

Laboratorijske vežbe – dvočas br. 14 - blok nastava

Zadatak 1

Napisati javnu klasu **Računar** koja može da bude serijalizovana i ima:

- Privatni atribut **tipProcesora** – tip procesora računara predstavljen kao String.
- Privatni atribut **ramMemorija** – realan broj koji predstavlja RAM memoriju računara izraženu u GB
- Privatni atribut **ipAdresa** – IP adresa računara koja ga jedinstveno identifikuje (String).
- Odgovarajuće javne get i set metode za ove attribute. Nedoizvoljena vrednost za attribute tipProcesora i ipAdresa je null String, dok su za atribut ramMemorija nedozvoljene vrednosti nula i vrednosti manje od nule. U slučaju unosa nedozvoljenih vrednosti potrebno je baciti izuzetak sa odgovarajućom porukom.
- Redefinisano toString metodu klase Object koja vraća String sa svim podacima o računaru uz odgovarajuću poruku.
- Redefinisano equals metodu klase Object koja kao ulazni argument prima objekat klase Object. Prvo je potrebno proveriti da li je unet objekat klase Računar, pa ako nije, baciti izuzetak sa odgovarajućom porukom. Metoda vraća true ako je IP adresa datog računara jednaka IP adresi unetog računara, a u suprotnom false.

Napisati javnu klasu **RačunskiCentar** koja ima:

- Privatni atribut **računari** koji predstavlja listu objekata klase Računar. Ovu listu je potrebno odmah inicijalizovati.
- Javnu metodu koja kao ulazni argument dobija objekat klase Računar i unosi ga u listu. Računar se unosi samo ako u listi već ne postoji isti računar. U slučaju da u listi već postoji taj računar potrebno je baciti **PROVERAVANI** izuzetak sa odgovarajućom porukom.
- Javnu metodu koja u fajl „racunari_za_zamenu.out“ upisuje (serijalizuje) samo one računare iz liste kod kojih je kapacitet RAM memorije manji od 2GB i tip procesora nije “Core 2 Duo”.
- Javnu metodu koja sa tastature učitava podatke o jednom računaru i unosi ih u listu i to samo ako u listi već ne postoji isti računar. U slučaju neke greške pri unosu, metoda bi trebalo da uhvati svaki bačen izuzetak i na ekranu ispiše poruku o grešci koju taj izuzetak sadrži.

```
import java.io.Serializable;

public class Racunar implements Serializable{

    private String tipProcesora;
    private double ramMemorija;
    private String ipAdresa;

    public String getTipProcesora() {
        return tipProcesora;
    }
    public void setTipProcesora(String tipProcesora) {
        if (tipProcesora == null)
            throw new RuntimeException("Tip procesora ne sme biti null");

        this.tipProcesora = tipProcesora;
    }
    public double getRamMemorija() {
        return ramMemorija;
    }
    public void setRamMemorija(double ramMemorija) {
        if (ramMemorija <= 0)
            throw new RuntimeException("RAM memorija mora biti veka od nule");

        this.ramMemorija = ramMemorija;
    }
    public String getIpAdresa() {
        return ipAdresa;
    }
    public void setIpAdresa(String ipAdresa) {
        if (ipAdresa == null)
            throw new RuntimeException("IP adresa ne sme biti null");

        this.ipAdresa = ipAdresa;
    }
}
```

```

    }
    public String toString(){
        return "Procesor: "+tipProcesora+" RAM: "+ramMemorija+
            " IP adresa: "+ipAdresa;
    }
    public boolean equals(Object o){
        if (!(o instanceof Racunar))
            throw new RuntimeException("Morate uneti objekat klase Racunar");

        Racunar r = (Racunar)(o);

        if(ipAdresa.equals(r.getIpAdresa())) return true;
        else return false;
    }
}

import java.util.LinkedList;
import java.util.Scanner;
import java.io.*;

public class RacunskiCentar {

    private LinkedList<Racunar> racunari = new LinkedList<Racunar>();

    public void unesi(Racunar r) throws Exception {
        if (racunari.contains(r))
            throw new Exception("Taj racunar vec postoji u centru");

        racunari.add(r);
    }

    public void sastaviSpisakZaZamenu() {
        try {
            ObjectOutputStream out = new ObjectOutputStream(
                new BufferedOutputStream(new FileOutputStream(
                    "racunari_za_zamenu.out")));

            for (int i = 0; i < racunari.size(); i++)
                if (racunari.get(i).getRamMemorija() < 2 &&
                    !(racunari.get(i).getTipProcesora().equals("Core 2 Duo")))
                    out.writeObject(racunari.get(i));

            out.close();
        } catch (Exception e) {
            System.out.println(e.getMessage());
        }
    }

    public void unesiRacunar() {
        Scanner in = new Scanner(System.in);

        try {
            System.out.print("Unesite IP adresu:");
            String ipAdresa = in.nextLine();

            System.out.print("Unesite tip procesora:");
            String tipProcesora = in.nextLine();

            System.out.print("Unesite velicinu RAM memorije:");
            double ramMemorija = in.nextDouble();

            Racunar r = new Racunar();
            r.setIpAdresa(ipAdresa);
            r.setRamMemorija(ramMemorija);
            r.setTipProcesora(tipProcesora);

            // Moze da se pozove vec gotova metoda za unos
            // koja proverava da li u listi vec postoji isti
            // racunar
            unesi(r);
        } catch (Exception e) {
            System.out.println(e.getMessage());
        }
    }
}

```

Zadatak 2

Napraviti javnu klasu **Kontakt** koja ima:

- Privatni atribut imePrezime. Početna vrednost za ovaj atribut je null.
- Privatni atribut telefon. Početna vrednost za ovaj atribut je null.
- Privatni atribut email. Početna vrednost za ovaj atribut je null.
- Odgovarajuće javne get i set metode za ove atribute. Nedoovoljene vrednosti za sva tri atributa su null String-ovi. U slučaju unosa nedozvoljenih vrednosti potrebno je baciti izuzetak sa porukom “UNELI STE PRAZAN STRING”.
- Redefinisane equals metodu klase Object koja kao ulazni argument dobija objekat klase Object. Metoda prvo proverava da li je uneti objekat klase Kontakt, pa ako nije baca izuzetak. Metoda vraća true ako su vrednosti atributa imePrezime i telefon jednaki vrednostima istih atributa unetog objekta a false ako nisu.

Napraviti vizuelnu klasu **TelefonskiImenikGUI** koja izgleda kao na slici. Naslov prozora bi trebalo da bude “Telefonski imenik”. Podesiti grafički interfejs tako da se, u toku rada aplikacije, ne mogu menjati dimenzije forme.

- Klasa TelefonskiImenikGUI bi trebalo da sadrži privatni atribut **kontakti** koji predstavlja listu objekata klase Kontakt. Odmah inicijalizovati listu.
- Kada se pritisne dugme “Dodaj”, preuzimaju se podaci o imenu i prezimenu, telefonu i e-mail adresi iz polja za unos, kreira se objekat klase Kontakt i unosi se u listu. Unošenje se vrši samo ako u listi već ne postoji isti kontakt.
- Kada se pritisne dugme “Sačuvaj”, svi podaci o kontaktima iz liste se upisuju u TEKSTUALNI fajl “telefonski_imenik.txt” i to tako da se podaci o svakom kontaktu upišu u poseban red.
- Kada se pritisne dugme “Obriši”, briše se sadržaj svih polja za unos.



Rešenje

- Napraviti klasu Kontakt

```
public class Kontakt {  
  
    private String imePrezime = null;  
    private String telefon = null;  
    private String email = null;  
  
    public String getImePrezime() {  
        return imePrezime;  
    }  
    public void setImePrezime(String imePrezime) {  
        if (imePrezime == null)  
            throw new RuntimeException("Uneli ste prazan String");  
  
        this.imePrezime = imePrezime;  
    }  
}
```

```

    }
    public String getTelefon() {
        return telefon;
    }
    public void setTelefon(String telefon) {
        if (telefon == null)
            throw new RuntimeException("Uneli ste prazan String");

        this.telefon = telefon;
    }
    public String getEmail() {
        return email;
    }
    public void setEmail(String email) {
        if (email == null)
            throw new RuntimeException("Uneli ste prazan String");

        this.email = email;
    }
    public boolean equals(Object o){
        if (!(o instanceof Kontakt))
            throw new RuntimeException("Greska");

        Kontakt k = (Kontakt)(o);

        if (k.getImePrezime().equals(imePrezime)&&
            k.getTelefon().equals(telefon))
            return true;
        else
            return false;
    }
}

```

- Kreirati nov grafički interfejs (vizuelnu klasu) sa imenom TelefonskiImenikGUI.
- Promeniti naslov na formi tako da bude “Telefonski imenik”.
- Potrebno je podesiti formu tako da ima null layout.
- Dodati dugme na formu sa imenom DodajDugme i naslovom “Dodaj”.
- Dodati dugme na formu sa imenom SacuvajDugme i naslovom “Sacuvaj”.
- Dodati dugme na formu sa imenom ObrisiDugme i naslovom “Obrisi”.
- Dodati labelu na formu sa bilo kojim imenom i naslovom “Ime i prezime”.
- Dodati labelu na formu sa bilo kojim imenom i naslovom “Telefon”.
- Dodati labelu na formu sa bilo kojim imenom i naslovom “Email”.
- Ispod labele na kojoj piše “Ime i prezime” dodati novo polje za unos sa imenom PoljeZaUnosImena.
- Ispod labele na kojoj piše “Telefon” dodati novo polje za unos sa imenom PoljeZaUnosTelefona.
- Ispod labele na kojoj piše “Email” dodati novo polje za unos sa imenom PoljeZaUnosEmaila.
- Podesiti atribut forme “resizable” na false uz pomoć Properties tab-a.
- Dodati direktno u kod klase TelefonskiImenikGUI privatni atribut kontakti:

```

private LinkedList<Kontakt> kontakti = new LinkedList<Kontakt>();

```

- Dodati event handler za dugme DodajDugme i u njega dodati sledeći kod:

```

Kontakt k = new Kontakt();

k.setImePrezime(PoljeZaUnosImena.getText());
k.setTelefon(PoljeZaUnosTelefona.getText());
k.setEmail(PoljeZaUnosEmaila.getText());

if(!kontakti.contains(k))
    kontakti.add(k);

```

- Dodati event handler za dugme SacuvajDugme i u njega dodati sledeći kod:

```

try {
    PrintWriter out=

```

```

        new PrintWriter(
            new BufferedWriter(
                new FileWriter("telefonski_imenik.txt"))));

        for(int i=0; i<kontakti.size();i++)
            out.println(kontakti.get(i));

        out.close();
    } catch (Exception e2) {
        System.out.println(e2.getMessage());
    }
}

```

- Dodati event handler za dugme ObrisiDugme i u njega dodati sledeći kod:

```

PoljeZaUnosImena.setText(null);
PoljeZaUnosTelefona.setText(null);
PoljeZaUnosEmaila.setText(null);

```

Zadatak 3

Napisati javnu klasu **Računar** koja ima:

- Privatni atribut **tipProcesora** – tip procesora računara predstavljen kao String.
- Privatni atribut **ramMemorija** – realan broj koji predstavlja RAM memoriju računara izraženu u GB
- Privatni atribut **veličinaDiska** – ceo broj koji predstavlja veličinu hard disk-a računara izraženu u GB
- Odgovarajuće javne get i set metode za ove attribute. Nedorazvoljena vrednost za atribut tipProcesora je null String i prazan String, dok su za preostala dva atributa nedozvoljene vrednosti nula i vrednosti manje od nule. U slučaju unosa nedozvoljenih vrednosti, baciti izuzetak sa odgovarajućom porukom.
- Redefinisatu toString metodu klase Object koja vraća String sa svim podacima o računaru uz odgovarajuću poruku.

Napisati javnu klasu **RačunskiCentar** koja ima:

- Privatni atribut **računari** koji predstavlja listu objekata klase Računar.
- Javni konstruktor bez argumenata koji inicijalizuje listu.
- Javnu metodu koja na ekranu ispisuje podatke o svim računarima. Ako je lista prazna, ova metoda bi trebalo da baci **PROVERAVANI** izuzetak sa odgovarajućom porukom.
- Javnu metodu koja iz data fajla „racunari.out“ učitava podatke o računarima i puni listu. Ukoliko lista nije prazna, pre unosa podataka iz fajla listu je potrebno obrisati. Podaci u fajlu „racunari.out“ su dati u formatu <tip_procesora><tab><ram_memorija><tab><velicina_diska>. Podaci o svakom računaru su dati u posebnom redu.
- Javnu metodu koja u tekstualni fajl „statistike.txt“ upisuje izveštaj koji kao prvu liniju teksta sadrži samo naslov “Računski centar - statistika”. Posle toga bi trebalo u izveštaj upisati tri podatka i to svaki u posebnom redu. Prvi podatak je broj računara iz računskog centra čiji je procesor tipa “Core 2 Duo”. Drugi podatak je prosečna veličina RAM memorije svih računara u centru. Treći podatak je ukupan broj računara u računskom centru.

```

public class Racunar{

    private String tipProcesora;
    private double ramMemorija;
    private int velicinaDiska;

    public String getTipProcesora() {
        return tipProcesora;
    }
    public void setTipProcesora(String tipProcesora) {
        if (tipProcesora == null || tipProcesora.equals(""))
            throw new RuntimeException("Tip procesora ne sme biti null niti prazan String");

        this.tipProcesora = tipProcesora;
    }
    public double getRamMemorija() {
        return ramMemorija;
    }
}

```

```

    public void setRamMemorija(double ramMemorija) {
        if (ramMemorija <= 0)
            throw new RuntimeException("RAM memorija mora biti veka od nule");

        this.ramMemorija = ramMemorija;
    }
    public int getVelicinaDiska() {
        return velicinaDiska;
    }
    public void setVelicinaDiska(int velicinaDiska) {
        if (velicinaDiska <= 0)
            throw new RuntimeException("Velicina diska mora biti veka od nule");

        this.velicinaDiska = velicinaDiska;
    }
    public String toString(){
        return "Procesor: "+tipProcesora+" RAM: "+ramMemorija+
            " Velicina diska: "+velicinaDiska;
    }
}

import java.util.LinkedList;
import java.io.*;

public class RacunskiCentar {

    private LinkedList<Racunar> racunari;

    public RacunskiCentar(){
        racunari = new LinkedList<Racunar>();
    }

    public void ispisi() throws Exception {
        if (racunari.size() == 0)
            throw new Exception("Lista racunara je prazna");

        for(int i=0; i<racunari.size();i++)
            System.out.println(racunari.get(i));
    }

    public void ucitajRacunare() {
        try {
            DataInputStream in = new DataInputStream( new BufferedInputStream(
                new FileInputStream("racunari.out")));

            racunari.clear();

            while(in.available() != 0){
                Racunar r = new Racunar();

                r.setTipProcesora(in.readUTF());
                in.readChar();

                r.setRamMemorija(in.readDouble());
                in.readChar();

                r.setVelicinaDiska(in.readInt());
                in.readChar();

                racunari.add(r);
            }

            in.close();
        } catch (Exception e) {
            System.out.println(e.getMessage());
        }
    }

    public void napraviIzvestaj() {
        try {
            PrintWriter out = new PrintWriter( new BufferedWriter(
                new FileWriter("statistike.txt")));

            int brojCore2Duo = 0;
            double ram = 0;

```

```

        for(int i=0;i<racunari.size();i++){
            if (racunari.get(i).getTipProcesora().equals("Core 2Duo"))
                brojCore2Duo++;
            ram = ram + racunari.get(i).getRamMemorija();
        }

        ram = ram/racunari.size();

        out.println("Racunarski centar - statistika");
        out.println("Broj racunara sa Core 2 Duo procesorom: "+brojCore2Duo);
        out.println("prosecna velicina RAM memorije: "+ram);
        out.println("Ukupan broj racunara: "+racunari.size());

        out.close();
    } catch (Exception e) {
        System.out.println(e.getMessage());
    }
}

```

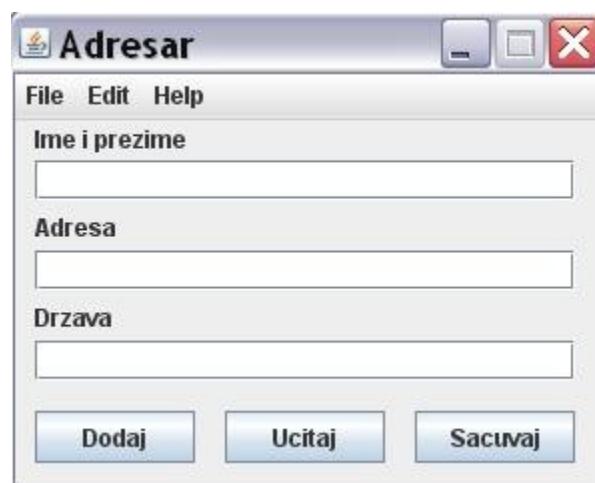
Zadatak 4

Napraviti javnu klasu **StavkaAdresara** koja može da bude serijalizovana i ima:

- Privatni atribut imePrezime. Početna vrednost za ovaj atribut je “nepoznato”.
- Privatni atribut adresa. Početna vrednost za ovaj atribut je “nepoznato”.
- Privatni atribut država. Početna vrednost za ovaj atribut je “nepoznato”.
- Odgovarajuće javne get i set metode za ove atribute. Nedozvoljene vrednosti za sva tri atributa su null String-ovi. U slučaju unosa nedozvoljenih vrednosti potrebno je baciti izuzetak sa porukom “UNELI STE PRAZAN STRING”.
- Redefinisana equals metodu klase Object koja kao ulazni argument dobija objekat klase Object. Metoda prvo proverava da li je uneti objekat klase StavkaAdresara, pa ako nije baca izuzetak. Metoda vraća true ako su vrednosti atributa imePrezime, adresa i država jednake vrednostima istih atributa unetog objekta a false ako nisu.

Napraviti **vizuelnu klasu AdresarGUI** koja izgleda kao na slici (na drugoj strani). Naslov prozora bi trebalo da bude “Adresar”. Podesiti grafički interfejs tako da se, u toku rada aplikacije, ne mogu menjati dimenzije forme.

- Klasa AdresarGUI bi trebalo da sadrži privatni atribut **stavke** koji predstavlja listu objekata klase StavkaAdresara. Odmah inicijalizovati listu.
- Kada se pritisne dugme “Dodaj”, preuzimaju se podaci o imenu i prezimenu, adresi i državi iz polja za unos, kreira se objekat klase StavkaAdresara i unosi se u listu. Unošenje se vrši samo ako u listi već ne postoji ista stavka.
- Kada se pritisne dugme “Učitaj”, učitavaju se (**deserijalizuju**) svi podaci (stavke adresara) iz fajla “adresar.out” i puni se lista. Pre učitavanja je potrebno izbrisati listu.
- Kada se pritisne dugme “Sačuvaj”, **cela lista sa svim stavkama** se upisuje (**serijalizuje**) u fajl “adresar_lista.out”.



Rešenje

- Napraviti klasu StavkaAdresara

```
import java.io.Serializable;

public class StavkaAdresara implements Serializable{

    private String imePrezime = "nepoznato";
    private String adresa = "nepoznato";
    private String drzava = "nepoznato";

    public String getImePrezime() {
        return imePrezime;
    }
    public void setImePrezime(String imePrezime) {
        if (imePrezime == null)
            throw new RuntimeException("Uneli ste prazan String");

        this.imePrezime = imePrezime;
    }
    public String getAdresa() {
        return adresa;
    }
    public void setAdresa(String adresa) {
        if (adresa == null)
            throw new RuntimeException("Uneli ste prazan String");

        this.adresa = adresa;
    }
    public String getDrzava() {
        return drzava;
    }
    public void setDrzava(String drzava) {
        if (drzava == null)
            throw new RuntimeException("Uneli ste prazan String");

        this.drzava = drzava;
    }
    public boolean equals(Object o){
        if (!(o instanceof StavkaAdresara))
            throw new RuntimeException("Greska");

        StavkaAdresara s = (StavkaAdresara)(o);

        if (s.getImePrezime().equals(imePrezime)&&
            s.getAdresa().equals(adresa) &&
            s.getDrzava().equals(drzava))
            return true;
        else
            return false;
    }
}
```

- Kreirati nov grafički interfejs (vizuelnu klasu) sa imenom AdresarGUI.
- Promeniti naslov na formi tako da bude “Adresar”.
- Potrebno je podesiti formu tako da ima null layout.
- Dodati dugme na formu sa imenom DodajDugme i naslovom “Dodaj”.
- Dodati dugme na formu sa imenom UcitajDugme i naslovom “Ucitaj”.
- Dodati dugme na formu sa imenom SacuvajDugme i naslovom “Sacuvaj”.
- Dodati labelu na formu sa bilo kojim imenom i naslovom “Ime i prezime”.
- Dodati labelu na formu sa bilo kojim imenom i naslovom “Adresa”.
- Dodati labelu na formu sa bilo kojim imenom i naslovom “Drzava”.
- Ispod labele na kojoj piše “Ime i prezime” dodati novo polje za unos sa imenom PoljeZaUnosImena.
- Ispod labele na kojoj piše “Adresa” dodati novo polje za unos sa imenom PoljeZaUnosAdrese.
- Ispod labele na kojoj piše “Drzava” dodati novo polje za unos sa imenom PoljeZaUnosDrzave.
- Podesiti atribut forme “resizable” na false uz pomoć Properties tab-a.

- Dodati direktno u kod klase TelefonskiImenikGUI privatni atribut kontakti:

```
private LinkedList<StavkaAdresara> stavke = new LinkedList<StavkaAdresara>();
```

- Dodati event handler za dugme DodajDugme i u njega dodati sledeći kod:

```
StavkaAdresara s = new StavkaAdresara();

s.setImePrezime(PoljeZaUnosImena.getText());
s.setAdresa(PoljeZaUnosAdrese.getText());
s.setDrzava(PoljeZaUnosDrzave.getText());

if(!stavke.contains(s))
    stavke.add(s);
```

- Dodati event handler za dugme UcitajDugme i u njega dodati sledeći kod:

```
try {
    ObjectInputStream in =
        new ObjectInputStream(
            new BufferedInputStream(
                new FileInputStream("adresar.out")));

    stavke.clear();

    try {
        while(true)
            stavke.add((StavkaAdresara)(in.readObject()));
    } catch (Exception e2) {}

    in.close();
} catch (Exception e2) {
    System.out.println(e2.getMessage());
}
```

- Dodati event handler za dugme SacuvajDugme i u njega dodati sledeći kod:

```
try {
    ObjectOutputStream out=
        new ObjectOutputStream(
            new BufferedOutputStream(
                new FileOutputStream("adresar_lista.out")));

    out.writeObject(stavke);

    out.close();
} catch (Exception e2) {
    System.out.println(e2.getMessage());
}
```