Dežurni nastavnik: _____

---

Elektrotehnički fakultet u Beogradu
Katedra za računarsku tehniku i informatiku

*Predmet:* Testiranje softvera (SI3TS)
*Nastavnik:* doc. dr Dragan Bojić
*Asistent:* dipl. ing. Dražen Drašković
*Ispitni rok:* Septembar 2012.
*Datum:* 02.09.2012.

*Kandidat*:* _____

*Broj indeksa*:* _____

*Ispit traje 3 sata, prvih sat vremena nije dozvoljeno napuštanje ispita.*
*Upotreba literature nije dozvoljena.*

| | | | |
|---|---|---|---|
| *Zadatak 1* | _____/6 | *Zadatak 5* | _____/8 |
| *Zadatak 2* | _____/6 | *Zadatak 6* | _____/10 |
| *Zadatak 3* | _____/9 | *Zadatak 7* | _____/6 |
| *Zadatak 4* | _____/9 | *Zadatak 8* | _____/6 |

***Ukupno na ispitu:*** **_____/60** ***Ukupno na domaćem*:*** **_____/40**

***Rok u kome je odbranjen domaći*:*** **_____** (primer: januar 2012)

**Ukupno:** _____/100

**Ocena:** _____ ( _____ )

**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**. * popunjava student.

---

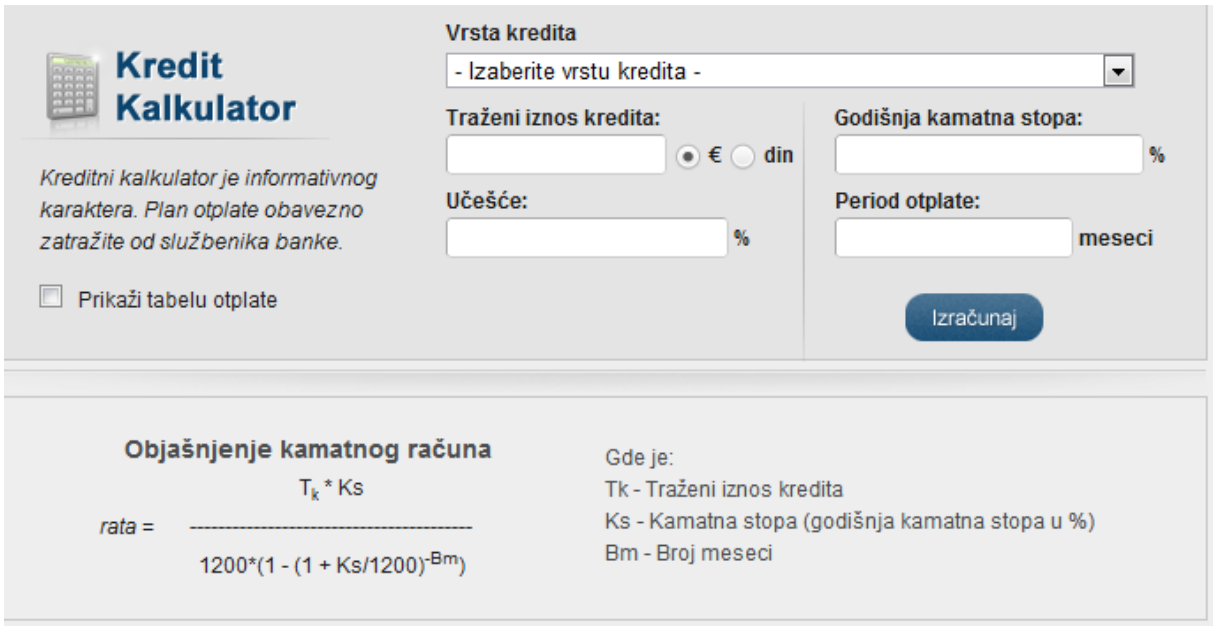
1. [6] Navesti i ukratko objasniti dve osnovne osobine za kriterijum selekcije test primera. Šta se može zaključiti ako kriterijum selekcije zadovoljava obe osnovne osobine?

2. [6] Posmatra se proces korišćenja dokumenata u višekorisničkom okruženju. Korisnici otvaraju dokumente radi čitanja, s tim što dok jedan korisnik čita fajl drugi korisnik može da ga otvori i čita. Kada neki korisnik želi da piše u fajl, sistem mu to ne dozvoljava, ako dokument već neko čita. Tek kada svi korisnici koji čitaju zatvore dokumente, dokument postaje dostupan, odnosno moguće je vršiti upis. Dok neko upisuje u dokument i menja ga, nikome nije dozvoljeno niti da ga čita niti da upisuje u njega.
    a. Identifikovati osnovna stanja dokumenta i nacrtati model (dijagram) stanja.
    b. Identifikovati test slučajeve ako se testiraju SVI pojedinačni prelazi sistema.
    c. Navesti skript koji treba napisati za manuelno testiranje završetka testiranja (prelaz *Close* u stanju koje odgovara čitanju dokumenta, od strane više čitalaca).

3. [9] Neka banka daje keš, stambene i auto kredite. Klijenti banke su zaposleni i penzioneri. Penzioneri sa primanjima ispod 30 000 dinara mogu uzimati samo keš kredite, a sa primanjima između 30 000 i 60 000 dinara mogu uzimati keš ili auto kredite. Penzioneri ne mogu imati primanja veća od 60 000 dinara. Zaposleni koji imaju primanja ispod 30 000 dinara i nemaju auto, mogu uzeti auto kredit. Zaposleni sa primanjima između 30 000 i 60 000 dinara mogu uzeti keš ili auto kredit. Zaposleni sa primanjima preko 60 000 mogu uzeti sve tri vrste kredita.

Nacrtati uzročno-posledični graf i prikazati sva ograničenja među uzrocima. Izvesti tabelu odlučivanja koristeći tehniku senzitizacije putanja (Basic Path Sensitization Technique).

b) Metodom *All-pairs* testirati formu ove veb aplikacije i napisati koliki je broj svih ulaznih kombinacija



4. [9] Neka je dat Java program koji vraća zbir vrednosti svih onih elemenata niza t, takvih da je `element >= l` i `element < u`. Pretpostavlja se da je na ulazu niz rastuće sortiran i da je `l < u`.

```
public int SearchSum(int t[], int l, int u){
   int k = 0;
   for(int i=0; i<t.length && t[i]<l; i++)
   k = i;
   return(Sum(t,k,u));
}

public int Sum(int t[], int i, int u){
   int tot = 0;
   while (i<t.length && t[i]<u){
      tot = tot+t[i]; i++;
   }
   return(tot);
}
```

a) Razviti test primere za dati program metodom pokrivanja svih elementarnih uslova (navesti prvo uslove, a zatim i konkretne test primere).
b) Razviti test primere za dati program metodom pokrivanja svih složenih uslova.

5. [8] Neka je dat algoritam za sortiranje direktnim umetanjem. Ulaz u proceduru predstavlja neuređeni niz ključeva a[1:n], koji ona treba da napravi uređenim.

```
for i = 2 to n do
   K = a[i]
   j = i - 1
   while (j > 0) and (a[j] > K) do
      a[j+1] = a[j]
      j = j - 1
   end_while
   a[j+1] = K
end_for
```

a) Testirati sve petlje u ovoj proceduri i detaljno obrazložiti.
Da li je ispravno napisana ova procedura?

b) U kom slučaju ova procedura daje najbolje performanse?
Nakon procesa testiranja da li možete da utvrdite kako da ovaj algoritam učinite efikasnijim?

6. [10] Razvijena su četiri modula, označena sa modul1 do modul4.
   a) Predložiti pristup za integraciono testiranje i obrazložiti. Naznačiti kako bi izgledali drajveri/stabovi u svakom koraku postupka integracije.
   b) Kako bi izgledala mešovita integracija *(Sandwich testing)*?

   Napomena: Podrazumevati da za svaki interfejs postoji odgovarajuća implementacija.

**MODUL 1**

```
public class Sortiranje {

  public static void sort(KolekcijaElemenata elementi,
                          Uredjenje uredjenje,
                          Zamena zamena) {
    for (int i = 0; i < elementi.brojElemenata() - 1; i++)
      for (int j = i + 1; j < elementi.brojElemenata(); j++) {
        Elem e1 = elementi.dohvatiElem(i);
        Elem e2 = elementi.dohvatiElem(j);
        if (uredjenje.ispred(e1, e2)) {
          zamena.zameni(e1, e2);
        }
      }
  }
}
```

**MODUL 2**

```
public interface Zamena {
  public void zameni(Elem e1, Elem e2);
}
```

**MODUL 3**

```
public interface KolekcijaElemenata {
  public Elem dohvatiElem(int index);
  public int brojElemenata();
}


public interface Elem {
}
```

**MODUL 4**

```
public interface Uredjenje {
  public boolean ispred(Elem e1, Elem e2);
}
```

7. [6] Neka je dat sledeći program u Javi. Odrediti sve LCSAJ za dati program. Pretpostaviti da su sve funkcije definisane i da vraćaju očekivani rezultat.

```
1.   Scanner sc = new Scanner (System.in);
2.   int br, a, b, c;
3.   System.out.println ("br?");
4.   br = sc.nextInt();
5.   System.out.println ("br?");
6.   a = sc.nextInt();
7.   b = sc.nextInt();
8.   while (brojac > 0) {
9.    if (x < y)
10.        x = razlika(x, y);
11.     else
12.        y = zbir(x, y);
13.     brojac = brojac - 1;
14.  }
15.  c = stepen(x, y);
16.  System.out.println ("Rezultat: " + c);
```

8. [6] Dati su sledeći program P, koji prima argumente sa komandne linije (koji se preslikavaju u vrednosti vektora argv, da prvi argument ide u argv[1], drugi u argv[2], itd, dok je argv[0] ime programa), skup test primera (TP1, TP2, TP3) i skup mutanata (M1, M2, M3):

```
1. main(int argc, char* argv[])
   {
2.    int r = 1, i;
3.    for (i = 2; i <= 3; )
4.          if (atoi(argv[i]) > atoi(argv[r]))
5.                r = i;
6.    printf("Value of the rank is %d \n", r);
7.    exit(0);
   }
```

| Test primer | Ulazni argumenti | Očekivani izlaz |
|---|---|---|
| TP1 | 1 2 3 | Value of the rank is 3 |
| TP2 | 1 2 1 | Value of the rank is 2 |
| TP3 | 3 1 2 | Value of the rank is 1 |

| Mutant | Promena |
|---|---|
| M1 | Menja liniju 4 u `if (atoi(argv[i]) >= atoi(argv[r]))` |
| M2 | Menja liniju 4 u `if (i > atoi(argv[r]))` |
| M3 | Menja liniju 4 u `if (atoi(argv[r]) > atoi(argv[r]))` |

a) Odrediti za svakog mutanta kako će se ponašati testovi.
b) Na osnovu zaključaka iz tačke a) poboljšati seriju testova dodavanjem jednog ili više novih testova.