Implementacija Petrogradskog paradoksa u programskom jeziku C

Grabić, Dominik

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU
ELEKTROTEHNIČKI FAKULTET

Sveučilišni studij

IMPLEMENTACIJA PETROGRADSKOG PARADOKSA U PROGRAMSKOM JEZIKU C

Završni rad

Dominik Grabić

Osijek, 2016.

FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

Obrazac Z1P - Obrazac za ocjenu završnog rada na preddiplomskom sveučilišnom studiju

Osijek, 09.09.2016.

Odboru za završne i diplomske ispite

Prijedlog ocjene završnog rada

Ime i prezime studenta:	Dominik Grabić
Studij, smjer:	Preddiplomski sveučilišni studij Računarstvo
Mat. br. studenta, godina upisa:	R3417, 03.09.2012.
OIB studenta:	94582666526
Mentor:	Doc.dr.sc. Tomislav Rudec
Sumentor:	Doc.dr.sc. Alfonzo Baumgartner
Naslov završnog rada:	Implementacija Petrogradskog paradoksa u programskom jeziku C
Znanstvena grana rada:	**Obradba informacija (zn. polje računarstvo)**
Predložena ocjena 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: 3 Jasnoća pismenog izražavanja: 2 Razina samostalnosti: 2
Datum prijedloga ocjene mentora:	09.09.2016.
Datum potvrde ocjene Odbora:	12.09.2016.

Potpis mentora za predaju konačne verzije rada u Studentsku službu pri završetku studija:	Potpis:
	Datum:

FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

IZJAVA O ORIGINALNOSTI RADA

Osijek, 12.09.2016.

Ime i prezime studenta:	Dominik Grabić
Studij:	Preddiplomski sveučilišni studij Računarstvo
Mat. br. studenta, godina upisa:	R3417, 03.09.2012.
Ephorus podudaranje [%]:	0

Ovom izjavom izjavljujem da je rad pod nazivom**: Implementacija Petrogradskog paradoksa u programskom jeziku C**

izrađen pod vodstvom mentora Doc.dr.sc. Tomislav Rudec

i sumentora Doc.dr.sc. Alfonzo Baumgartner

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

Cilj ovog završnog rada je opisati Petrogradski paradoks i implementirati simulaciju Petrogradskog paradoksa u programskom jeziku C. Na početku rada daje se uvod u problematiku, spominje se povijest otkrića i opisuju se neka od dosadašnjih rješenja. Nakon toga bit će govora o važnosti izvedenih zaključaka i uporabi istih u rješavanju problema iz stvarnoga svijeta.

Glavni dio završnog rada podijeljen je u dva poglavlja, odnosno dvije funkcionalne cjeline. Prvo poglavlje ima naziv „Teorijski opis Petrogradskog paradoksa“, a drugo poglavlje se zove „Implementacija Petrogradskog paradoksa u programskom jeziku C“.

Prvo poglavlje, „Teorijski opis Petrogradskog paradoksa“, bavi se objašnjenjem teorijske podloge paradoksa. U tom dijelu najprije će biti opisana povijest otkrića paradoksa. U okviru povijesti bit će prikazani neki od značajnijih ljudi zaslužnih za otkriće i rješavanje problema. Nakon toga spominju se neka od danih rješenja paradoksa, s naglaskom na Bernoullijevo rješenje. Na samom kraju poglavlja daje se pregled uporabe izvedenih zaključaka, odnosno njihov utjecaj na rješavanje problema iz raznih područja ljudskog djelovanja.

Drugo poglavlje naziva „Implementacija Petrogradskog paradoksa u C-u“ bavi se upravo time. U tom poglavlju bit će opisana izrada programa vezanog uz ovu tematiku. Program predstavlja simulaciju izvođenja pokusa s velikim brojem ponavljanja, koji bi bilo nepraktično izvoditi bez uporabe računala. Najprije se daje prikaz dijagrama toka programa, nakon čega slijedi ispis programa u pseudojeziku i u C jeziku. Na kraju poglavlja daje se uvid u rezultat simulacije i njegov komentar.

1.1. Zadatak završnog rada

Zadatak ovog završnog rada je napraviti opis Petrogradskog paradoksa. U teorijskom dijelu zadatka potrebno je dati teorijsku podlogu problema, dok je u praktičnom dijelu cilj implementirati problem u programskom jeziku C i napraviti simulaciju.

2. TEORIJSKI OPIS PETROGRADSKOG PARADOKSA

2.1.Povijest

Problem Petrogradskog paradoksa prvi je uočio švicarski matematičar Nicolaus Bernoulli. [1] U svojem izvornom obliku Petrogradski paradoks opisan je kao igra na sreću namijenjena za dva igrača. Jedan igrač ima ulogu kockara, te želi osvojiti nagradu igrajući igru. Drugi igrač predstavlja banku. Banka kockaru nudi priliku za igranje i isplaćuje nagradu nakon svakog kruga igre. Na samom početku igrači se dogovaraju o iznosu uloga potrebnog za sudjelovanje u igri. Svaki krug igre započinje time da kockar plati banci dogovoreni iznos uloga. Zatim počinje igra. Igra se odvija bacanjem poštenog novčića. Pošteni novčić je novčić koji pri bacanju ima jednaku vjerojatnost pojavljivanja pisma i glave. Takav novčić baca se sve dok ne padne glava, a usput se bilježi koliko puta je bačen novčić. Kada padne glava prestaje bacanje novčića i kockaru banka isplaćuje nagradu. Iznos nagrade ovisi o tome na kojem je bacanju završila igra. Ako je igra završila na prvom bacanju tada nagrada iznosi 2 kune. Ukoliko igra završi na drugom bacanju nagrada iznosi 4 kune. Za završetak igre na trećem bacanju nagrada iznosi 8 kuna itd. Općenito, nagrada za svaki krug igre računa se na način $nagrada = 2^n$ gdje n predstavlja redni broj bacanja na kojem je završila igra, tj. redni broj bacanja na kojem se pojavila glava. Isplatom nagrade završava jedan krug igre Petrogradskog paradoksa. Ako kockar može nastaviti igru tada započinje novi krug. Igra je završena kada kockar više ne može platiti ulog. Pitanje koje se postavlja pri proučavanju ove igre glasi: koliki bi ulog u takvoj igri racionalan igrač bio spreman platiti?

Tim pitanjem pozabavio se spomenuti matematičar Nicolaus Bernoulli i postavlja ga u jednom od pisama iz 1713. godine koje razmjenjuje s Pierreom Raymondom de Montmortom [2]. Nicolaus u pismu daje svoje rješenje paradoksa u kojem se koristi idejom da se svaki ishod s ekstremno malom vjerojatnošću pojavljivanja može zanemariti. Smatrao je da stoga što je frekvencija pojavljivanja takvih događaja jako niska, čovjek se ne bi njima trebao zamarati. Za gornju granicu zanemarivih ishoda postavio je vjerojatnost od 10^{-4}.

Iduća osoba koja se bavila problemom Petrogradskog paradoksa je švicarski matematičar Gabriel Cramer. On je svoje rješenje iznio Nicolausu Bernoulliju u pismu napisanom 1728. godine.

U tom pismu zaključuje da postoji granica preko koje daljnji dobitak ne stvara veće zadovoljstvo niti utječe na odluku za sudjelovanje u igri. Kao gornju granicu navodi vrijednost od 2^{24} što iznosi oko 16 milijuna. Svaka veća svota prema njemu vrijedi isto kao tih 16 milijuna.

Detaljno rješenje problema predstavlja rođak Nicolausa Bernoullija, Daniel Bernoulli. On provodi detaljno istraživanje vezano uz tu problematiku i iz toga proizlazi njegovo djelo o teoriji mjerenja rizika [3]. Djelo je po prvi puta objavljeno na Znanstvenoj akademiji u Petrogradu, dajući tako ime paradoksu.

2.2.Bernoullijevo rješenje paradoksa

Prije opisivanja samog rješenja paradoksa vratimo se na primjer s prošle stranice koji se odnosi na opisanu igru na sreću. U tablici 2.1. prikazane su vrijednosti za: vjerojatnost ishoda (P(n)), pripadajuću isplatu i očekivanu vrijednost za prvih deset slučajeva, te za općeniti n-ti slučaj.

Tablica 2.1Prikaz ishoda igre na sreću

n	$P(n)$	Isplata	Očekivana isplata
1	$\frac{1}{2}$	2	1
2	$\frac{1}{4}$	4	1
3	$\frac{1}{8}$	8	1
4	$\frac{1}{16}$	16	1
5	$\frac{1}{32}$	32	1
6	$\frac{1}{64}$	64	1
7	$\frac{1}{128}$	128	1
8	$\frac{1}{256}$	256	1
9	$\frac{1}{512}$	512	1
10	$\frac{1}{1024}$	1024	1
n	$\frac{1}{2^n}$	2^n	1

Veličina n u tablici odnosi se na redni broj bacanja na kojem se pojavila glava, a očekivana isplata računa se prema relaciji (2-1).

$$O(n) = \frac{1}{2^n} \cdot 2^n \tag{2-1}$$

gdje je:

$\boldsymbol{n}$ – redni broj bacanja na kojem se pojavila glava

$\boldsymbol{O(n)}$ – očekivana vrijednost isplate za n-ti slučaj

Idući korak nakon izrade tablice je izračunavanje očekivane vrijednosti igre. Pošto se ovdje radi o diskretnoj slučajnoj varijabli tada očekivana vrijednost u ovom slučaju glasi (2-2):

$$E[X] = \sum_{i=1}^{\infty} p_i x_i = \sum_{i=1}^{\infty} \frac{1}{2^i} \cdot 2^i = 1 + 1 + \cdots + 1 = \infty \tag{2-2}$$

gdje je:

$\boldsymbol{E[X]}$ – matematičko očekivanje diskretne slučajne varijable X

$\boldsymbol{p_i}$ – vjerojatnost i-tog ishoda

$\boldsymbol{x_i}$ – isplata pri i-tom ishodu

Uzimajući u obzir tablicu i izračun očekivane vrijednosti igre, ponovno se postavlja pitanje s početka poglavlja. Koliko novaca bi racionalan igrač bio spreman platiti za sudjelovanje u takvoj igri? Razumno bi bilo, iz izvedene analize, reći da bi igrač trebao platiti bilo koji iznos manji od očekivane vrijednosti igre za sudjelovanje. Budući da je ranije izračunato da očekivana vrijednost ide u beskonačnost, logično je zaključiti da bi igrač trebao platiti bilo koju konačnu vrijednost za sudjelovanje u igri. Međutim, takav zaključak nam se ne čini intuitivan. [1] Mnogi ljudi uključeni u rješavanje paradoksa slažu se s izjavom Iana Hackinga iz 1980. godine „Mali broj nas bi pristao platiti i 25 dolara za sudjelovanje u takvoj igri.“

Daniel Bernoulli za objašnjenje ovog paradoksa uvodi pojam korisnosti. [4] Prema njegovoj teoriji pojedinac pridružuje subjektivnu vrijednost svakoj dodatnoj jedinici novca ovisno o tome koliko novaca već posjeduje. Prema njegovim riječima „... svako povećanje bogatstva, bez obzira koliko beznačajno, za rezultat ima povećanje korisnosti, koja je obrnuto proporcionalna količini dobara koju [pojedinac] posjeduje.“ Matematički to se može prikazati sljedećom relacijom (2-3):

$$\Delta U \propto \frac{\Delta w}{w} \tag{2-3}$$

gdje je:

$\boldsymbol{\Delta U}$ – promjena korisnosti dobara

$\boldsymbol{\Delta w}$ – promjena ukupno posjedovanih dobara

$\boldsymbol{w}$ – ukupna količina posjedovanih dobara

Nadalje, Bernoulli predlaže da se za opisivanje korisnosti dobara koristi logaritamska funkcija u obliku (2-4):

$$U(w) = k \cdot log(w) + konst. \tag{2-4}$$

gdje je:

$\boldsymbol{U(w)}$ – korisnost u ovisnosti o količini posjedovanih dobara

$\boldsymbol{k}$ – parametar

$\boldsymbol{w}$ – ukupna količina posjedovanih dobara

Uzevši u obzir Bernoullijevu hipotezu, tablicu 2.1 s početka potpoglavlja moguće je nadopuniti s dva stupca. Prvi od njih označava korisnost isplate pozivajući se na relaciju (2-4), dok drugi stupac prikazuje očekivanu korisnost, koja se slično kao i kod očekivane vrijednosti dobiva kao umnožak korisnosti isplate i vjerojatnosti pripadajućeg ishoda. Ako se u relaciji (2-4) postavi k=1, te $konst.$=0, te ako se postavi $P(n)$ za pojavljivanje glave na n-tom bacanju tada se dobiva tablica (2.2):

Tablica 2.2Prikaz korisnosti isplate i marginalne korisnosti

n	$P(n)$	Isplata	Korisnost isplate	Očekivana korisnost
1	$\frac{1}{2}$	2	0,301	0,1505
2	$\frac{1}{4}$	4	0,602	0,1505
3	$\frac{1}{8}$	8	0,903	0,1129
4	$\frac{1}{16}$	16	1,204	0,0753
5	$\frac{1}{32}$	32	1,505	0,0470
6	$\frac{1}{64}$	64	1,806	0,0282
7	$\frac{1}{128}$	128	2,107	0,0165
8	$\frac{1}{256}$	256	2,408	0,0094
9	$\frac{1}{512}$	512	2,709	0,0053
10	$\frac{1}{1024}$	1024	3,010	0,0029

U slučaju iz tablice (2.2) očekivana korisnost iznosi:

$$\frac{1}{2}\log 2 + \frac{1}{4}log4 + \cdots + \frac{1}{n}logn = \sum_{i=1}^{\infty} \frac{log2^i}{2^i} = log4$$

Očekivana korisnost odgovara ulogu od 4 novčane jedinice, pa se prema Bernoullijevoj pretpostavci može reći da će racionalan igrač biti spreman platiti najviše 4 novčane jedinice za sudjelovanje u toj igri.

2.3.Kritike Bernoullijevog rješenja i novija rješenja paradoksa

Kritičari Bernoullijevog istraživanja smatraju da predstavljanje korisnosti logaritamskom funkcijom nije dovoljno precizno. Prema njegovom razmatranju, razlika između nagrade od 4 i 8 novčane jedinice pruža jednaku razliku u korisnosti kao skok s iznosa nagrade od 512 na 1024 novčane jedinice. Stoga su razvijene i druge metode smanjenja korisnosti u ovisnosti o povećanju nagrade, a sve vode do istog zaključka; ako suma očekivane korisnosti teži ka konačnoj vrijednosti, tada je paradoks riješen. U tom slučaju racionalno je platit iznos manji od te sume.

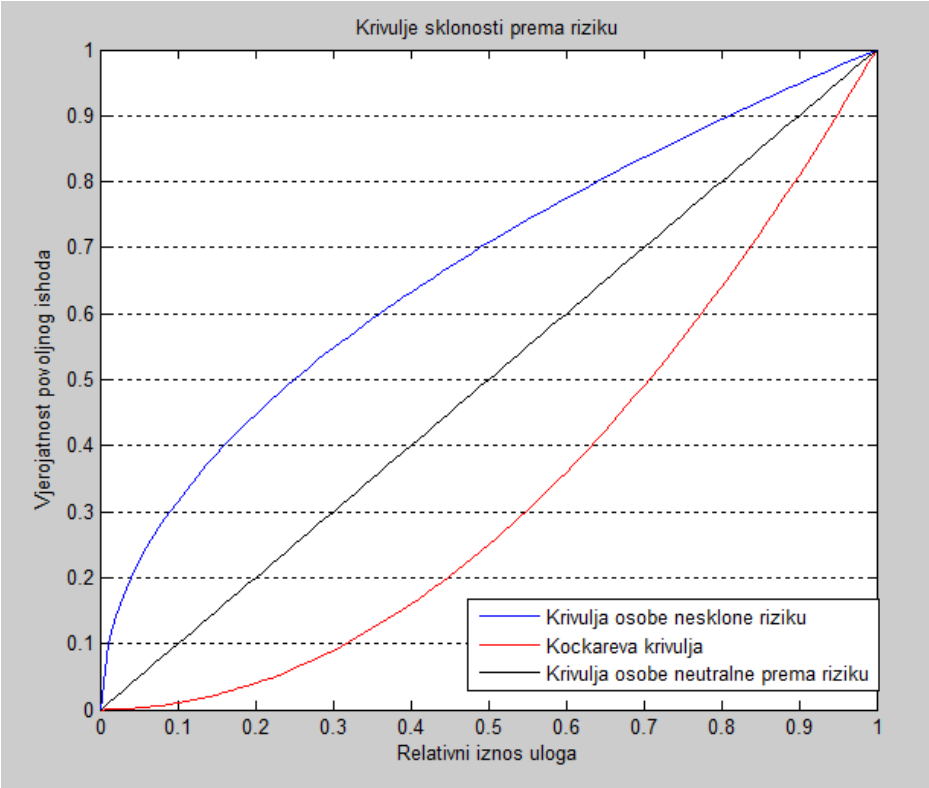
[1] Modificirana verzija igre Petrogradskog paradoksa pokazuje da problem nije riješen ako suma očekivane korisnosti teži ka konačnoj vrijednosti. Pod pretpostavkom da se korisnost može opisati logaritamskom ovisnošću, kako je to Bernoulli postavio, a povećanjem isplate na iznos oblika 10^{2^n} dobiva se sljedeća tablica (2.3):

Tablica 2.3Modificirana verzija Petrogradskog paradoksa

n	$P(n)$	Isplata	Korisnost isplate	Očekivana korisnost
1	$\frac{1}{2}$	10^2	2	1
2	$\frac{1}{4}$	10^4	4	1
3	$\frac{1}{8}$	10^8	8	1
4	$\frac{1}{16}$	10^{16}	16	1
5	$\frac{1}{32}$	10^{32}	32	1
6	$\frac{1}{64}$	10^{64}	64	1
7	$\frac{1}{128}$	10^{128}	128	1
8	$\frac{1}{256}$	10^{256}	256	1
9	$\frac{1}{512}$	10^{512}	512	1
10	$\frac{1}{1024}$	10^{1024}	1024	1

Suma očekivane korisnosti u ovom slučaju ide u beskonačnost. Ponovno se može izvući zaključak da treba platiti bilo koju konačnu svotu novca za pristup igri, te se pojavljuje isti problem.

[5] Pri razmatranju rješenja paradoksa treba uzeti u obzir sklonost igrača prema riziku. To se može učiniti na više načina. Jedan od načina je modificiranje funkcije korisnosti na način da je primjerice, korist dobitka od 10 kuna s vjerojatnošću od 100% veća nego korist pri dobitku od 20 kuna s vjerojatnošću od 50%. Drugi način bi bio uvođenje negativne vrijednosti za korisnost, uzrokovane nelagodom zbog neizvjesnosti događaja. Weirich u svom rješenju iznosi zaključak da sklonost riziku treba uzeti u obzir, te daje način na koji se sklonost riziku može uklopiti u proračunima očekivane korisnosti. Rezultat njegovog razmatranja je taj da postoji konačna gornja granica za cijenu sudjelovanja u igri. Međutim, sklonost prema riziku se ne može u općenitom slučaju uzeti u obzir jer postoje ljudi koji uživaju u riskiranju, te su skloni precijeniti vjerojatnost za dobitak. Velik broj ljudi uplaćuje lutriju iako je šansa za osvajanje glavne nagrade praktički nepostojeća. Osobe je stoga prema njihovoj sklonosti riziku moguće podijeliti u tri grupe; u prvu grupu spadaju ljudi neskloni riskiranju, u drugu grupu osobe neutralnog stava prema riziku, dok se u treću grupu mogu smjestiti osobe sklone riskiranju, odnosno kockare. Krivulje sklonosti pojedinih grupa prema riziku skicirane su na slici 2.1.



Slika 2.1Skica krivulja sklonosti riziku

[6] Sredinom 20. stoljeća dolazi se do značajnog napretka u raspravi o paradoksu. Tijekom prevođenja djela Daniela Bernoullija o paradoksu s latinskog na engleski pronađena je greška u njegovom rješenju. Bernoulli u svojem radu piše: „Broj slučajeva koji se ovdje treba uzeti u obzir je beskonačan: u polovici tih slučajeva igra završava na prvom bacanju, u četvrtini slučajeva završava na drugom, u osmini slučajeva na trećem, u šesnaestini slučajeva na četvrtom, i tako dalje... do beskonačnosti...“. Karl Menger, koji je tada bio tehnički konzultant tijekom prijevoda Bernoullijeva rada primjećuje grešku: „Budući da je broj slučajeva beskonačan, nemoguće je govoriti o polovini slučajeva, četvrtini slučajeva itd...“ Menger nije shvaćao značaj greške i stoga ju nije u potpunosti ispravio. Ako se uzme u obzir da igra mora završiti nakon konačnog broja bacanja, te ako se sukladno tome korigira očekivana vrijednost igre, tada ta vrijednost postaje konačna, i poklapa se s iznosom koji bi ljudi bili spremni platiti za sudjelovanje u igri. S pravilnim izvodom očekivane vrijednosti, dakle, nema paradoksa.

2.4.Značaj Petrogradskog paradoksa

Otkriće i opis Petrogradskog paradoksa imalo je značajan utjecaj na daljnja istraživanja u raznim granama ekonomije i u statistici. U ekonomiji istraživači su se počeli baviti razvijanjem teorije o riziku i procjeni rizika, što je u današnjem poslovnom svijetu tema od velike važnosti. Osim toga, doalazi i do proširivanja Bernoullijeve ideje o koristi dobara. Potaknuti Bernoullijevim razmatranjem, Von Neumann i Morgenstern razvijaju teoriju očekivane korisnosti. [7] U svojem djelu po uzoru na Bernoullijevu funkciju korisnosti u ovisnosti o bogatstvu, deifiniraju funkciju očekivane korisnosti igara na sreću.

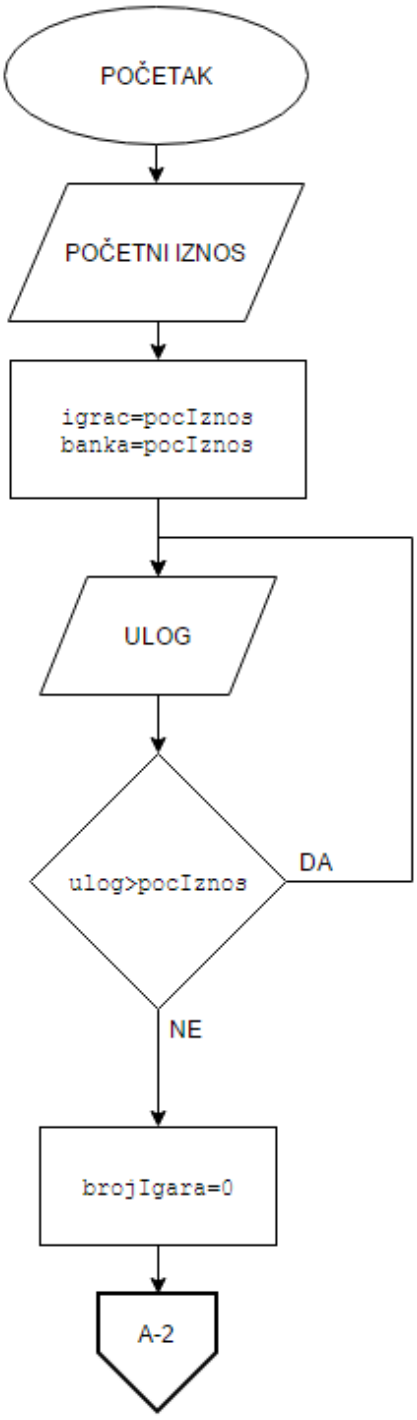
3. IMPLEMENTACIJA PETROGRADSKOG PARADOKSA U C-u

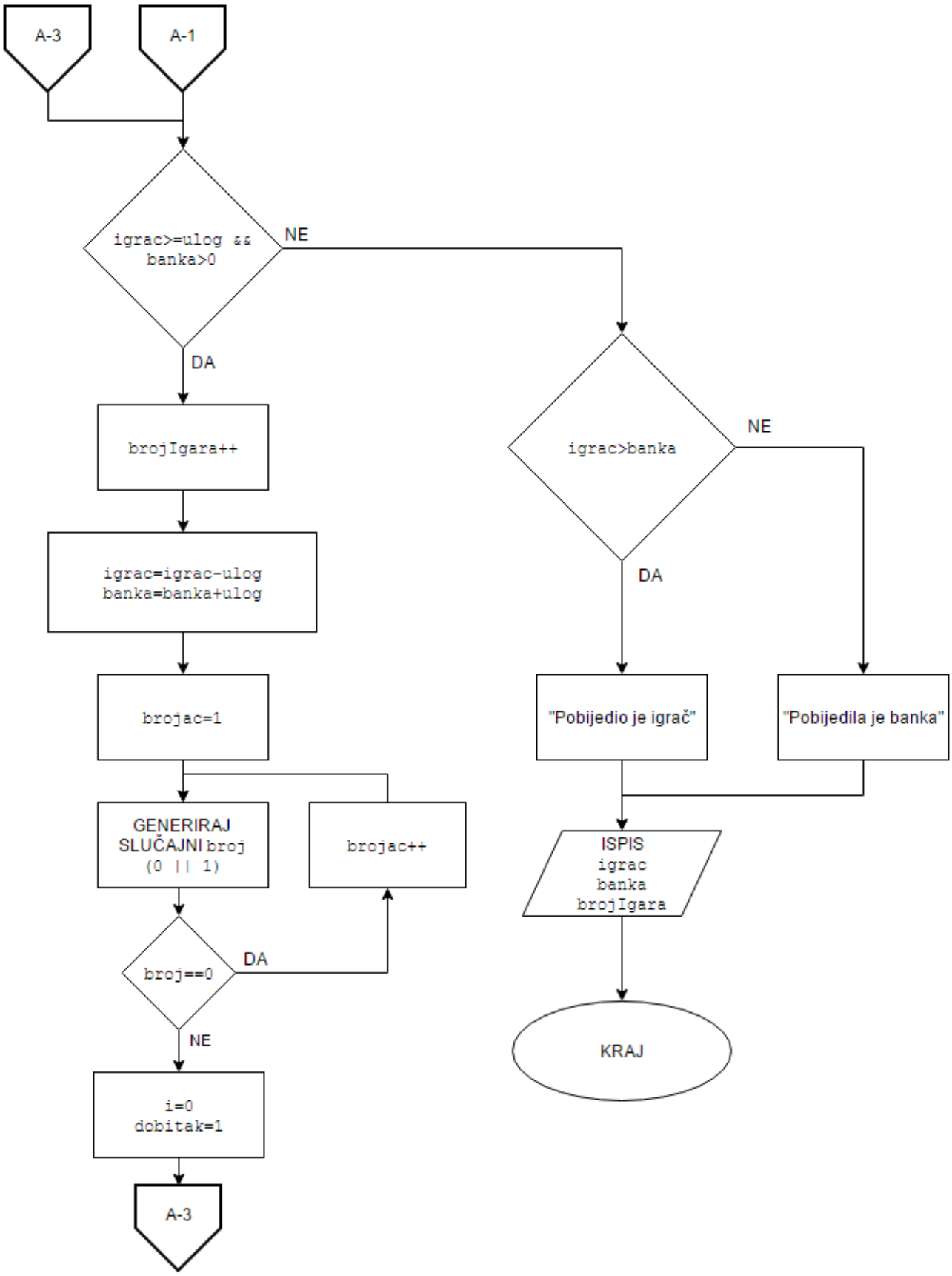
3.1. Opis zadatka

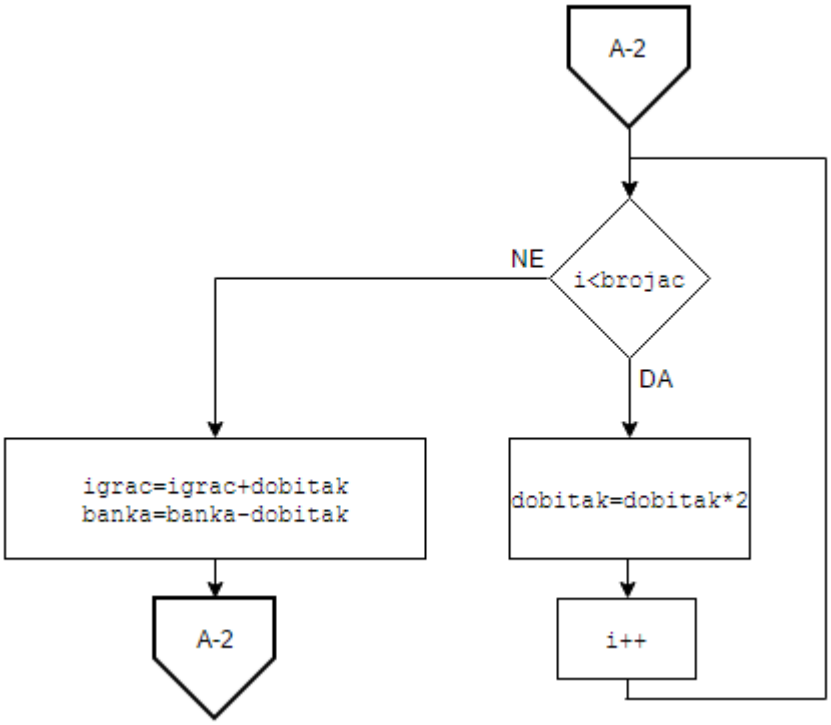
U praktičnom dijelu ovog završnog rada treba napisati C program za simulaciju igre Petrogradskog paradoksa. Igra se odvija prema sljedećim koracima. Igra je zamišljena za dva igrača. prvi igrač predstavlja kockara, odnosno *igrača* kojemu se nudi sudjelovanje u igri Petrogradskog paradoksa. Drugi igrač predstavlja *banku*, odnosno igrača koji nudi igru Petrogradskog paradoksa. Na početku programa od korisnika se traži unos početnog stanja. Početno stanje predstavlja iznos novaca kojima raspolažu igrači. Pretpostavka je da i igrač i banka započinju igru s istom svotom novaca. Nakon toga korisnik treba unijeti iznos uloga potrebnog za sudjelovanje u igri. Pri tome se provjerava je li iznos uloga veći od iznosa kojim raspolažu igrači. Ako je ulog veći od iznosa kojim raspolažu igrači, tada program traži ponovni unos uloga, jer se u protivnom igra ne može održati. Nakon što su vrijednosti za početni iznos i ulog pravilno postavljene, započinje se s igrom. Od iznosa igrača najprije se oduzima ulog, koji se zatim dodaje banci. Bacanje novčića simulira se odabiranjem slučajnog broja (nula ili jedinica). Nula u programu označava ishod u kojemu je palo pismo, a jedinica označava ishod u kojemu je pala glava. Program pseudo slučajnim odabirom odabire brojeve sve dok se ne pojavi prva jedinica. Pri tome brojač bilježi broj bacanja koji se odigrao do ishoda prve jedinice. Nakon toga program računa nagradu za igrača. Nagrada za igru računa se prema sljedećoj formuli: $nagrada = 2^n$ gdje n predstavlja broj bacanja na kojemu se prvi put pojavljuje glava. Vrijednost za n određuje se pomoću brojača. Zatim se izračunata vrijednost nagrade pridodaje iznosu igrača, dok se taj isti iznos oduzima od iznosa banke. Time se zaključuje jedan ciklus igre. Igra Petrogradskog paradoksa se ponavlja dokle god su ispunjeni navedeni uvjeti; igrač raspolaže iznosom koji je veći ili jednak ulogu i dok banka raspolaže iznosom koji je veći od nule, tj. pozitivan je. Kada nije zadovoljen barem jedan od ta dva slučaja igra se prekida. Tada postoje dva ishoda. U prvom od tih ishoda igrač nema dovoljno novaca za daljnje sudjelovanje u igri. Tada pobjeđuje banka. Prilikom drugog ishoda svota kojom raspolaže banka iznosi ili nula ili manje od nule. Ako je iznos manji od nule to znači da banka ostaje dužna igraču navedenu svotu. Na samom kraju programa ispisuju se iznosi igrača i banke te broj igara potrebnih da se dođe do kraja.

3.2.Dijagram toka programa

U ovom potpoglavlju bit će prikazan dijagram toka programa simulacije igre Petrogradskog paradoksa.




A-3
A-1
igrac>=ulog &&
banka>0
NE
DA
brojIgara++
igrac=igrac-ulog
banka=banka+ulog
brojac=1
GENERIRAJ
SLUČAJNI broj
(0 || 1)
brojac++
broj==0
DA
NE
i=0
dobitak=1
A-3
igrac>banka
NE
DA
"Pobijedio je igrač"
"Pobijedila je banka"
ISPIS
igrac
banka
brojIgara
KRAJ


A-2
NE
i<brojac
DA
igrac=igrac+dobitak
banka=banka-dobitak
dobitak=dobitak*2
A-2
i++

3.3.Pseudokod programa

```
početak
činiti
	ulaz pocIznos;
dok je pocIznos < 1
igrac := pocIznos;
banka:= pocIznos;
činiti
	ulaz ulog;
dok je ulog > pocIznos ILI ulog <= 1
brojIgara := 0;
dok je igrac >= ulog I banka > 0 činiti
	brojIgara := brojIgara + 1;
	igrac:= igrac – ulog;
	banka:= banka + ulog;
	brojac := 1;
	činiti
		broj := slučajni broj;
		ako broj = 0 onda brojac = brojac + 1;
	dok je broj = 0;
	i := 0;
	dobitak := 1;
	dok je i < brojac činiti
		dobitak := dobitak * 2;
		i := i + 1;
	igrac:= igrac + dobitak;
	banka:= banka – dobitak;
ako igrac > banka onda
	izlaz “Pobijedio je igrač”;
inače
	izlaz “Pobijedila je banka”;
izlaz igrac;
izlaz banka;
izlaz brojIgara;
kraj
```

3.4.Rješenje zadatka u programskom jeziku C

Ovo potpogavlje rada bavi se rješenjem zadanog problema u programskom jeziku C. U tu svrhu će u nastavku biti prikazani neki isječci programa uz kratki komentar za svaki od njih. Cjelokupni ispis programa nalazi se u prilogu (8.1).

```
do
{
    printf("# Unesite pocetni iznos za igrace: ");
    scanf_s("%d", &pocIznos);
    if (pocIznos <= 0)
        printf("# Vrijednost pocetnog iznosa mora biti pozitivan cijeli broj!\n");
    pombr++;
    if (pombr >= 10)return -1;
} while (pocIznos <= 0);
```

Sukladno opisu s početka poglavlja u prvom dijelu programa od korisnika se traži unos početnog iznosa kojim raspolažu igrač i banka. U ovom slučaju se pretpostavlja da oba igrača započinju igru s jednakim iznosom. Početni iznos mora imati pozitivnu cjelobrojnu vrijednost. Da bi se prilikom izvođenja programa spriječila mogućnost beskonačne petlje postavlja se varijabla `pombr` koja bilježi broj neuspješnih postavljanja početne vrijednosti. Nakon desetog neuspješnog pokušaja postavljanja izvođenje programa se prekida.

```
do
{
    printf("# Unesite iznos uloga: ");
    scanf_s("%d", &ulog);
    if (ulog <= 0)
        printf("# Vrijednost uloga ne smije biti manja od 1!\n");
    if (ulog > pocIznos)
        printf("# Vrijednost uloga ne smije prelaziti vrijednost pocetnog iznosa!\n");
    pombr++;
    if (pombr >= 10)return -1;
} while (ulog <= 0 || ulog > pocIznos);
```

Nakon što se ispravno postavi vrijednost početnog iznosa za igrače slijedi postavljanje vrijednosti uloga. Petlja za postavljanje uloga provjerava je li vrijednost uloga ispravno postavljenja, tj. ulog ne smije biti manji od jedan niti ne smije prelaziti vrijednost početnog iznosa.

```
while (igrac >= ulog && banka > 0){
    brojIgara++;
    igrac -= ulog;
    banka += ulog;
    brojac = 1;
    do{
        broj = rand() & 1;
        if (broj == 0)
            brojac++;
    }while (broj == 0);
    dobitak = izracunajDobitak(brojac);
    if (dobitak == 2) dobitak2++;
    if (dobitak > maxDobitak) maxDobitak = dobitak;
    if (igrac > maxIgrac) maxIgrac = igrac;
    if (banka > maxBanka) maxBanka = banka;
    igrac += dobitak;
    banka -= dobitak;
}
```

Glavni dio programa predstavlja petlja koja simulira igru Petrogradskog paradoksa. Osnovni uvjet da bi se mogla provesti igra je taj da igrač ima dovoljno novaca za plaćanje uloga i da u isto vrijeme banka nije na nuli ili u minusu. Ako je taj uvjet zadovoljen tada se započinje s igrom. Igra se odvija na način da se u varijablu `broj` spremi nasumično odabrana vrijednost nula ili jedan. Sve dok ta varijabla ne poprimi vrijednost nula igra se ponavlja, a u svakoj iteraciji vrijednost brojača se povećava za jedan. Kada se pojavi nula, igra se završava i pozivanjem funkcije računa se vrijednost dobitka te se ažuriraju vrijednosti za igrača i banku te se program vraća na početni uvjet.

```
int izracunajDobitak(int brojac){
    int i, dobitak = 1;
    for (i = 0; i < brojac; i++)
        dobitak *= 2;
    return dobitak;
}
```

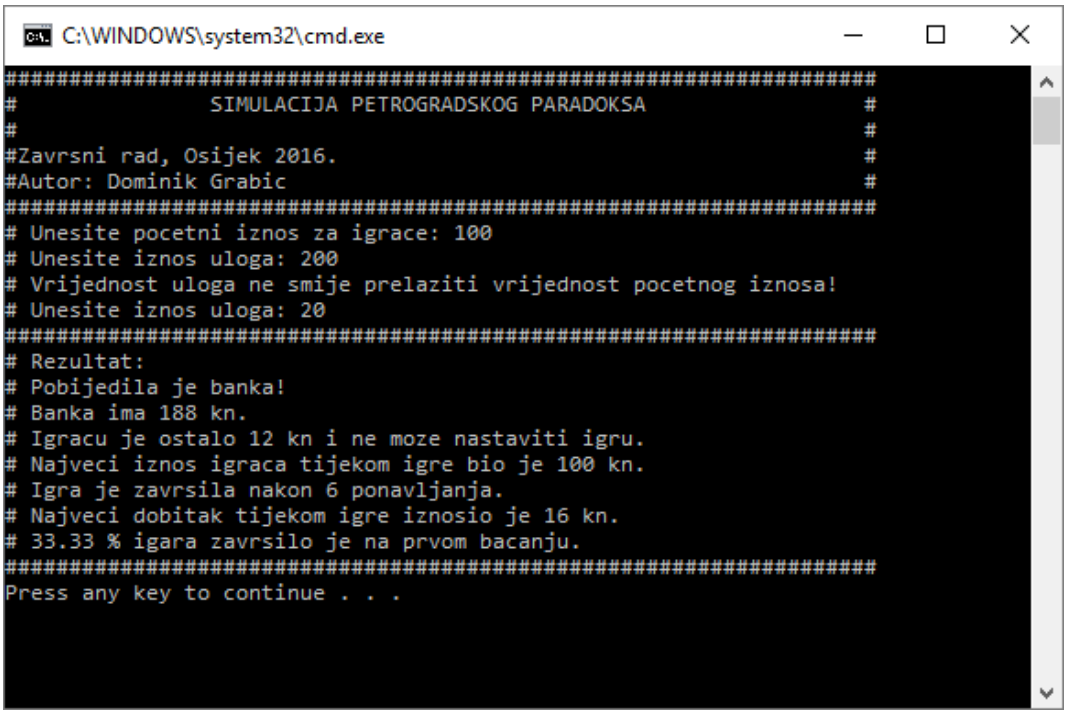
Funkcija izračunavanja dobitka prilikom igre zove se `izracunajDobitak`. Ta funkcija kao rezultat vraća cjelobrojnu vrijednost koja predstavlja dobitak igre. Dobitak ovisi o vrijednosti brojača koji pak predstavlja redni broj iteracije na kojoj se po prvi puta pojavljuje nula. Dobitak se u funkciji računa prema sljedećoj formuli $dobitak = 2^{brojac}$.

```
if (igrac > banka){
    printf("\n# Pobijedio je igrac!\n");
    if (banka < 0)
        printf("# Banka je ostala igracu duzna %d kn.\n", banka*-1);
    if (banka == 0)
        printf("# Banci je ostalo %d kn.\n", banka);
    printf("# Igrac ima %d kn.\n", igrac);
}
else{
    printf("\n# Pobijedila je banka!\n");
    printf("# Banka ima %d kn.\n", banka);
    printf("# Igracu je ostalo %d kn i ne moze nastaviti igru.\n", igrac);
}
```

Kada jedan od igrača ne može nastaviti igru program izlazi iz glavne petlje i ispisuje se informacija o pobjedniku i konačno stanje oba igrača.

```
printf("# Igra je zavrsila nakon %d ponavljanja.\n", brojIgara);
printf("# Najveci dobitak tijekom igre iznosio je %d kn.\n", maxDobitak);
printf("# %.2f %% igara zavrsilo je na prvom bacanju.\n", (float)dobitak2 / brojIgara * 100);
printf("#########################################################################");
getchar();
return 0;
```

Na samom kraju programa ispisuju se informacije vezane za cjelokupnu igru i program završava s radom.



Slika 3.1 Ispis programa

3.5.Izračunavanje vrijednosti igre uz zadane uvjete

Budući da u realnom okruženju nije moguće beskonačno dugo igrati igru Petrogradskog paradoksa nego je vrijeme ograničeno razmatra se sljedeći slučaj; neka dva igrača igraju igru tijekom večeri, maksimalnog trajanja od četiri sata, između 20 i 24 sata. Prvi igrač predstavlja igrača, a drugi igrač predstavlja banku. Nadalje, neka se pretpostavi da trajanje jednog bacanja i bilježenje rezultata traje u prosjeku 6 sekundi. Tada se za to vrijeme novčić stigne baciti ukupno $10 * 60 * 4 = 2400$ puta. Početni iznos fiksiran je na vrijednost 1000, a vrijednost nagrade se izračunava prema formuli $nagrada = 2^n$ gdje n predstavlja broj bacanja na kojemu se prvi put pojavljuje glava. Igra se prekida kada igrač više nema dovoljno novaca za nastavak ili kada se novčić baci 2400 puta. Banka u ovoj simulaciji raspolaže neograničenom svotom novaca, što znači da za razliku od prošlog slučaja nastavak igre ne ovisi o iznosu kojim raspolaže banka. Zadatak je pronaći vrijednost uloga za koju oba igrača uz navedene uvjete imaju približno jednaku šansu za pobjedu. U tu svrhu koristi se modificirana verzija programa iz prethodnog potpoglavlja. Cjelokupni ispis ovog programa nalazi se u prilogu (8.2.).

```
FILE *f;//pokazivac na file
errno_t err;
err = fopen_s(&f, "test.txt", "a+");
if (err == 0)printf("The file was opened\n");
else printf("The file was not opened\n");
```

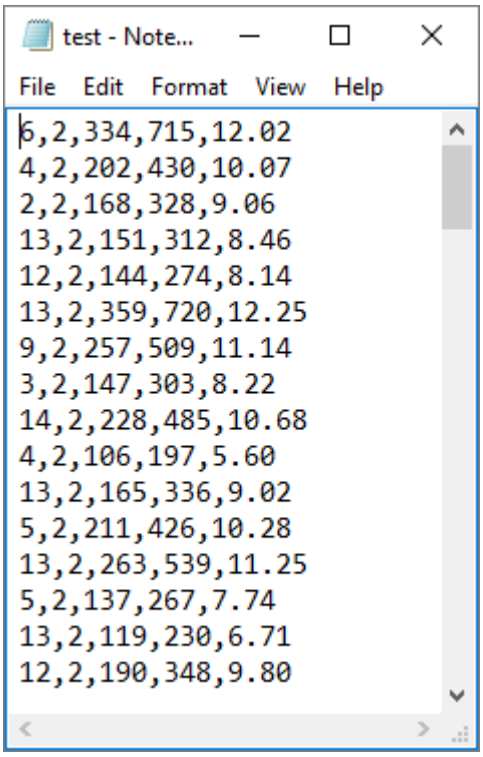
U modificiranoj verziji programa, umjesto u konzolu rezultati simulacije se upisuju u datoteku `test.txt`.

```
#define MAX_TRAJANJE_IGRE 2400
#define BROJ_IGARA 100
#define POCETNO_STANJE 1000
#define ULOG 11
```

Budući da su vrijednosti unaprijed definirane više nije potrebno tražiti korisnički unos. Zbog toga se dio za unos početnih vrijednosti radi jednostavnosti zamjenjuje definicijama na samom početku programa.

```
while (flag == 0 && igrac >= ULOG){
```

Još jedna promjena je u glavnoj funkciji je uvođenje varijable `flag` u uvjetu koja služi za ograničavanje igre na maksimalno 2400 ponavljanja, kako je prethodno izračunato.



Slika 3.2Izvoz podataka u datoteku

Na slici 3.2. prikazan je dio podataka izvezenih u datoteku nakon uspješnog izvršavanja programa. Parametri u ovom slučaju su bili sljedeći:

```
POCETNA_VRIJEDNOST = 1000
ULOG = 15
MAX_TRAJANJE_IGRE = 2400
BROJ_IGARA = 100
```

Svaki red ispisanih podataka predstavlja jednu igru u trajanju od najviše 2400 bacanja. Prvi broj prikazuje stanje igrača na kraju igre, drugi broj označava pobjednika igre (0-neriješeno, 1-pobjednik je igrač, 2-pobjednik je banka), treći broj predstavlja broj odigranih igara, četvrti broj u retku predstavlja ukupan broj bacanja, a posljednji broj je aritmetička sredina isplate tijekom igre.

ULOG=10				
Igrač	Pobjednik	Broj igara	Broj bacanja	Prosječni dobitak
680	2	1183	2400	9,73
8320	1	1224	2400	15,98
1558	1	1172	2400	10,48
1786	1	1199	2400	10,66
1582	1	1220	2400	10,48
10750	1	1160	2400	18,41
1658	1	1199	2400	10,55
6	2	244	460	5,93

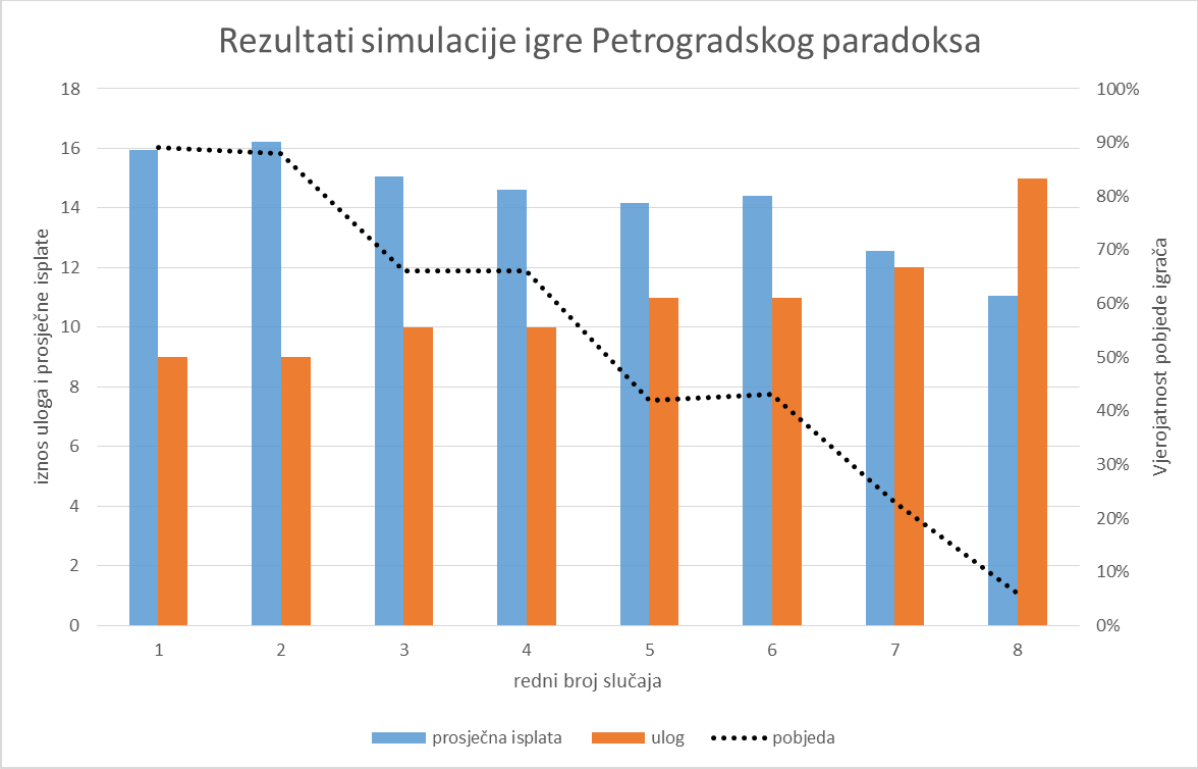
Slika 3.3Dio tablice dobivene iz datoteke

Slika 3.3. prikazuje izvadak iz tablice dobiven iz datoteke `test.csv` koja služi za izvoz podataka iz programa.

Tablica 3.1Statistika pojedinih slučajeva

slučaj	pobjeda	prosječna isplata	ulog
1	89%	15,96	9
2	88%	16,23	9
3	66%	15,07	10
4	66%	14,60	10
5	42%	14,17	11
6	43%	14,41	11
7	23%	12,57	12
8	6%	11,05	15

Tablica 3.1. predstavlja statistiku za 8 ispitanih slučajeva simulacije paradoksa, a rezultati su grafički prikazani na sljedećoj stranici.



Slika 3.4Rezultat simulacije paradoksa

Na slici 3.4. prikazan je graf dobiven iz 8 različitih slučajeva simulacije Petrogradskog paradoksa uz postavljene uvjete. Iz grafa je vidljivo da povećanjem uloga vjerojatnost pobjede igrača opada. Prosječna isplata se također smanjuje povećanjem uloga zbog smanjenja ukupno odigranih igara. Uz postavljene uvjete približno jednaku šansu za pobjedu imaju igrač i banka kada ulog ima vrijednost 10 ili 11, te se stoga može zaključiti da te vrijednosti predstavljaju vrijednost igre pri postavljenim uvjetima.

4. ZAKLJUČAK

Cilj ovog završnog rada bio je obraditi problematiku Petrogradskog paradoksa. Glavni dio rada podijeljen je u dvije cjeline.

Prva od njih bavi se teorijskom pozadinom Petrogradskog paradoksa. Tako se na samom početku tog dijela opisuje povijest otkrića Petrogradskog paradoksa, te se postavlja konkretna situacija kako bi se čitatelja uputilo u problematiku. Nakon toga slijedi opis Bernoullijevog rješenja paradoksa uz pripadajuće jednadžbe i izvode. Potom slijede neke od kritika Bernoullijevog rješenja paradoksa kao i kraći pregled nekih novijih rješenja paradoksa. Na kraju poglavlja daje se uvid u značaj Petrogradskog paradoksa na nastanak novih ideja i razvoj pojedinih grana ekonomije i statistike.

U drugom dijelu rada opisan je postupak izrade simulacije igre Petrogradskog paradoksa u programskom jeziku C. Na početku poglavlja daje se opis tijeka igre i pravila prema kojima se igra odvija. Nakon toga se ispisuje dijagram toka programa te program u pseudojeziku. U nastavku slijedi ispis dijelova gotovog programa u C jeziku uz kratka obrazloženja. Zatim se koristi modificirana verzija programa u svrhu automatiziranja izvođenja pokusa i ispis rezultata simulacije uz postavljene parametre. Na kraju poglavlja korišteni su rezultati simulacije za izračunavanje vrijednosti igre, odnosno one vrijednosti uloga za koju je igra Petrogradskog paradoksa uz postavljene uvjete poštena.

5. LITERATURA

[1] http://plato.stanford.edu/entries/paradox-stpetersburg/ pristup ostvaren: 3.3.2016.

[2] http://cerebro.xu.edu/math/Sources/NBernoulli/correspondence_petersburg_game.pdf pristup ostvaren: 4.3.2016.

[3] http://www.econ.ucsb.edu/~tedb/Courses/GraduateTheoryUCSB/Bernoulli.pdf pristup ostvaren: 9.3.2016.

[4] http://www.econport.org/econport/request?page=man_ru_basics2 pristup ostvaren:14.3.2016.

[5] http://www.econport.org/econport/request?page=man_ru_basics4 pristup ostvaren: 14.3.2016.

[6] https://mpra.ub.uni-muenchen.de/50515/1/MPRA_paper_50515.pdf pristup ostvaren: 14.3.2016

[7] http://www.econport.org/econport/request?page=man_ru_basics3 pristup ostvaren: 15.3.2016.

6. SAŽETAK

Ključni pojmovi: paradoks, igra na sreću, matematičko očekivanje, korisnost, sklonost riziku, ograničenje, simulacija, programski jezik C

Petrogradski paradoks je matematički problem koji je prvi opisao švicarski matematičar Nicolaus Bernoulli. Radi se o igri na sreću za dva igrača koja se odvija bacanjem poštenog novčića. Paradoks ove igre leži u tome da se matematičko očekivanje igre ne poklapa s iznosom koji bi racionalan pojedinac pristao uložiti za sudjelovanje u igri. Mnogi poznati matematičari poput Daniela Bernoullija i Gabriela Cramera pokušali su dati rješenje paradoksa. Njihovi napori rezultirali su otkrićima u području ekonomije i statistike te su doveli do pojavljivanja novih ideja. U ovom završnom radu najprije je opisana teorijska podloga Petrogradskog paradoksa, dok je u praktičnom dijelu rada izrađen program koji simulira odigravanje velikog broja slučajeva igre Petrogradskog paradoksa.

6.1. Abstract

Implementation of St. Petersburg Paradox in C Programming Language

Keywords: paradox, game of chance, expected value, utility, risk aversion, limit, simulation, C programming language

St. Petersburg paradox was first described by Swiss mathematician Nicolaus Bernoulli. It is a game of chance for two players which is played by tossing a fair coin. Paradox here lies within the fact that the expected value of the game does not coincide with the amount of money which a rational decision maker would be willing to pay in order to enter the game. Many famous mathematicians such as Daniel Bernoulli and Gabriel Cramer attempted to solve this paradox. Their efforts led to many breakthroughs in economy and statistics and inspired birth of new ideas. First part of this final paper deals with theoretical background behind the paradox, while a C program for simulation of large quantities of st. petersburg paradox game was written in second, practical, part of this paper.

7. ŽIVOTOPIS

Dominik Grabić

Vinogradska 30, 31 000 Osijek (Hrvatska)

Tel: +385 97 730 58 99

dgrabic@outlook.com

Spol: muško Datum rođenja: 7.3.1994. Državljanstvo: Hrvatsko

Obrazovanje i osposobljavanje:

rujan 2012 Sveučilišni preddiplomski studij računarstva, Elektrotehnički fakultet Osijek,

Kneza Trpimira 2B 31000 Osijek

rujan 2008– lipanj 2012 III. gimnazija Osijek, Ul. Kamila Firingera 14, 31000 osijek

Stečena srednja stručna sprema(SSS).

Osobne vještine:

Materinski jezik: Hrvatski

Ostali jezici: Engleski

Razumjevanje: C1 razina

Govor: B2 razina

Pisanje:B2 razina

Računalne vještine: Osnove rada na računalu, savladano korištenje MS office–a, poznavanje programskih jezika c, c++, upoznat s radom u Linux terminalu, poznavanje osnova SQL-a

Ostale vještine: Razvijene komunikacijske vještine stečene tijekom rada na raznim grupnim projektima, predanost poslu, strpljivost i odgovornost

Dominik Grabić

8. PRILOG

8.1. Ispis programa

```
#include<stdio.h>
#include<stdlib.h>
#include<time.h>

int izracunajDobitak(int);
void pocPoruka(void);
int main(){
	int pocIznos, broj, ulog, igrac, banka, brojIgara, brojac, dobitak, maxDobitak = 0,
maxIgrac, maxBanka, dobitak2 = 0, pombr = 0;
	srand(time(NULL));
	pocPoruka();
	do{
		printf("# Unesite pocetni iznos za igrace: ");
		scanf_s("%d", &pocIznos);
		if (pocIznos <= 0)
			printf("# Vrijednost pocetnog iznosa mora biti pozitivan cijeli
broj!\n");
		pombr++;
		if (pombr >= 10)return -1;
	}while (pocIznos <= 0);
	maxIgrac = pocIznos;
	maxBanka = pocIznos;
	pombr = 0;
	do{
		printf("# Unesite iznos uloga: ");
		scanf_s("%d", &ulog);
		if (ulog <= 0)
			printf("# Vrijednost uloga ne smije biti manja od 1!\n");
		if (ulog > pocIznos)
			printf("# Vrijednost uloga ne smije prelaziti vrijednost pocetnog
iznosa!\n");
		pombr++;
		if (pombr >= 10)return -1;
	}while (ulog <= 0 || ulog > pocIznos);
	igrac = pocIznos;
	banka = pocIznos;
	brojIgara = 0;
	printf("#########################################################################\n");
	printf("# Rezultat:");
	while (igrac >= ulog && banka > 0){
		brojIgara++;
		igrac -= ulog;
		banka += ulog;
		brojac = 1;
		do{
			broj = rand() & 1;
			if (broj == 0)
				brojac++;
```

```
		}while (broj == 0);
		dobitak = izracunajDobitak(brojac);
		if (dobitak == 2) dobitak2++;
		if (dobitak > maxDobitak) maxDobitak = dobitak;
		if (igrac > maxIgrac) maxIgrac = igrac;
		if (banka > maxBanka) maxBanka = banka;
		igrac += dobitak;
		banka -= dobitak;
	}
	if (igrac > banka){
		printf("\n# Pobijedio je igrac!\n");
		if (banka < 0)
			printf("# Banka je ostala igracu duzna %d kn.\n", banka*-1);
		if (banka == 0)
			printf("# Banci je ostalo %d kn.\n", banka);
		printf("# Igrac ima %d kn.\n", igrac);
	}
	else{
		printf("\n# Pobijedila je banka!\n");
		printf("# Banka ima %d kn.\n", banka);
		printf("# Igracu je ostalo %d kn i ne moze nastaviti igru.\n", igrac);
	}
	if (igrac > banka) printf("# Najveci iznos banke tijekom igre bio je %d kn.\n",
maxBanka);
	else printf("# Najveci iznos igraca tijekom igre bio je %d kn.\n", maxIgrac);
	printf("# Igra je zavrsila nakon %d ponavljanja.\n", brojIgara);
	printf("# Najveci dobitak tijekom igre iznosio je %d kn.\n", maxDobitak);
	printf("# %.2f %% igara zavrsilo je na prvom bacanju.\n", (float)dobitak2 / brojIgara
* 100);
	printf("################################################################################\n");
	getchar();
	return 0;
}

int izracunajDobitak(int brojac){
	int i, dobitak = 1;
	for (i = 0; i < brojac; i++)
		dobitak *= 2;
	return dobitak;
}

void pocPoruka(){
	printf("################################################################################\n");
	printf("#\t\tSIMULACIJA PETROGRADSKOG PARADOKSA                    #");
	printf("\n#                                                                      #\n");
	printf("#Zavrsni rad, Osijek 2016.                                            #\n");
	printf("#Autor: Dominik Grabic                                                #\n");
	printf("################################################################################\n");
	}
```

8.2.Ispis modificiranog programa

```
#include<stdio.h>
#include<stdlib.h>
#include<time.h>
#define _CRT_SECURE_NO_DEPRECATE
#define MAX_TRAJANJE_IGRE 2400
#define BROJ_IGARA 100
#define POCETNO_STANJE 1000
#define ULOG 15

int izracunajDobitak(int);

int main(){
	int broj, igrac, banka, brojBacanja, brojIgara, brojac, dobitak, flag, ukupniDobitak,
pobjedaIgrac, pobjedaBanka, nerijeseno, ulog;
	float prosjecniDobitak;
	srand(time(NULL));
	FILE *f;//pokazivac na file
	errno_t err;
	err = fopen_s(&f, "test.txt", "a+");
	if (err == 0)printf("Datoteka je uspjesno otvorena\n");
	else printf("Pojavila se greska pri otvaranju datoteke\n");
	pobjedaIgrac = 0;
	pobjedaBanka = 0;
	nerijeseno = 0;
	ulog = ULOG;
	for (int i = 1; i <= BROJ_IGARA; i++){
		igrac = POCETNO_STANJE;
		brojBacanja = 0;
		brojIgara = 0;
		flag = 0;
		ukupniDobitak = 0;
		while (flag == 0 && igrac >= ULOG){
			brojIgara++;
			igrac -= ULOG;
			brojac = 1;
			do{
				broj = rand() & 1;
				if (broj == 0)
					brojac++;
				if (++brojBacanja >= MAX_TRAJANJE_IGRE)
					flag = 1;
			} while (broj == 0 && flag == 0);
			dobitak = izracunajDobitak(brojac);
			ukupniDobitak += dobitak;
			igrac += dobitak;
		}
		prosjecniDobitak = (float)ukupniDobitak / brojIgara;
		fprintf(f, "%d,", igrac);
		if (igrac > POCETNO_STANJE){
			fprintf(f, "1,");
```

```
                        pobjedaIgrac++;
                }
                else if (igrac == POCETNO_STANJE){
                        fprintf(f, "0,");
                        nerijeseno++;
                }

                else{
                        fprintf(f, "2,");
                        pobjedaBanka++;
                }
                fprintf(f, "%d,", brojIgara);
                fprintf(f, "%d,", brojBacanja);
                fprintf(f, "%.2f\n", prosjecniDobitak);
                /*fprintf(f, "%.2f,", prosjecniDobitak);
                fprintf(f,"%d\n",brojIgara);*/
        }
        fprintf(f, "%d,%d,%d,%d", pobjedaIgrac, pobjedaBanka, nerijeseno, ulog);
        printf("%d,%d,%d\n", pobjedaIgrac, pobjedaBanka, nerijeseno);
        fclose(f);
        return 0;
}

int izracunajDobitak(int brojac){
        int j, dobitak = 1;
        for (j = 0; j < brojac; j++)
                dobitak *= 2;
        return dobitak;
}
```