Elektrotehnicki fakultet Banjaluka

PROGRAMSKI JEZICI II

Ispitni rok: septembar 2005

Napomena: zadatak 4 rješava se 20 minuta na papiru. Zadaci 1, 2 i 3 rješavaju se 180 minuta na racunaru uz proizvoljnu literaturu. Na osnovu zadatka 1 vrši se validacija domaceg rada.

1. U jeziku Java, u objektno-orijentisanom maniru, napisati primitivni softvera za potrebe muzicke izdavacke kuce. Izdavacka kuca objavljuje muzicke sadržaje u formi albuma na razlicitim medijima – DVD, CD i MD. Na svakom od albuma nalazi se više zapisa. Zapisi mogu biti audio zapisi i video zapisi. Na DVD i CD albumima mogu se nalaziti i audio i video zapise, dok se na MD albumu mogu nalaziti samo audio zapisi. Svaki zapis ima svoj naziv i dužinu (trajanje). Video zapisi mogu postojati u razlicitim formatima. Za svaki od albuma vežu se i podaci o izvodacu – grupi ili pojedincu. Pored toga, svaki album ima kataloški broj i naziv.
   Napomena:
   Rezultat simulacije treba biti ispis svih izdatih albuma, koji podrazumijeva naziv albuma, dužinu (trajanje), podatke o izvodacu, spisak svih zapisa sa albuma, kao i format, ako je rijec o video zapisu.

2. Implementirati jednostavan SMTP server i SMTP klijent. SMTP klijenti e-mail poruke prosljeduju SMTP serveru. Prihvacene e-mail poruke (koje se sastoje od izvorišne e-mail adrese, odredišne e-mail adrese, naslova i tijela) SMTP server ce cuvati u datotekama. Komunikacija izmedu SMTP servera i SMTP klijenta odvija se prema sljedecem protokolu:

```
SMTP Server:  220 SMTP Server ispit.etfbl.net
klijent    :  HELO testuser.etfbl.net
SMTP Server:  250 hello ispit.etfbl.net
klijent    :  MAIL FROM:testuser@etfbl.net
SMTP Server:  250 ok
klijent    :  RCPT TO:webmaster@etfbl.net
SMTP Server:  250 ok
klijent    :  DATA
SMTP Server:  354 Please start mail input, end with <CRLF>.<CRLF>
klijent    :  Subject: Ispitna poruka
klijent    :
klijent    :  Pozdrav,
klijent    :  ovo je ispitni primjer...
klijent    :
klijent    :  S postovanjem,
klijent    :  Testuser
klijent    :.
SMTP Server:  250 Mail queued for delivery.
klijent    :  QUIT
```

   Napomena:
   SMTP klijent poruku QUIT može poslati u bilo kom trenutku, cime se prekida konekcija sa serverom. Sve ostale poruke SMTP klijenta moraju biti poslate u predvidenom poretku, kao što je opisano protokolom. Potrebno je implmentirati i sljedece greške koje se mogu javiti u komunikaciji:

`SMTP Server: 501 user@nonexisting.com... Sender domain must exist`
U slucaju kada domen pošiljaoca ne postoji, tj. kada domen pošiljaoca nije etfbl.net.

`SMTP Server: 502 unimplemented`
U slucaju kada SMTP klijent pošalje SMTP serveru poruku koja nije predvidena protokolom.

`SMTP Server: 503 (No sender) ili (No recipient)`
U situaciji kada redoslijed poruka nije u skladu sa predvidenim protokolom.

3. Napisati aplikaciju Adresar koristeci CORBA tehnologiju. U adresaru se cuvaju imena osoba i njihove e-mail adrese. Ova aplikacija treba da omoguci unos novih zapisa (ime i njemu odgovarajuca e-mail adresa), pretragu po imenu i e-mail adresi. Potrebno je realizovati i izuzetke koji se mogu javiti prilikom unosu novog zapisa i prilikom pretrage. Prilikom pokretanja aplikacije smatra se da u adresaru ne postoji nijedan zapis. Na klijentskoj strani potrebno je realizovati meni putem kog ce korisnik birati odgovarajuce funkcionalnosti aplikacije.
   Napomena:
   Prilikom pretrage, za zadato ime aplikacija treba da vrati odgovarajucu e-mail adresu i obrnuto.

4. Napisati izlaz sljedeceg programa:

```
////////////// A.java ////////////
package a;

public class A {

    protected static int i = 1;

    public A() {
        System.out.println("konstruktor A");
    }

    public int metoda() {
        System.out.println(i++);
        return i;
    }

    public int metoda(int i) {
        System.out.println(--i);
        return i;
    }
}
```

```
////////////// B.java //////////////
import a.A;

public class B extends A {

    static int i = 1;

    B() {
        System.out.println("konstruktor B");
    }

    public int metoda(int l) {
        System.out.println(i++);
        return l;
    }

    class D{
        D(){
            System.out.println("konstruktor D");
        }

        public int metoda() {
            System.out.println(i++);
            return i;
        }
    }

}

////////////// C.java //////////////
import a.A;
public class C extends B {
    protected C() {
        System.out.println("konstruktor C");
    }

    public int metoda(A a) {
        System.out.println(a.metoda());
        return i;
    }

    public int metoda(int i) {
        System.out.println(i--);
        return ++i;
    }

    public int metoda() {
        System.out.println(i);
        return i++;
    }
}
```

```
////////////// Test.java ////////////
import a.A;
public class Test {


    public static void main(String[] args) {
        A a = new A();
        a.metoda();
        C c = new C();
        B b = new B();
        c.metoda(b);
        A a2 = new A();
        a2.metoda();
        B.D d =  b.new D();
        a2.metoda();
        c.metoda(3);
        b.metoda(2);
        c.metoda();
        c.metoda();
        b.metoda();
        d.metoda();
    }
}
```