

Univerzitet u Beogradu - Elektrotehnički fakultet  
Katedra za računarsku tehniku i informatiku, studijski program „Softversko inženjerstvo“

*Predmet:* Testiranje softvera (13S113TS)

*Nastavnici:* Prof. dr Dragan Bojić, doc. dr Dražen Drašković

*Ispitni rok:* Januar 2022.

*Datum:* 18.1.2022.

*Ime i prezime studenta\*:* \_\_\_\_\_

*Broj indeksa\*:* \_\_\_\_\_

*\*popunjava student*

*Ispit traje 2.5 sata, a u toku prvog sata nije dozvoljeno napuštanje ispita.*

*Upotreba literature ili posedovanje bilo kog komunikacionog uređaja nije dozvoljena tokom trajanja ispita i strogo je kažnjiva.*

<i>Zadatak 1</i> _____ /8	<i>Zadatak 4</i> _____ /12
<i>Zadatak 2</i> _____ /8	<i>Zadatak 5</i> _____ /12
<i>Zadatak 3</i> _____ /10	<i>Zadatak 6</i> _____ /10
<b><i>Ukupno na ispitu:</i> _____ /60</b>	<b><i>Ukupno na domaćem:</i> _____ /40</b>

*Rok u kome branite domaći\*:* \_\_\_\_\_ (primer: DZ1 Jan 2022, DZ2 Feb 2022)

<b>Ukupno:</b> _____ /100
<b>Ocena:</b> _____ ( _____ )

**Napomena:** Ukoliko u zadatku nešto nije dovoljno precizno definisano, student treba da uvede razumnu pretpostavku, da je uokviri (da bi se lakše prepoznala prilikom ocenjivanja) i da nastavi da izgrađuje preostali deo svog odgovora na temeljima uvedene pretpostavke. Kod pitanja koja imaju ponuđene odgovore treba **samo zaokružiti** jedan odgovor. Na ostala pitanja odgovarati **čitko, kratko i precizno.**

1. [8] Oglašivač putem veb sistema „Novi Dom“ treba da unese oglas kojim prodaje svoju nekretninu. On unosi naziv lokacije, odabira jednu od opcija za tip nekretnine (stan ili kuća), i sledeće podatke:

- kvadratura nekretnine, kao numeričku vrednost realnog tipa, u opsegu od 30.0 do 500.0 (sa zaokruživanjem na jednu decimalu vrednosti 0 ili 5);
- godina izgradnje, kao numeričku vrednost celobrojnog tipa, u opsegu 1930. do prethodne godine;
- sprat, kao numeričku vrednost celobrojnog tipa, u opsegu od 0 (prizemlje) do najviše kolika je ukupna spratnost;
- ukupna spratnost, kao numeričku vrednost celobrojnog tipa, u opsegu od 0 do 20;
- cena nekretnine, kao numeričku vrednost pozitivnog celog broja, sa najvišim iznosom od 5 miliona.

Naziv lokacije: Crveni krst - Mileševska novogradnja

Tip:  Stan  Kuća

Kvadratura: 82.5

Godina izgradnje: 2005

Sprat: 4

Ukupna spratnost: 7

Cena: 208 750

**Unesi oglas**

**Karakteristike**

<input type="checkbox"/> Terasa	<input type="checkbox"/> Podrum	<input checked="" type="checkbox"/> Internet
<input type="checkbox"/> Lođa	<input type="checkbox"/> Garaža	<input checked="" type="checkbox"/> Interfon
<input checked="" type="checkbox"/> Franc. balkon	<input type="checkbox"/> Sa baštom	<input type="checkbox"/> Telefon
<input checked="" type="checkbox"/> Lift	<input checked="" type="checkbox"/> Klima	

Korišćenjem metode graničnih slučajeva, na osnovu priloženog opisa veb sistema, napisati sve potrebne legalne i nelegalne klase za tražena numerička polja prilikom unošenja novog oglasa, kao i skup test primera koji je potrebno realizovati.

Rešenje:



2. [8] Data je C funkcija *bintr()* za binarno pretraživanje po ključu K sadržaja rastuće sortiranog vektora M od N elemenata. Rezultat je indeks elementa koji sadrži K, ili -1 ako traženje nije uspelo.

- a) Nacrtati graf toka kontrole i odrediti broj ciklomatske kompleksnosti.
- b) Navesti bazični skup putanja.
- c) Formirati test primere koji pokrivaju putanje iz b).

```
1. int bintr(int K, int M[], int N){
2.     int ID = 0;
3.     int IG = N - 1;
4.     while (ID <= IG) {
5.         IS = (ID + IG) / 2;
6.         if (K == M[IS])
7.             return IS;
8.         else if (K < M[IS])
9.             IG = IS;
10.        else ID = IS;
11.    }
12.    return -1;
13.}
```

Rešenje:

3. [10] Dat je sledeći kod u programskom jeziku Java, koji pronalazi proste brojeve u opsegu:

```

1 import java.util.Scanner;
2
3 public class Prime {
4     public static void main(String[] args) {
5         Scanner ulaz = new Scanner(System.in);
6         System.out.print("Unesite donju granicu? ");
7         int low = ulaz.nextInt();
8         System.out.print("Unesite gornju granicu? ");
9         int high = ulaz.nextInt();
10
11         while (low < high) {
12             if(checkPrimeNumber(low))
13                 System.out.print(low + " ");
14             ++low;
15         }
16         ulaz.close();
17     }
18     public static boolean checkPrimeNumber(int num) {
19         boolean flag = true;
20         for(int i = 2; i <= num/2; ++i) {
21             if(num % i == 0) {
22                 flag = false;
23                 break;
24             }
25         }
26         return flag;
27     }
28 }

```

a) [2] Kolika je pokrivenost odluka i iskaza, ako imamo realizovana sledeća dva test primera:

TP1: low = 32, high = 36

TP2: low = 0, high = 1

Pokrivenost odluka: \_\_\_\_\_

\_\_\_\_\_

Pokrivenost iskaza: \_\_\_\_\_

\_\_\_\_\_

b) [3] Napisati koje su definicije, c- i p-upotrebe u datom programu, kao i DU lanci, i definisati dodatne test primere koji bi doveli do maksimalne pokrivenosti odluka (odnosno 100%).

<u>Definicije</u>	<u>C-upotrebe</u>	<u>P-upotrebe</u>	<u>DU lanci</u>

Test primeri:

c) [3] Napisati sve LCSAJ za metodu *checkPrimeNumber(int num)* i napisati da li definisanim test primerima iz tačaka a) i b) su pokrivene sve sekvence?

Sekvence pisati kao uređene trojke [START, END, JUMP].

Rešenje:

d) [2] Kojim (nelegalnim) klasama ekvivalencije za ulazne podatke i test primerima biste dodatno proširili testiranje ovog programskog koda, tako da se spreči pojava neželjenog završetka programa od strane krajnjeg korisnika? Napisati takve klase i testove.

Da li bi i sam programski kod mogao da se poboljša sa dodatnim uslovima i kako, da bi se sprečio neželjeni završetak programa?

Rešenje:

*Prazna stranica*

4. [12] Date su sledeće klase u programskom jeziku Java. Potrebno je:

a) Nacrtati klasni dijagram.

b) Nacrtati **use/include** hijerarhiju za međuklasno testiranje. Eventualne povratne grane prikazati isprekidano.

c) Navesti redosled testiranja klasa prema rezultatu tačke b). Objasniti kako se pri tome eliminišu povratne grane.

```
public interface State {
    public void raiseUserUp();
    public void raiseUserDown();
    public void raisePosSensorUpperPosition();
}

abstract public class AbstractState implements State {
    protected StateMachine stateMachine;
    public AbstractState(StateMachine stateMachine) {
        this.stateMachine = stateMachine;
    }
    public void raiseUserUp() {}
    public void raiseUserDown() {}
    public void raisePosSensorUpperPosition() {}
}

public class Idle extends AbstractState {
    public Idle(StateMachine stateMachine) { super(stateMachine); }
    @Override
    public void raiseUserUp() {
        stateMachine.activateState(new MovingUp(stateMachine));
    }
}

public class MovingUp extends AbstractState {
    public MovingUp(StateMachine stateMachine) { super(stateMachine); }
    @Override
    public void raiseUserDown() {
        stateMachine.activateState(new Idle(stateMachine));
    }
    @Override
    public void raisePosSensorUpperPosition() {
        stateMachine.activateState(new Idle(stateMachine));
    }
}

public class StateMachine {
    public StateMachine() { activateState(new Idle(this)); }
    State activeState = null;
    public void activateState(State state) { activeState = state; }
    public State getActiveState() { return activeState; }
    public void raiseUserUp() {
        activeState.raiseUserUp();
    }
    public void raiseUserDown() {
        activeState.raiseUserDown();
    }
    public void raisePosSensorUpperPosition() {
        activeState.raisePosSensorUpperPosition();
    }
}
```



Rešenje:

5. [12] Dat je sledeći programski kod neke metode našeg glavnog programa (GP):

```
int trouglovi(int a, int b, int c){
    if(a <= 0 || b <= 0 || c <= 0) {
        return 4; //nevalidan
    }
    if(!(a + b > c && a + c > b && b + c > a)){
        return 4; //nevalidan
    }
    if (a == b && b == c){
        return 2; //jednakostranični
    }
    if (a == b || b == c || a == c) {
        return 1; //jednakokraki
    }
    return 3; //raznostrani
}
```

Dati su i sledeći mutanti prvog reda (izmene u odnosu na GP su obojene):

Mutant 1	Mutant 2
<pre>int trouglovi(int a, int b, int c){     if(a &lt;= 0    b &lt;= 0    c &lt;= 0) {         return 4; //nevalidan     }     if(!(a-b &gt; c &amp;&amp; a+c &gt; b &amp;&amp; b+c &gt; a)){         return 4; //nevalidan     }     if (a == b &amp;&amp; b == c){         return 2; //jednakostranični     }     if (a == b    b == c    a == c) {         return 1; //jednakokraki     }     return 3; //raznostrani }</pre>	<pre>int trouglovi(int a, int b, int c){     if(a &lt;= 0    b &lt;= 0    c &lt;= 0) {         return 4; //nevalidan     }     if(!(a+b &gt; c &amp;&amp; a+c &gt; b &amp;&amp; b+c &gt;=a)){         return 4; //nevalidan     }     if (a == b &amp;&amp; b == c){         return 2; //jednakostranični     }     if (a == b    b == c    a == c) {         return 1; //jednakokraki     }     return 3; //raznostrani }</pre>
<pre>int trouglovi(int a, int b, int c){     if(a &lt;= 0    b &lt;= 0    c &lt;= 0) {         return 4; //nevalidan     }     if(!(a*b &gt; c &amp;&amp; a+c &gt; b &amp;&amp; b+c &gt; a)){         return 4; //nevalidan     }     if (a == b &amp;&amp; b == c){         return 2; //jednakostranični     }     if (a == b    b == c    a == c) {         return 1; //jednakokraki     }     return 3; //raznostrani }</pre>	<pre>int trouglovi(int a, int b, int c){     if(a &lt;= 0    b &lt;= 0    c &lt;= 0) {         return 4; //nevalidan     }     if(!(a+b &gt; c &amp;&amp; a+c &gt; b &amp;&amp; b+c &gt; a)){         return 4; //nevalidan     }     if (a == b    b == c){         return 2; //jednakostranični     }     if (a == b    b == c    a == c) {         return 1; //jednakokraki     }     return 3; //raznostrani }</pre>
<pre>int trouglovi(int a, int b, int c){     if(a &lt;= 0    a &lt;= 0    c &lt;= 0) {         return 4; //nevalidan     }     if(!(a+b &gt; c &amp;&amp; a+c &gt; b &amp;&amp; b+c &gt; a)){         return 4; //nevalidan     }     if (a == b &amp;&amp; b == c){         return 2; //jednakostranični     }     if (a == b    b == c    a == c) {         return 1; //jednakokraki     }     return 3; //raznostrani }</pre>	<pre>int trouglovi(int a, int b, int c){     if(a &lt;= 0    b &lt;= 0    c &lt;= 0) {         return 4; //nevalidan     }     if(!(a+b &gt; c &amp;&amp; a+c &gt; b &amp;&amp; b+c &gt; a)){         return 4; //nevalidan     }     if (a == b &amp;&amp; b == c){         return 2; //jednakostranični     }     if (a == b    b == c    a &gt;= c) {         return 1; //jednakokraki     }     return 3; //raznostrani }</pre>

Mutant 7	Mutant 8
<pre>int trouglovi(int a, int b, int c){     if(a &lt;= 0    b &lt;= 0    c &lt;= 0) {         return 4; //nevalidan     }     if(!(a+b &gt; c &amp;&amp; a+c &gt; b &amp;&amp; b+c &gt; a)){         return 4; //nevalidan     }     if (a == b &amp;&amp; b == c){         return 2; //jednakostranični     }     if (a == b    b == c    a == c++) {         return 1; //jednakokraki     }     return 3; //raznostrani }</pre>	<pre>int trouglovi(int a, int b, int c){     if(a &lt;= 0    b &lt;= 0    c &lt;= 0) {         return 4; //nevalidan     }     if(!(a+b &gt; c &amp;&amp; a+c &gt; b &amp;&amp; b+c &gt; a)){         return 4; //nevalidan     }     if (a == b &amp;&amp; ++b == c){         return 2; //jednakostranični     }     if (a == b    b == c    a == c) {         return 1; //jednakokraki     }     return 3; //raznostrani }</pre>

a) [6] Za date test primere sa ulazom (a,b,c) napisati izlazne vrednosti glavnog programa, u prvom redu donje tabele. Takođe, za date test primere iz tabele, napisati izlaze za sve mutante, date u gornjoj tabeli.

Program/ Mutant	TP1 (0,0,0)	TP2 (1,1,3)	TP3 (2,2,2)	TP4 (2,2,3)	TP5 (2,3,4)	LIVE skup	KILL skup
GP (t)						{ svi mutanti }	/
M1 (t)							
M2 (t)							
M3 (t)							
M4 (t)							
M5 (t)							
M6 (t)							
M7 (t)							
M8 (t)							

Napisati formulu za mutacioni skor, kao i mutacioni skor nakon ovih izvršenih test primera:

MUTACIONI SKOR = ----- =

b) [3] Da li dobijeni mutacioni skor u tački a) može da se popravi? Predložiti test primere koji bi dodatno mogli da se realizuju, da bi test skor bio bolji:

c) [3] Napisati dva ekvivalentna mutanta, primenom dva različita mutaciona operatora:

Ekvivalentan mutant 1:	Ekvivalentan mutant 2:

6. [10] Dat je izgled interfejsa jednostavne aplikacije za slušanje muzike. Pesma izabrana u listi (e1) se prikazuje na polju *Song Track* zajedno sa statusom reprodukcije (prazno ili *playing* ili *paused*). Inicijalno je izabrana prva pesma u listi i uvek je tačno jedna stavka izabrana. Izabrana pesma može da se pokrene (e2), zaustavi (e3), ili pauzira (e4)/otpausira (e5) pri čemu se u desnom polju, pored imena pesme, ažurira status.



- Nacrtati odgovarajući graf događaja (EFG).
- Navesti test sekvence na osnovu rezultata tačke a).

Rešenje:

