliko ima prostih brojeva znali su još starogrčki matematičari. Euklid u IX knjizi svojih „Elemenata“ daje sledeći dokaz činjenice da je skup prostih brojeva beskonačan.

Euklidova teorema: Ne postoji naveći prost broj.

## **Pseudoprosti brojevi**

Specijalna klasa složenih brojeva su tzv. pseudoprosti brojevi. Njihovo definisanje je neophodno za razumevanje nedeterminističkih testova. Oni predstavljaju složene brojeve koji prolaze test ili sekvence testova za ispitivanje da li je zadati broj prost, a koji inače padaju za većinu složenih brojeva.

Definišimo najpre *malu Fermaovu teoremu* koja je neophodna za razumevanje određenih klasa pseudoprostih brojeva.

**Teorema 2.2.1 (mala Fermaova).** Neka je *p* prost broj i *a* ∈ N takav da p ne deli a. Tada je

Karmajklovi brojevi su neparni složeni Fermaovi pseudoprosti brojevi za svaku od baza.

**Definicija 2.3.** Složen ceo broj *n* nazivamo *Karmajklovim brojem*, ako n deli za svako a ∈ N.

Još jedna klasa pseudoprostih brojeva su jaki pseudoprosti brojevi.

**Definicija 2.4.** *Jak pseudoprost broj* za bazu *a* je neparan složen broj , gde je *d* neparno I za koji važi jedna od sledećih kongruencija:

Za neko *r = 0,1,…,s-1*.

Složen broj *n* je jak pseudoprost broj za najvise ¼ svih baza manjih od *n*. Ova činjenica se koristi kod Miller-Rabin testa.

## **Gustoća prostih brojeva**

Ne postoji pravilo po kojem možemo odrediti kada će se u zadatom nizu pojaviti prost broj. Na primer, između brojeva 1 i 10 pronalazimo četiri prosta broja, dok između brojeva 80 i 90 postoje dva prosta broja, a izmedju 2000 i 2100 jedan prost broj.

Dokazano je da prostih brojeva ima beskonačno mnogo, ali nije otkrivena formula za pronalaženje prostih brojeva pa se počelo empirijki istraživati udeo prostih brojeva u skupu N.

Krajem 18. veka Gauss i Legendre su pretpostavili da je funkcija raspodele prostih brojeva manjih ili jednakih broju *n* (označava se sa π(*n*)) približno *n/ln(n)*

**Teorema 2.3.1** (Teorema o prostim brojevima) Dobra aproksimacija π(*n*) je data kao:

,

Dakle, možemo reći da za dovoljno veliki broj *a* ∈ N koji je slučajno izabran postoji verovatnoću od *1/ln(a)* da bude prost.

## **Značaj prostih brojeva za asimetričnu kriptografiju**

RSA algoritam je jedan od najpoznatijih i najkorišćenijih asimetričnih šifraskih sistema. Sistem je dobio ime po trojici svojih pronalazača Rivest-Shamir-Adleman.

U literaturi koja se odnosi na RSA algoritme, često se predlaže da se u izboru para ključeva koriste jaki prosti brojevi p I q za generisanje n. Jaki prosti brojevi imaju određene osobine koje čine proizvod n teško faktorizovanim. Razlog za ovakav izbor je to što su neke od metoda faktorizacije, kao što su Polardove P - 1 i P + 1 metode, naročito pogodne za proste brojeve *p* za koje P – 1 ili P + 1 imaju samo male proste faktore. Jaki prosti brojevi su otporni na ovakve napade. Napredak u faktorizaciji tokom poslednjih deset godina pokazuje opravdanost upotrebe jakih prostih brojeva. Za generisanje ključeva u RSA algoritmu preporučuje se upotreba velikih prostih brojeva veličine 100 decimalnih cifara I više.

Generisanje ključa za RSA se odvija tako što se odaberu dva velika broja *p* i *q*, a zatim računamo *n = p\*q*. Nakon toga odaberemo *e* i *d* uz poštovanje nekih uslova. Postoji dobar razlog zašto *n* ima tačno dva prosta faktora. Prvi razlog je jednostavnost formule za izračunavanje ϕ*(n*) = (*p-1*)(*q-1*), a drugi je teža faktorizacija *n*, jer je poznato da je faktorizaciju teže izvršiti ukoliko broj ima manje prostih faktora. Da bi smo stekli jasniju sliku o trenutno najvećim prostim brojevima u Tabeli 1. prikazaćemo podatke sa Web stranice [\*].

|  |  |  |  |
| --- | --- | --- | --- |
|  | Prost broj | Broj cifara | Godina pronalaska |
| Najveći prost broj | 277232917-1 | 23249425 | 2018 |
| Najveći prost broj “blizanac” | 2996863034895·21290000+1 | 388342 | 2016 |
| Najveći Marsenov prost broj | 277232917-1 | 23249425 | 2018 |
| Najveći Sophie Germain prost broj | 2618163402417·21290000+1 | 388342 | 2016 |

Tabela 1. *Najveći poznati prosti brojevi*

Kod RSA algoritma za komercijalnu upotrebu se obično koriste brojevi n od 1024 bita, odnosno n ≈ 10308. Za važne potrebe obično se koristi 2048-bitno n, odnosno n ≈ 10617.

# **Testovi za ispitivanje prostih brojeva**

U ovom poglavlju govorićemo testovima, odnosno o algoritama za proveru da li je zadati broj prost. Ovi testovi odgovaraju na pitanje da li je uneti broj prost. Za razliku od faktorizacije, pomenuti testovi generalno ne vraćaju proste faktore, već samo odgovaraju na pitanje da li je uneti broj prost. Problem faktorizacije broja je značajno kompleksniji od provere da li je prost koristeći se momenutim testovima. Većina testova pokazuje da je broj prost, ali postoje i oni kao što je Miller-Rabin (4.nesto), koji pokazuju da je uneti broj složen, tako da njih možemo zvati i testovima složenosti.

Testove za proveru da li je zadati broj prost delimo na dve velike grupe:

* Nedeterministički
* Deterministički

Deterministički testovi sa potpunom sigurnošću utvrđuju da li je traženi broj prost. Za razliku od njih, nedeterministički testovi mogu potencijalno(mada sa veoma malom verovatnoćom) pogrešno identifikovati složeni broj kao prost. Nedeterministički testovi su znatno brži, dok je značaj determinističkih testova u njihovoj sigurnosti, zato se u praksi najčešće koristi njihova kombinacija. Najpre se primeni neki od nedeterminističkih testova i ako zadati broj prođe test označi se kao verovatno prost broj. Zatim pomoću nekog od determinističkih testova se može sa sigurnošću utvrditi da li je dati broj prost tj. odbaciti mogućnost da se radi o pseudoprostom boju koji može da prodje nedeterministički test, a inače je složen.

Generisanje velikih prostih brojeva ima primenu u kriptografiji. Generalni meod za njihovo generisanje je da se izabere neparni broj *n* odgovarajuće dužine, a zatim na scenu stupaju pomenuti testovi, kojima se utvrđuje da li je izabrani broj prost ili se mora pokušati sa generisanjem novog.

Pre daljeg razmatranja pojedinih testova, potrebno je nešto više reći o dve bitne karakteristike bilo kojeg testa koji poverava da li je zadati broj prost: o njegovoj vremenskoj složenosti i klasi složenosti kojoj pripada. Te dve karakteristike određuju efikasnost i značaj samog testa.

## **Vremenska složenost i klase složenosti**

U računarstvu, *vremenska složenost algoritma* predstavlja matematičku funkciju koja veličinu unosa pretvara u količinu vremana potrebnu da se algoritam izvrši. Ona se najčešće izražava koristeći se *O* notacijom koja zanemaruje konstantne faktore i sabirke nižeg reda veličine. Kada se koristi na prethodno opisan način, kaže se da se vremenska složenost označava asimptotski kada veličina ulaza teži beskonačnosti. Vremenska složenost se obično procenjuje na taj način što se odredi broj elementarnih operacija koje se izvršavaju u algoritmu, gde se pretopostavlja da je za izvršenje svake od tih operacija potrebna fiksno određena količina vremena. Stoga se količina vremena i broj elementarnih operacija algoritma razlikuju za najviše konstantan faktor.

U računskoj teoriji složenosti, *klasa složenosti* je skup problema povezane složenosti. Tipična klasa složenosti se definiše kao skup problema koji se može rešiti uz pomoć apstraktne mašine *M* koristeći *O*(*f(n))* resursa *R*, gde *n* predstavlja veličinu ulaza. Za algoritam se kaže da je polinomijane vremenske složenosti, ako je njegovo vreme izvršenja ograničeno odozgo sa polinomijalnim izrazom u zavisnosti od veličine ulaza. Na primer, ako je vroj n ulazna veličina nekog algoritma polinomijalne vremenske složenosti, onda se negova složenost može predstaviti sa *O(),* za neki konstantan faktor *k*. Problemi za koje algoritmi sa polinomnom vremenskom složenošću postoje, pripadaju klasi složenosti P. Klasa složenosti P je centralna u računskoj teoriji složenosti, a takođe i u našoj priči o prostim brojevima.

## **Nedeterministički testovi**

Zbog svoje jednostavnosti, najpopularniji testovi za proveru da li je zadati broj prost su upravo nedeterministički testovi. Ovi testovi pored broja *n* čija se složenost testira, koriste i slučajno izabran broj *a* iz određenog skupa mogućih vrednosti. Oni nikada ne proglase prost broj složenim, ali je moguće da složen broj bude označen kao prost (pseudoprosti brojevi). Najpoznatiji nedeterministički testovi su: Fermat, Miller-Rabin, Solovay-Strassen…

Koraci u izvršenju proizvoljnog nedeterminističkog testa su sledeći:

* Proizvoljno se izabere broj *a.*
* Ispita se neka jednakost (koja je specifična za svaki od testova) koja uključuje izabrano *a* i broj *n*  koji se testira. Ako se dobije negativan odgovor na postavljenu jednakost, onda je *n* složen broj, *a* se naziva *svedokom složenosti* broja *n* i test se prekida.
* Vrati se ponovo na korak 1, sve dok se ne postigne željena sigurnost.

Ako je nakon određenog broja iteracija postignuta željena tačnost I za *n* nije ustanovljeno da je složen, onda se on može deklarisati kao *verovatno prost broj.* Termin verovatno prostog broja podrazumeva da će u većini slučajeva zadati broj biti stvarno prost, ali će u nekim slučajevima biti pseudoprost broj.

## **Deterministički testovi**

Za razliku od nedeterminističkih testova gde rezultat tesitranja može biti da je broj složen ili prost, ali sa određenom verovatnoćom, rezultat determinističkih testova sa sigurnošću tvrdi da je broj prost ili složen. Oni nam daju sertifikat,odnosno dokaz da je uneti broj prost. Neki od najpoznatijih testova su: Lucas-Lehmer, Jacobi sums, Elliptic curve primality proving, AKS(Agrawal-Kayal-Saxena)…

Prvi deterministički test koji je bio značajno brži od linearnih metoda (Naive, Eratosten...) je APRCL(Adleman-Pomerance-Rumely) test. Njegovo vreme izvršenja je *O(),* gde je *n* broj koji se testira, a *c* konstanta nezavisna od tog broja. Za test pomoću eliptičnih krivih se može pokazati da je njegovo vreme izvršenja *O()*, ali samo ako se neka nedokazana tvrđenja iz analitičke teorije brojeva(koje se koriste u njegovoj implementaciji) pretpostavi da su tačna. Prvi deterministički test koji se dokazano izvršava u polinomijalnom vremenu je AKS. Otkriven je 2002. Godine od strane trojice autora Agrawal, Kayal, Saxena. Vreme izvršenja prve verzije je iznosilo *O()*, a nove verzije su sve efikasnije.

Pokusaj da nadjese sliku ili napravi sam tabelu podele Det-NeDet da popunis prostor .

1. Broj 1 je poseban slučaj za koji se uzima da nije ni prost, ni složen. Razlog zbog koga se 1 ne uzima kao prost broj je zati što onda ne bi važila jedinstvena faktorizacija broja u *Osnovnoj teoremi aritmetike.* [↑](#footnote-ref-1)