

UNIVERZITET U BEOGRADU
ELEKTROTEHNIČKI FAKULTET



**NAJBOLJE RACIONALNE APROKSIMACIJE REALNIH
BROJEVA**

Prvi projektni zadatak

Mentor:

Branko Malešević, prof. dr

Kandidat:

Filip Kojić 2023/3297

Beograd, Januar 2024.

SADRŽAJ

SADRŽAJ.....	2
1. POSTAVKA PROJEKTOG ZADATKA.....	3
2. PREGLED REŠENJA PROJEKTOG ZADATKA.....	4
3. TESTIRANJE REŠENJA PROJEKTOG ZADATKA	17
SPISAK SLIKA	20
LITERATURA.....	21

1. POSTAVKA PROJEKTOG ZADATAKA

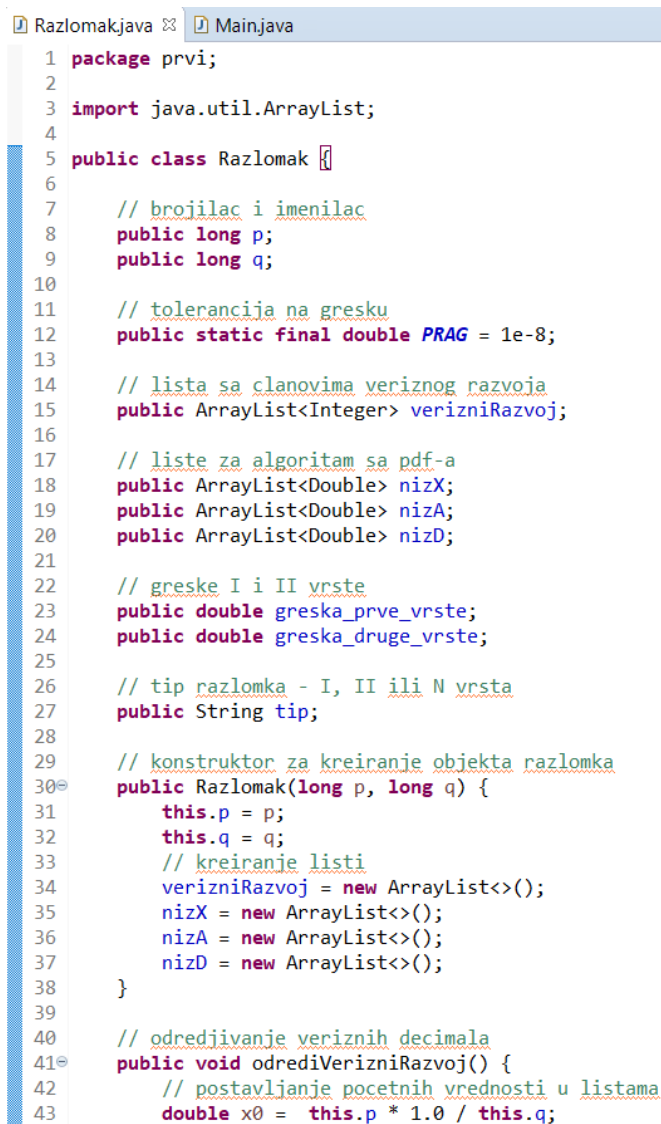
Projektni zadatak. Neka je dat pozitivan realan broj α sa konačnim decimalnim zapisom i neka su dati prirodni brojevi n i m , tako da $n < m$. Formirati niz razlomaka p/q takvih da za imenilac q važi $n \leq q \leq m$ (tj. $q = n, n + 1, \dots, m$) i pri tom imeniocu q pridružujemo brojilac p koji određujemo zaokruživanjem na najbliži prirodan broj proizvoda $\alpha \cdot q$. Predstaviti svaki razlomak p/q u obliku verižnog razlomka. U nizu razlomaka p/q izdvojiti:

- najbolje racionalne aproksimacije I vrste
- najbolje racionalne aproksimacije II vrste,
- sortirati sve razlomke p/q po uslovu minimalnosti apsolutne greške $|x - p/q|$ (tj. ε -ranga) .

Teorijska osnova za izradu ovog projektnog zadatka data je u materijalima sa predavanja profersora Maleševića. [1]

2. PREGLED REŠENJA PROJEKTOG ZADATKA

Programski jezik u kome je rađena implementacija ovog projektnog zadatka je Java (JavaSE-13). Implementacija se sastoji iz 2 klase, Razlomak.java i Main.java, čije ćemo detalje ukratko izložiti u nastavku. Razlomak.java je klasa koja služi za predstavljanje objekata koji predstavljaju razlomak formiran iz zadatog realnog broja a . Njena implementacija data je na sledecim slikama:



```
1 package prvi;
2
3 import java.util.ArrayList;
4
5 public class Razlomak {
6
7     // brojilac i imenilac
8     public long p;
9     public long q;
10
11     // tolerancija na gresku
12     public static final double PRAG = 1e-8;
13
14     // lista sa clanovima veriznog razvoja
15     public ArrayList<Integer> verizniRazvoj;
16
17     // liste za algoritam sa pdf-a
18     public ArrayList<Double> nizX;
19     public ArrayList<Double> nizA;
20     public ArrayList<Double> nizD;
21
22     // greske I i II vrste
23     public double greska_prve_vrste;
24     public double greska_druge_vrste;
25
26     // tip razlomka - I, II ili N vrsta
27     public String tip;
28
29     // konstruktor za kreiranje objekta razlomka
30     public Razlomak(long p, long q) {
31         this.p = p;
32         this.q = q;
33         // kreiranje listi
34         verizniRazvoj = new ArrayList<>();
35         nizX = new ArrayList<>();
36         nizA = new ArrayList<>();
37         nizD = new ArrayList<>();
38     }
39
40     // odredjivanje veriznih decimala
41     public void odrediVerizniRazvoj() {
42         // postavljanje pocetnih vrednosti u listama
43         double x0 = this.p * 1.0 / this.q;
```

Slika 2.1. Razlomak.java – deo 1

```

39
40 // odredjivanje veriznih decimala
41 public void odrediVerizniRazvoj() {
42     // postavljanje pocetnih vrednosti u listama
43     double x0 = this.p * 1.0 / this.q;
44     double a0 = Math.floor(x0);
45     double d0 = x0 - a0;
46     nizX.add(x0); nizA.add(a0); nizD.add(d0);
47     verizniRazvoj.add((int)a0);
48
49     // racunanje veriznih decimala
50     int i = 0;
51     while (nizD.get(i) > PRAG) {
52         i++;
53         double x = 1.0 / nizD.get(i - 1);
54         double a = Math.floor(x);
55         double d = x - a;
56         nizX.add(x); nizA.add(a); nizD.add(d);
57         verizniRazvoj.add((int) a);
58     }
59
60     // azuriranje pretposlednje verizne decimala ukoliko je
61     // poslednja decimala jednaka 1
62     int brojClanova = verizniRazvoj.size();
63     if(brojClanova >= 2) {
64         int poslednjaDecimala = verizniRazvoj.get(brojClanova - 1);
65         if(poslednjaDecimala == 1) {
66             int pretposlednjaDecimala = verizniRazvoj.get(brojClanova - 2) + 1;
67             verizniRazvoj.set(brojClanova - 2, pretposlednjaDecimala);
68             verizniRazvoj.remove(brojClanova - 1);
69         }
70     }
71 }
72
73 // racunanje gresaka I i II vrste
74 public void izracunajGreske(double a) {
75     this.greska_prve_vrste = a - (this.p * 1.0 / this.q);
76     this.greska_druge_vrste = this.q * a - this.p;
77 }
78
79 // ispis clanova veriznog razvoja
80 public String ispisVerizniRazvoj() {
81     StringBuilder ispisBuilder = new StringBuilder();
82     ispisBuilder.append("F");

```

Slika 2.2. Razlomak.java – deo 2

```

79 // ispis članova verižnog razvoja
80 public String ispiVerizniRazvoj() {
81     StringBuilder ispisBuilder = new StringBuilder();
82     ispisBuilder.append("[");
83
84     for(int i = 0; i < verizniRazvoj.size(); i++) {
85         if(i == 0) {
86             ispisBuilder.append(verizniRazvoj.get(i));
87             if (verizniRazvoj.size() > 1) {
88                 ispisBuilder.append(" ");
89             }
90         } else {
91             ispisBuilder.append(verizniRazvoj.get(i));
92             if (i < (verizniRazvoj.size() - 1)) {
93                 ispisBuilder.append(", ");
94             }
95         }
96     }
97
98     ispisBuilder.append("]");
99     return ispisBuilder.toString();
100 }
101
102 // metoda za trazenje najveceg zajednickog delioca
103 // za skracivanje razlomka ukoliko je to potrebno
104 public long nzd() {
105     long qq = q;
106     long pp = p;
107     while (qq != 0) {
108         long t = qq;
109         qq = pp % qq;
110         pp = t;
111     }
112     return pp;
113 }
114
115 // ispis razlomka kao p/q
116 public String toString() {
117     return this.p + "/" + this.q;
118 }
119
120 }
121

```

Slika 2.3. Razlomak.java – deo 3

Klasa Razlomak je Java klasa koja predstavlja racionalni broj sa brojiocem (p) i imeniocem (q). Klasa sadrži različite metode i atribute koji omogućavaju manipulaciju i analizu razlomaka. Evo kratkog opisa svakog dela klase:

1. Polja:

- **public long p:** Brojilac razlomka.
- **public long q:** Imenilac razlomka.
- **public static final double PRAG = 1e-8:** Konstanta koja određuje toleranciju na grešku pri izračunavanju verižnih decimala.
- **public ArrayList<Integer> verizniRazvoj:** Lista koja sadrži članove verižnog razvoja razlomka.

- **public ArrayList<Double> nizX, nizA, nizD:** Liste koje se koriste u algoritmu za određivanje verižnih decimala.
- **public double greska_prve_vrste, greska_druge_vrste:** Vrednosti grešaka I i II vrste pri aproksimaciji realnog broja a sa razlomkom.
- **public String tip:** Tip razlomka (I, II ili N vrsta) u kontekstu aproksimacije realnog broja.

2. Konstruktor:

public Razlomak(long p, long q): Konstruktor koji inicijalizuje razlomak sa zadatim brojiocem i imeniocem i kreira liste za računanje verižnog razvoja.

3. Metode:

- **public void odrediVerizniRazvoj():** Izračunava verižni razvoj razlomka koristeći algoritam opisan u dokumentaciji.
- **public void izracunajGreske(double a):** Izračunava greške I i II vrste za dati realni broj a prema definicijama datim u dokumentaciji.
- **public String ispisiVerizniRazvoj():** Vraća string reprezentaciju verižnog razvoja razlomka.
- **public long nzd():** Pronalazi najveći zajednički delilac (NZD) brojioca i imenioca kako bi se razlomak skratio, ako je to moguće. Ovo je posebno korisno u situacijama kada je potrebno prepoznati i grupisati identične razlomke unutar liste, čak i kada su oni inicijalno predstavljeni u različitim, ali ekvivalentnim oblicima. Na primer, razlomci $\frac{2}{4}$ i $\frac{1}{2}$ će nakon skraćivanja biti prepoznati kao isti razlomak ($\frac{1}{2}$), što omogućava njihovo klasifikovanje u istu kategoriju u analizi koja se izvršava.
- **public String toString():** Vraća string reprezentaciju razlomka u obliku p/q .

Klasa Razlomak.java omogućava detaljnu analizu razlomaka, uključujući njihov verižni razvoj, greške pri aproksimaciji, i mogućnost skraćivanja razlomka. Ove funkcionalnosti su korisne u matematičkim analizama i algoritmima koji zahtevaju rad sa racionalnim brojevima.

Klasa Main.java je glavni deo Java aplikacije koja se bavi analizom racionalnih aproksimacija realnog broja a . Ova klasa sadrži metode za izračunavanje i klasifikaciju razlomaka u odnosu na njihovu sposobnost da aproksimiraju dati realan broj. Njena implementacija data je na sledećim slikama:

```

Razlomak.java Main.java
1 package prvi;
2
3 import java.util.ArrayList;
4 import java.util.Collections;
5 import java.util.HashSet;
6 import java.util.Scanner;
7
8 public class Main {
9
10     // realni broj
11     static double a;
12
13     // granicne vrednosti imenioca q
14     static int n;
15     static int m;
16
17     // razlomci p/q
18     public static ArrayList<Razlomak> razlomci = new ArrayList<>();
19
20     // lista sa clanovima veriznog razvoja broja a
21     public static ArrayList<Integer> verizniRazvoj = new ArrayList<>();
22
23     // liste za algoritam sa pdf-a
24     public static ArrayList<Double> nizX = new ArrayList<>();
25     public static ArrayList<Double> nizA = new ArrayList<>();
26     public static ArrayList<Double> nizD = new ArrayList<>();
27
28     // konvergente broja a
29     public static ArrayList<Razlomak> konvergente = new ArrayList<>();
30
31     // set sa konvergentama u vidu stringa
32     public static HashSet<String> setKonvergenti = new HashSet<>();
33     // set sa razlomcima prve vrste u vidu stringa
34     public static HashSet<String> setPrvaVrsta = new HashSet<>();
35
36     // aproksimacije I i II i N vrste
37     public static ArrayList<Razlomak> aproksimacije1 = new ArrayList<>();
38     public static ArrayList<Razlomak> aproksimacije2 = new ArrayList<>();
39     public static ArrayList<Razlomak> aproksimacijeN = new ArrayList<>();
40
41     // formiranje svih p/q razlomaka
42     public static void formirajRazlomke() {
43         System.out.println("*****");

```

Slika 2.4. Main.java – deo 1


```

41 // formiranje svih p/q razlomaka
42 public static void formirajRazlomke() {
43     System.out.println("*****");
44     System.out.println("Svi razlomci: ");
45     System.out.printf("%-10s %-25s", "RAZLOMAK", "VERIZNI RAZVOJ");
46     System.out.println();
47     for(int i = n; i <= m; i++) {
48         int q = i;
49         int p = (int) Math.round((double) a * q);
50         Razlomak razlomak = new Razlomak(p, q);
51         razlomak.odrediVerizniRazvoj();
52         razlomak.izracunajGreske(a);
53         razlomci.add(razlomak);
54         System.out.printf("%-10s %-25s", razlomak, razlomak.ispisiVerizniRazvoj());
55         System.out.println();
56     }
57 }
58
59 // odredjivanje veriznih decimala broja a
60 private static void odrediVerizniRazvoj() {
61     // postavljanje pocetnih vrednosti u listama
62     double x0 = a;
63     double a0 = Math.floor(x0);
64     double d0 = x0 - a0;
65     nizX.add(x0); nizA.add(a0); nizD.add(d0);
66     verizniRazvoj.add((int) Math.round(a0));
67
68     // racunanje veriznih decimala
69     for(int i = 0; i <= m - n; i++) {
70         if (nizD.get(i) <= Razlomak.PRAG) {
71             break;
72         }
73         double x = 1.0 / nizD.get(i); nizX.add(x);
74         double a = Math.floor(x); nizA.add(a);
75         double d = x - a; nizD.add(d);
76         verizniRazvoj.add((int) a);
77     }
78
79     // azuriranje pretposlednje verizne decimala ukoliko je
80     // poslednja decimala jednaka 1
81     int brojClanova = verizniRazvoj.size();
82     if (brojClanova >= 2) {
83         int posledniaDecimala = verizniRazvoj.get(brojClanova - 1);

```

Slika 2.5. Main.java – deo 2

```

79 // azuriranje pretposlednje verizne decimala ukoliko je
80 // poslednja decimala jednaka 1
81 int brojClanova = verizniRazvoj.size();
82 if (brojClanova >= 2) {
83     int poslednjaDecimala = verizniRazvoj.get(brojClanova - 1);
84     if (poslednjaDecimala == 1) {
85         int pretposlednjaDecimala = verizniRazvoj.get(brojClanova - 2) + 1;
86         verizniRazvoj.set(brojClanova - 2, pretposlednjaDecimala);
87         verizniRazvoj.remove(brojClanova - 1);
88     }
89 }
90
91 // ispis clanova veriznog razvoja
92 System.out.println("*****");
93 StringBuilder ispisBuilder = new StringBuilder();
94 ispisBuilder.append("[");
95
96 for(int i = 0; i < verizniRazvoj.size(); i++) {
97     if (i == 0) {
98         ispisBuilder.append(verizniRazvoj.get(i));
99         if (verizniRazvoj.size() > 1) {
100             ispisBuilder.append("; ");
101         }
102     } else {
103         ispisBuilder.append(verizniRazvoj.get(i));
104         if (i < (verizniRazvoj.size() - 1)) {
105             ispisBuilder.append(", ");
106         }
107     }
108 }
109 ispisBuilder.append("]");
110 System.out.println("Verizni razvoj broja a: " + ispisBuilder.toString());
111 }
112
113 // racunanje konvergenti broja a
114 public static void odrediKonvergente() {
115     // racunanje prve dve konvergente
116     long p0 = verizniRazvoj.get(0);
117     long q0 = 1;
118     konvergente.add(new Razlomak(p0, q0));
119     setKonvergenti.add(p0 + "/" + q0);
120     if(verizniRazvoj.size() < 2) {
121         // ispis konvergenti

```

Slika 2.6. Main.java – deo 3

```

113 // racunanje konvergenti broja a
114 public static void odrediKonvergente() {
115     // racunanje prve dve konvergente
116     long p0 = verizniRazvoj.get(0);
117     long q0 = 1;
118     konvergente.add(new Razlomak(p0, q0));
119     setKonvergenti.add(p0 + "/" + q0);
120     if(verizniRazvoj.size() < 2) {
121         // ispis konvergenti
122         System.out.println("*****");
123         System.out.print("Konvergente: ");
124         System.out.println(konvergente);
125         return;
126     }
127     long p1 = p0 * verizniRazvoj.get(1) + 1;
128     long q1 = verizniRazvoj.get(1);
129     konvergente.add(new Razlomak(p1, q1));
130     setKonvergenti.add(p1 + "/" + q1);
131
132     // racunanje ostalih konvergenti rekurentnom formulom
133     for (int j = 2; j < verizniRazvoj.size(); j++) {
134         long p = konvergente.get(j - 1).p * verizniRazvoj.get(j) + konvergente.get(j - 2).p;
135         long q = konvergente.get(j - 1).q * verizniRazvoj.get(j) + konvergente.get(j - 2).q;
136         konvergente.add(new Razlomak(p, q));
137         setKonvergenti.add(p + "/" + q);
138     }
139
140     // ispis konvergenti
141     System.out.println("*****");
142     System.out.print("Konvergente: ");
143     System.out.println(konvergente);
144 }
145
146 // podela razlomaka na aproksimacije I, II i N vrste
147 public static void odrediAproksimacije() {
148     for(int i = 0; i < razlomci.size(); i++) {
149         Razlomak razlomak = razlomci.get(i);
150         // proveru da li je razlomak nakon skracivanja u konvergentama
151         long nzd = razlomak.nzd();
152         long p = razlomak.p / nzd;
153         long q = razlomak.q / nzd;
154         String s = p + "/" + q;
155         // konvergente su aproksimacije II vrste

```

Slika 2.7. Main.java – deo 4

```

146 // podela razlomaka na aproksimacije I, II i N vrste
147 public static void odrediAproksimacije() {
148     for(int i = 0; i < razlomci.size(); i++) {
149         Razlomak razlomak = razlomci.get(i);
150         // proverda da li je razlomak nakon skracivanja u konvergentama
151         long nzd = razlomak.nzd();
152         long p = razlomak.p / nzd;
153         long q = razlomak.q / nzd;
154         String s = p + "/" + q;
155         // konvergente su aproksimacije II vrste
156         if(setKonvergenti.contains(s)) {
157             aprokcimacije2.add(razlomak);
158             razlomak.tip = "II";
159             continue;
160         }
161
162         if(setPrvaVrsta.contains(s)) {
163             aprokcimacije1.add(razlomak);
164             razlomak.tip = "I";
165             continue;
166         }
167
168         // proverda da li je manja greska nego u prethodnicima
169         double apsGreska = Math.abs(razlomak.greska_prve_vrste);
170         boolean najboljaAproksimacija = true;
171         for (int j = i - 1; j >= 0; j--) {
172             Razlomak prethodni = razlomci.get(j);
173             double apsGreskaPrethodni = Math.abs(prethodni.greska_prve_vrste);
174             if(apsGreskaPrethodni <= apsGreska) {
175                 najboljaAproksimacija = false;
176                 break;
177             }
178         }
179         if(najboljaAproksimacija) {
180             aprokcimacije1.add(razlomak);
181             setPrvaVrsta.add(s);
182             razlomak.tip = "I";
183             continue;
184         }
185         aprokcimacijeN.add(razlomak);
186         razlomak.tip = "N";
187     }
188 }

```

Slika 2.8. Main.java – deo 5

```

Razlomak.java Main.java
190 // sortiranje i ispis svih vrsta aproksimacija
191 private static void sortirajIspisiAproksimacije() {
192     // sortiranje aproksimacija po rastućoj vrednosti odstupanja od a
193     Collections.sort(aprokcimacije1, (r1, r2) ->
194         Double.compare(Math.abs(r1.greska_prve_vrste), Math.abs(r2.greska_prve_vrste))
195     );
196     Collections.sort(aprokcimacije2, (r1, r2) ->
197         Double.compare(Math.abs(r1.greska_prve_vrste), Math.abs(r2.greska_prve_vrste))
198     );
199     Collections.sort(aprokcimacijeN, (r1, r2) ->
200         Double.compare(Math.abs(r1.greska_prve_vrste), Math.abs(r2.greska_prve_vrste))
201     );
202     // ispisivanje aproksimacija
203     System.out.println("*****");
204     System.out.println("Aproksimacije I vrste sortirane rastuće po odstupanju: ");
205     System.out.printf("%-10s %-25s %-35s", "RAZLOMAK", "VERIZNI RAZVOJ", "ODSTUPANJE");
206     System.out.println();
207     for(int i = 0; i < aprokcimacije1.size(); i++) {
208         Razlomak razlomak = aprokcimacije1.get(i);
209         System.out.printf("%-10s %-25s %-35s", razlomak, razlomak.ispisiVerizniRazvoj(), razlomak.greska_prve_vrste);
210         System.out.println();
211     }
212
213     System.out.println("*****");
214     System.out.println("Aproksimacije II vrste sortirane rastuće po odstupanju: ");
215     System.out.printf("%-10s %-25s %-35s", "RAZLOMAK", "VERIZNI RAZVOJ", "ODSTUPANJE");
216     System.out.println();
217     for(int i = 0; i < aprokcimacije2.size(); i++) {
218         Razlomak razlomak = aprokcimacije2.get(i);
219         System.out.printf("%-10s %-25s %-35s", razlomak, razlomak.ispisiVerizniRazvoj(), razlomak.greska_prve_vrste);
220         System.out.println();
221     }
222
223     System.out.println("*****");
224     System.out.println("Aproksimacije N vrste sortirane rastuće po odstupanju: ");
225     System.out.printf("%-10s %-25s %-35s", "RAZLOMAK", "VERIZNI RAZVOJ", "ODSTUPANJE");
226     System.out.println();
227     for(int i = 0; i < aprokcimacijeN.size(); i++) {
228         Razlomak razlomak = aprokcimacijeN.get(i);
229         System.out.printf("%-10s %-25s %-35s", razlomak, razlomak.ispisiVerizniRazvoj(), razlomak.greska_prve_vrste);
230         System.out.println();
231     }
232 }

```

Slika 2.9. Main.java – deo 6

```

234 // sortiranje i ispis svih razlomaka
235 public static void sortirajIspisiRazlomke() {
236     // sortiranje razlomaka po rastucoj vrednosti odstupanja
237     Collections.sort(razlomci, (r1, r2) ->
238         Double.compare(Math.abs(r1.greska_prve_vrste), Math.abs(r2.greska_prve_vrste))
239     );
240     // ispisivanje razlomaka
241     System.out.println("*****");
242     System.out.println("Svi razlomci sortirani rastuce po odstupanju: ");
243     System.out.printf("%-10s %-25s %-35s %-5s", "RAZLOMAK", "VERIZNI RAZVOJ", "ODSTUPANJE", "TIP");
244     System.out.println();
245     for(int i = 0; i < razlomci.size(); i++) {
246         Razlomak razlomak = razlomci.get(i);
247         System.out.printf("%-10s %-25s %-35s %-5s", razlomak,
248             razlomak.ispisiVerizniRazvoj(), razlomak.greska_prve_vrste, razlomak.tip);
249         System.out.println();
250     }
251 }
252
253 public static void main(String[] args) {
254     Scanner scanner = new Scanner(System.in);
255     System.out.println("*****");
256
257     // unos a
258     while (true) {
259         System.out.print("Unesite realan broj: ");
260         if (scanner.hasNextDouble()) {
261             a = scanner.nextDouble();
262             break;
263         } else {
264             System.out.println("Unesite validan realan broj.");
265             scanner.next();
266         }
267     }
268
269     // unos n
270     while (true) {
271         System.out.print("Unesite pozitivan ceo broj n: ");
272         if (scanner.hasNextInt()) {
273             n = scanner.nextInt();
274             if (n > 0) {
275                 break;
276             } else {

```

Slika 2.10. Main.java – deo 7

```

269 // unos n
270 while (true) {
271     System.out.print("Unesite pozitivan ceo broj n: ");
272     if (scanner.hasNextInt()) {
273         n = scanner.nextInt();
274         if (n > 0) {
275             break;
276         } else {
277             System.out.println("Broj n mora biti veci od 0.");
278         }
279     } else {
280         System.out.println("Unesite validan pozitivan ceo broj.");
281         scanner.next(); // ocisti nevazeci unos
282     }
283 }
284
285 // unos m
286 while (true) {
287     System.out.print("Unesite pozitivan ceo broj m veci od n: ");
288     if (scanner.hasNextInt()) {
289         m = scanner.nextInt();
290         if (m > n) {
291             break;
292         } else {
293             System.out.println("Broj m mora biti veci od broja n.");
294         }
295     } else {
296         System.out.println("Unesite validan pozitivan ceo broj.");
297         scanner.next(); // ocisti nevazeci unos
298     }
299 }
300
301 formirajRazlomke();
302 odrediVerizniRazvoj();
303 odrediKonvergente();
304 odrediAproksimacije();
305 sortirajIspisiAproksimacije();
306 sortirajIspisiRazlomke();
307
308 scanner.close();
309 }
310 }
311

```

Slika 2.11. Main.java – deo 8

1. Promenljive klase:

- **public static double a:** Realan broj koji se aproksimira.
- **public static int n, m:** Donja i gornja granica za imenilac q.
- **public static ArrayList<Razlomak> razlomci:** Lista svih razlomaka koji se formiraju i analiziraju.
- **public static ArrayList<Razlomak> verizniRazvoj, nizX, nizA, nizD:** Liste koje se koriste za izračunavanje verižnog razvoja broja a.
- **public static ArrayList<Razlomak> konvergente:** Lista konvergenti broja a.
- **public static HashSet<String> setKonvergenti, setPrvaVrsta:** Skupovi setKonvergenti i setPrvaVrsta se koriste za efikasno praćenje i klasifikaciju razlomaka koji su već identifikovani kao konvergente ili kao najbolje aproksimacije prve vrste. Korišćenjem string reprezentacija razlomaka, program može brzo da proveri da li je trenutni razlomak

već poznat i da ga svrsta u odgovarajuću kategoriju bez potrebe za ponovnim izračunavanjem ili poređenjem..

- **public static ArrayList<Razlomak aprokcimacije1, aprokcimacije2, aprokcimacijeN:** Liste aproksimacija I, II i N vrste.

2. Metode:

- **public static void formirajRazlomke():** Formira sve razlomke p/q u datom opsegu i izračunava njihov verižni razvoj i greške.
- **public static void odrediVerizniRazvoj():** Izračunava verižni razvoj realnog broja a .
- **public static void odrediKonvergente():** Izračunava konvergente za realan broj a koristeći verižni razvoj.
- **public static void odrediAproksimacije():** Klasifikuje razlomke u aproksimacije I, II ili N vrste na osnovu njihovih grešaka i odnosa sa konvergentima.
- **public static void sortirajIspisiAproksimacije():** Rastuće sortira aproksimacije po veličini greške prve vrste i ispisuje ih.
- **public static void sortirajIspisiRazlomke():** Rastuće sortira sve razlomke po veličini greške prve vrste i ispisuje ih zajedno sa njihovim tipom.
- **public static void main(String[] args):** Ova metoda je ulazna tačka programa. Ona upravlja unosom korisnika za realan broj a i granice n i m , a zatim poziva ostale metode da izvrše izračunavanja i ispišu rezultate.

Klasa Main koristi klasu Razlomak za reprezentaciju i manipulaciju razlomaka, kao i za izračunavanje verižnog razvoja i grešaka aproksimacija. Sve u svemu, klasa Main je glavni deo aplikacije koja koordinira proces aproksimacije realnog broja i prikazuje rezultate korisniku.

3. TESTIRANJE REŠENJA PROJEKTOG ZADATKA

U nastavku će biti izloženo testiranje rešenja projektnog zadatka sa tri različita seta ulaznih podataka.

1. Test primer 1:

```
Markers Properties Servers Data Source Explorer Snippets Console
<terminated> Main (1) [Java Application] C:\Program Files\Java\jdk-14.0.1\bin\javaw.exe (Nov 5, 2023, 12:04:50 PM – 12:04:59 PM)
*****
Unesite realan broj: 3.1415926
Unesite pozitivan ceo broj n: 1
Unesite pozitivan ceo broj m veci od n: 10
*****
Svi razlomci:
RAZLOMAK VERIZNI RAZVOJ
3/1 [3]
6/2 [3]
9/3 [3]
13/4 [3; 4]
16/5 [3; 5]
19/6 [3; 6]
22/7 [3; 7]
25/8 [3; 8]
28/9 [3; 9]
31/10 [3; 10]
*****
Verizni razvoj broja a: [3; 7, 15, 1, 243, 1, 1, 9, 1, 1, 4]
*****
Konvergente: [3/1, 22/7, 333/106, 355/113, 86598/27565, 86953/27678, 173551/55243, 1648912/524865, 1822463/580108, 3471375/1104973, 15707963/5000000]
*****
Aproksimacije I vrste sortirane rastuce po odstupanju:
RAZLOMAK VERIZNI RAZVOJ ODSTUPANJE
19/6 [3; 6] -0.025074066666666645
16/5 [3; 5] -0.058407400000000011
13/4 [3; 4] -0.108407399999999993
*****
Aproksimacije II vrste sortirane rastuce po odstupanju:
RAZLOMAK VERIZNI RAZVOJ ODSTUPANJE
22/7 [3; 7] -0.0012645428571427253
3/1 [3] 0.141592600000000007
6/2 [3] 0.141592600000000007
9/3 [3] 0.141592600000000007
*****
Aproksimacije N vrste sortirane rastuce po odstupanju:
RAZLOMAK VERIZNI RAZVOJ ODSTUPANJE
25/8 [3; 8] 0.016592600000000007
28/9 [3; 9] 0.0304814888888888908
31/10 [3; 10] 0.041592599999999998
*****
Svi razlomci sortirani rastuce po odstupanju:
```

Slika 3.1. Test primer 1 – deo 1

```
*****
Svi razlomci sortirani rastuce po odstupanju:
RAZLOMAK VERIZNI RAZVOJ ODSTUPANJE TIP
22/7 [3; 7] -0.0012645428571427253 II
25/8 [3; 8] 0.016592600000000007 N
19/6 [3; 6] -0.025074066666666645 I
28/9 [3; 9] 0.0304814888888888908 N
31/10 [3; 10] 0.041592599999999998 N
16/5 [3; 5] -0.058407400000000011 I
13/4 [3; 4] -0.108407399999999993 I
3/1 [3] 0.141592600000000007 II
6/2 [3] 0.141592600000000007 II
9/3 [3] 0.141592600000000007 II
```

Slika 3.2. Test primer 1 – deo 2

2. Test primer 2:

```

Markers Properties Servers Data Source Explorer Snippets Console
<terminated> Main (1) [Java Application] C:\Program Files\Java\jdk-14.0.1\bin\javaw.exe (Nov 5, 2023, 2:38:49 PM - 2:38:53 PM)
*****
Unesite realan broj: 0.5849625007211561815
Unesite pozitivan ceo broj n: 7
Unesite pozitivan ceo broj m veci od n: 22
*****
Svi razlomci:
RAZLOMAK VERIZNI RAZVOJ
4/7 [0; 1, 1, 3]
5/8 [0; 1, 1, 1, 2]
5/9 [0; 1, 1, 4]
6/10 [0; 1, 1, 2]
6/11 [0; 1, 1, 5]
7/12 [0; 1, 1, 2, 2]
8/13 [0; 1, 1, 1, 1, 2]
8/14 [0; 1, 1, 3]
9/15 [0; 1, 1, 2]
9/16 [0; 1, 1, 3, 2]
10/17 [0; 1, 1, 2, 3]
11/18 [0; 1, 1, 1, 1, 3]
11/19 [0; 1, 1, 2, 1, 2]
12/20 [0; 1, 1, 2]
12/21 [0; 1, 1, 3]
13/22 [0; 1, 1, 2, 4]
*****
Verizni razvoj broja a: [0; 1, 1, 2, 2, 3, 1, 5, 2, 23, 2, 2, 1, 1, 55, 1, 4]
*****
Konvergente: [0/1, 1/1, 1/2, 3/5, 7/12, 24/41, 31/53, 179/306, 389/665, 9126/15601, 18641/31867, 46408/79335, 65049/111202, 111457/190537, 6195184/10590737, 6306641/10781274, 31421748/53715833]
*****
Aproksimacije I vrste sortirane rastuce po odstupanju:
RAZLOMAK VERIZNI RAZVOJ ODSTUPANJE
4/7 [0; 1, 1, 3] 0.01353392929258479
8/14 [0; 1, 1, 3] 0.01353392929258479
12/21 [0; 1, 1, 3] 0.01353392929258479
*****
Aproksimacije II vrste sortirane rastuce po odstupanju:
RAZLOMAK VERIZNI RAZVOJ ODSTUPANJE
7/12 [0; 1, 1, 2, 2] 0.0016291673878228163
6/10 [0; 1, 1, 2] -0.015037499278843791
9/15 [0; 1, 1, 2] -0.015037499278843791
12/20 [0; 1, 1, 2] -0.015037499278843791
*****
Aproksimacije N vrste sortirane rastuce po odstupanju:

```

Slika 3.3. Test primer 2 – deo 1

Aproksimacije N vrste sortirane rastuce po odstupanju:			
RAZLOMAK	VERIZNI RAZVOJ	ODSTUPANJE	
10/17	[0; 1, 1, 2, 3]	-0.0032727933964908917	
13/22	[0; 1, 1, 2, 4]	-0.005946590187934753	
11/19	[0; 1, 1, 2, 1, 2]	0.006015132300103532	
9/16	[0; 1, 1, 3, 2]	0.022462500721156187	
11/18	[0; 1, 1, 1, 1, 3]	-0.026148610389954974	
5/9	[0; 1, 1, 4]	0.029406945165600606	
8/13	[0; 1, 1, 1, 1, 2]	-0.030422114663459232	
6/11	[0; 1, 1, 5]	0.03950795526661077	
5/8	[0; 1, 1, 1, 2]	-0.04003749927884381	

Svi razlomci sortirani rastuce po odstupanju:			
RAZLOMAK	VERIZNI RAZVOJ	ODSTUPANJE	TIP
7/12	[0; 1, 1, 2, 2]	0.0016291673878228163	II
10/17	[0; 1, 1, 2, 3]	-0.0032727933964908917	N
13/22	[0; 1, 1, 2, 4]	-0.005946590187934753	N
11/19	[0; 1, 1, 2, 1, 2]	0.006015132300103532	N
4/7	[0; 1, 1, 3]	0.01353392929258479	I
8/14	[0; 1, 1, 3]	0.01353392929258479	I
12/21	[0; 1, 1, 3]	0.01353392929258479	I
6/10	[0; 1, 1, 2]	-0.015037499278843791	II
9/15	[0; 1, 1, 2]	-0.015037499278843791	II
12/20	[0; 1, 1, 2]	-0.015037499278843791	II
9/16	[0; 1, 1, 3, 2]	0.022462500721156187	N
11/18	[0; 1, 1, 1, 1, 3]	-0.026148610389954974	N
5/9	[0; 1, 1, 4]	0.029406945165600606	N
8/13	[0; 1, 1, 1, 1, 2]	-0.030422114663459232	N
6/11	[0; 1, 1, 5]	0.03950795526661077	N
5/8	[0; 1, 1, 1, 2]	-0.04003749927884381	N

Slika 3.4. Test primer 2 – deo 2

3. Test primer 3:

```
Markers Properties Servers Data Source Explorer Snippets Console
<terminated> Main (1) [Java Application] C:\Program Files\Java\jdk-14.0.1\bin\javaw.exe (Nov 5, 2023, 2:45:26 PM – 2:45:31 PM)
*****
Unesite realan broj: 2.7182818284
Unesite pozitivan ceo broj n: 3
Unesite pozitivan ceo broj m veci od n: 18
*****
Svi razlomci:
RAZLOMAK VERIZNI RAZVOJ
8/3 [2; 1, 2]
11/4 [2; 1, 3]
14/5 [2; 1, 4]
16/6 [2; 1, 2]
19/7 [2; 1, 2, 2]
22/8 [2; 1, 3]
24/9 [2; 1, 2]
27/10 [2; 1, 2, 3]
30/11 [2; 1, 2, 1, 2]
33/12 [2; 1, 3]
35/13 [2; 1, 2, 4]
38/14 [2; 1, 2, 2]
41/15 [2; 1, 2, 1, 3]
43/16 [2; 1, 2, 5]
46/17 [2; 1, 2, 2, 2]
49/18 [2; 1, 2, 1, 1, 2]
*****
Verizni razvoj broja a: [2; 1, 2, 1, 1, 4, 1, 1, 6, 1, 1, 8, 1, 1, 8, 2]
*****
Konvergente: [2/1, 3/1, 8/3, 11/4, 19/7, 87/32, 106/39, 193/71, 1264/465, 1457/536, 2721/1001, 23225/8544, 25946/9545, 49171/18089, 419314/154257, 887799/326603]
*****
Aproksimacije I vrste sortirane rastuce po odstupanju:
RAZLOMAK VERIZNI RAZVOJ ODSUPANJE
49/18 [2; 1, 2, 1, 1, 2] -0.003940393822222443
*****
Aproksimacije II vrste sortirane rastuce po odstupanju:
RAZLOMAK VERIZNI RAZVOJ ODSUPANJE
19/7 [2; 1, 2, 2] 0.003996114114285465
38/14 [2; 1, 2, 2] 0.003996114114285465
11/4 [2; 1, 3] -0.03171817160000012
22/8 [2; 1, 3] -0.03171817160000012
33/12 [2; 1, 3] -0.03171817160000012
8/3 [2; 1, 2] 0.05161516173333336
16/6 [2; 1, 2] 0.05161516173333336
24/9 [2; 1, 2] 0.05161516173333336
```

Slika 3.5. Test primer 3 – deo 1

```
Markers Properties Servers Data Source Explorer Snippets Console
<terminated> Main (1) [Java Application] C:\Program Files\Java\jdk-14.0.1\bin\javaw.exe (Nov 5, 2023, 2:45:31 PM – 2:45:36 PM)
*****
Aproksimacije II vrste sortirane rastuce po odstupanju:
RAZLOMAK VERIZNI RAZVOJ ODSUPANJE
19/7 [2; 1, 2, 2] 0.003996114114285465
38/14 [2; 1, 2, 2] 0.003996114114285465
11/4 [2; 1, 3] -0.03171817160000012
22/8 [2; 1, 3] -0.03171817160000012
33/12 [2; 1, 3] -0.03171817160000012
8/3 [2; 1, 2] 0.05161516173333336
16/6 [2; 1, 2] 0.05161516173333336
24/9 [2; 1, 2] 0.05161516173333336
*****
Aproksimacije N vrste sortirane rastuce po odstupanju:
RAZLOMAK VERIZNI RAZVOJ ODSUPANJE
30/11 [2; 1, 2, 1, 2] -0.008990898872727193
46/17 [2; 1, 2, 2, 2] 0.01239947545882325
41/15 [2; 1, 2, 1, 3] -0.015051504933333515
27/10 [2; 1, 2, 3] 0.0182818283999997
35/13 [2; 1, 2, 4] 0.025974136092307365
43/16 [2; 1, 2, 5] 0.030781828399999878
14/5 [2; 1, 4] -0.08171817159999994
*****
Svi razlomci sortirani rastuce po odstupanju:
RAZLOMAK VERIZNI RAZVOJ ODSUPANJE TIP
49/18 [2; 1, 2, 1, 1, 2] -0.003940393822222443 I
19/7 [2; 1, 2, 2] 0.003996114114285465 II
38/14 [2; 1, 2, 2] 0.003996114114285465 II
30/11 [2; 1, 2, 1, 2] -0.008990898872727193 N
46/17 [2; 1, 2, 2, 2] 0.01239947545882325 N
41/15 [2; 1, 2, 1, 3] -0.015051504933333515 N
27/10 [2; 1, 2, 3] 0.0182818283999997 N
35/13 [2; 1, 2, 4] 0.025974136092307365 N
43/16 [2; 1, 2, 5] 0.030781828399999878 N
11/4 [2; 1, 3] -0.03171817160000012 II
22/8 [2; 1, 3] -0.03171817160000012 II
33/12 [2; 1, 3] -0.03171817160000012 II
8/3 [2; 1, 2] 0.05161516173333336 II
16/6 [2; 1, 2] 0.05161516173333336 II
24/9 [2; 1, 2] 0.05161516173333336 II
14/5 [2; 1, 4] -0.08171817159999994 N
```

Slika 3.6. Test primer 3 – deo 2

SPISAK SLIKA

Slika 2.1. Razlomag.java – deo 1	4
Slika 2.2. Razlomag.java – deo 2	5
Slika 2.3. Razlomag.java – deo 3	6
Slika 2.4. Main.java – deo 1	8
Slika 2.5. Main.java – deo 2	9
Slika 2.6. Main.java – deo 3	10
Slika 2.7. Main.java – deo 4	11
Slika 2.8. Main.java – deo 5	12
Slika 2.9. Main.java – deo 6	13
Slika 2.10. Main.java – deo 7	14
Slika 2.11. Main.java – deo 8	15
Slika 3.1. Test primer 1 – deo 1	17
Slika 3.2. Test primer 1 – deo 2	17
Slika 3.3. Test primer 2 – deo 1	18
Slika 3.4. Test primer 2 – deo 2	18
Slika 3.5. Test primer 3 – deo 1	19
Slika 3.6. Test primer 3 – deo 2	19

LITERATURA

- [1] Branko J. Malešević, Najbolje racionalne aproksimacije (2023).pdf