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: Oktobar 2013.
Datum: 15.09.2013.

Kandidat:* _____

Broj indeksa:* _____

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

Zadatak 1	_____/4	*Zadatak 5*	_____/8
Zadatak 2	_____/8	*Zadatak 6*	_____/10
Zadatak 3	_____/9	*Zadatak 7*	_____/9
Zadatak 4	_____/12		

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

Rok u kome je odbranjen domaći*: **_____** (primer: januar 2013)

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. [4] Navesti elemente plana testiranja i ukratko ih objasniti.

2. [8] Dat je program koji (u funkciji main), za zadati niz stringova (bez duplikata), generiše sve permutacije tih stringova u leksikografskom poretku. Funkcija Successor() za zadatu permutaciju vraća prvu sledeću permutaciju. Na primer, ako je zadati niz { ant bat cow dog }, prvi poziv Successor će vratiti { ant bat dog cow }, a sledeći će vratiti { ant cow bat dog }.

a) Definisati invarijantu klase, preduslove i postuslove za metod Successor().
b) Napraviti automatski prediktor rezultata za metod Successor(), pri čemu se ne sme koristiti neposredna verifikacija izlaza, niti provere konzistencije dobijene pod a).

```
public class Permutacije {

    private String[] element;
    private int order;

    public Permutacije(String[] atoms) {
        element = atoms.clone();
        order = atoms.length;
    }

    public String toString() {
        StringBuilder result = new StringBuilder();
        result.append("{ ");
        for (int i = 0; i < order; ++i) {
            result.append(element[i]);
            result.append(" ");
        }
        result.append("}");
        return result.toString();
    }

    public Permutacije Successor() {
        Permutacije result = new Permutacije(element);
        int left, right;
        // Step #1 - Find left value
        left = result.order - 2;
        while ((result.element[left].compareTo(result.element[left + 1]))
            >= 0 && (left >= 1)) {
            --left;
        }
        if ((left == 0)
                && (element[left].compareTo(element[left + 1])) >= 0) {
            return null;
        }
```

```
        // Step #2 - find right; first value > left
        right = result.order - 1;

        while  (result.element[left].compareTo(result.element[right])  >=  0)
        {
            --right;
        }
        // Step #3 - swap [left] and [right]
        String temp = result.element[left];
        result.element[left] = result.element[right];
        result.element[right] = temp;
        // Step #4 - reverse order the tail
        int i = left + 1;
        int j = result.order - 1;
        while (i < j) {
            temp = result.element[i];
            result.element[i++] = result.element[j];
            result.element[j--] = temp;
        }
        return result;
    }

    public static void main(String[] args) {
        String[] atoms = new String[]{"ant", "bat", "cow", "dog"};
        System.out.println("In lexicographical order,
            all permutations are:\n");
        for (Permutacije p = new Permutacije(atoms); p != null;
            p = p.Successor()) {
                  System.out.println(p.toString());
        }
    }
}
```

3. [9] Neka je data komanda *Is* u Unix/Linux operativnom sistemu, koja prikazuje sadržaj nekog direktorijuma. Format *Is* komande je:

Is options [pathnames]

gde je *options* lista mogućih opcija, a *[pathnames]* je opciona lista imena direktorijuma sadržaja koji će biti prikazan. Pretpostaviti da spisak opcija obuhvata sledeće dve opcije: -a i -C. Opcija -a služi za prikazivanje svih fajlova, obuhvatajuću i fajlove koji su nevidljivi (*hidden*, i ti fajlovi počinju tačkom .). Opcija -C služi za prikazivanje izlaza u više kolona sa ulazima sortiranim rastuće. Obično je ovo podrazumevana (default) opcija.

Metodom klasa ekvivalencije, kombinovano sa graničnim slučajevima, izvršiti analizu svih test primera da testirate implementaciju P komande Is. Ne treba zaboraviti da fajlovi i direktorijumi služe kao ulaz za P. Za test primer je dovoljno navesti opis testa i kako on izgleda (na šta ukazuje: options, pathnames, sadržaj direktorijuma,...), nije neophodno navoditi sadržaj fajla ili direktorijuma. Klase ekvivalencije označiti sa E_1, E_2,... Označiti koji testovi obuhvataju granično testiranje.

4. [12] Neka je data sledeća metoda u programskom jeziku C++:

```
void CountDigits(int &VoCount, int&InCount) {
      char Digit;
      cin >> Digit;    //citanje Digit sa ulaza
      while((Digit >= 'A') && (Digit <= 'Z') && (InCount < MaxSize)){
            InCount++;
            if ((Digit == 'A') || (Digit == 'E') || (Digit == 'I') ||
               (Digit == 'O') || (Digit == 'U')){
                 VoCount++;
            }
            cin >> Digit;
      }
}
```

a) Nacrtati graf kontrole toka za datu metodu.

b) Napisati minimalan broj test primera tako da se pokriju:
   a. Svi iskazi i odluke
   b. Svi višestruki uslovi
   c. Sve putanje

   I napisati koji su to iskazi/odluke/uslovi/putanje.

c) Popuniti sledeću tabelu, ukoliko se neka od ponuđenih promenljiva javlja kao definicija ili upotreba u određenom čvoru grafa iz tačke pod a):

Promenljiva	Definicija	C-upotreba	P-upotreba
`Digit`			
`VoCount`			
`InCount`			

Napomena: čvorova u grafu označavati sa N_1, N_2, N_3,... N_i

5. [8] Neka je u programskom jeziku Java data sledeća metoda `convert (String broj)` koja konvertuje broj sa ulaza u binarni broj:

```
public String convert(String broj){
            int num=-1;
            result = "Uneti broj je van dozvoljenog opsega!";
            try{
                  num=Integer.parseInt(broj);
            }
            catch (Exception e){}

            if (num>=0 && num<=2147483647){
                  result="";
                  int exp = 0;
                  int temp = num;
                  while (temp > 0) {
                        exp++;
                        temp/=2;
                  }
                  exp -= 1;

                  while (exp >= 0) {
                        if (num >= Math.pow(2, exp)) {
                              result += "1";
                              num -= Math.pow(2, exp);
                        }
                        else result += "0";
                        exp-=1;
                  }
            }
            if (result=="") result="0";
            return result;
      }
```

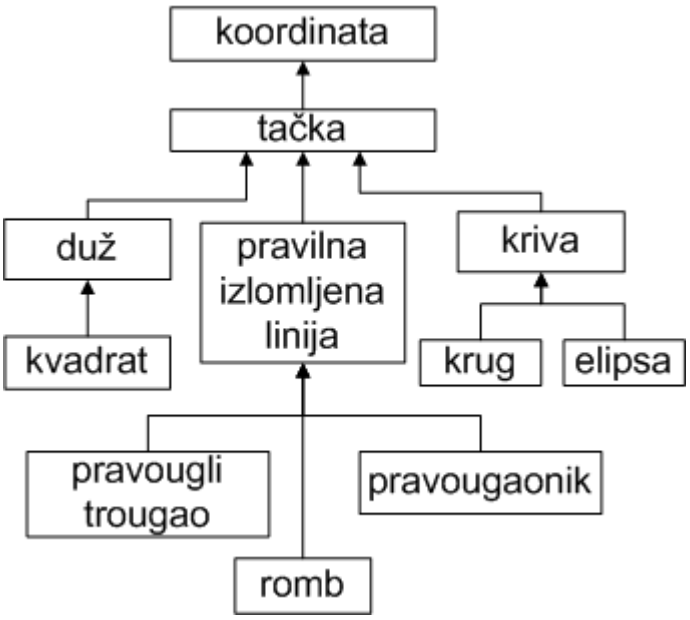
I neka su dati sledeći test primeri:

```
@Test
public void convertNeg(){
      assertEquals("Uneti   broj   je   van   dozvoljenog   opsega!",
      con.convert("-555"));
}
@Test
public void while1min0(){
      assertEquals("0",con.convert("0"));
}
@Test
public void while1max(){
      assertTrue("11111111111111111111111111111111".equals(con.convert
      ("2147483647")));
}
```

Napisati šta ovi test primeri pokrivaju, i napisati kolika je procentualno pokrivenost svake petlje korišćenjem ovih primera (loop-coverage).

Takođe napisati koji test primeri još nedostaju, a zatim dati njihovu realizaciju u JUnit-u.

6. [10] Neka je data aplikacija koja sadrži sledeće module:



a) Napisati korake po redosledu testiranja kod pristupa od vrha ka dnu i od dna ka vrhu. Naznačiti kako bi izgledali drajveri/stabovi u svakom koraku postupka integracije.
b) Kako bi izgledala mešovita integracija *(Sandwich testing)*? Navesti njene prednosti i nedostatke.

Napomena: Podrazumevati da za svaki modul postoji odgovarajuća implementacija i da su moduli povezani (kao izvedene klase ili kao implementacija određenog interfejsa).

7. [9] Neka je data sledeća HTML veb forma, kao na slici. U padajućoj listi postoje opcije: Red, Green, Blue, Yellow, Violet. Kada korisnik odabere jednu od ponuđenih opcija, boja pozadine se menja u izabranu. Polja ime, adresa, grad i država ograničena su na unos maksimalno 16 karaktera. Kada se klikne na Submit Form dugme, treba otvoriti novi prozor sa porukom uspešno ste uneli podatke.

Napisati šta sve treba testirati u ovoj formi koristeći neki alat za automatsko testiranje (GUI), npr. *Selenium*, koje događaje treba koristiti i napisati koji je minimalan skup test primera, da se proveri da ova stranica potpuno ispravno radi. Testove prikazati kratkim opisom i u koracima (prikazati u vidu tabele sa sledećim kolonama *Command-Target-Value* npr. Click-RadioButton-True, verifyText-stranica2.html-true,...).