

## Nizovi objekata

U trećem poglavlju su detaljno objašnjeni nizovi - šta predstavljaju i čemu služe, kako se formiraju i koriste itd. Tu je, međutim, prećutno uvedeno i jedno ograničenje - elementi niza su uvek bili prostog tipa podataka (char, int, boolean, double). U Javi je moguće napraviti niz čiji elementi nisu prostog tipa, već je **svaki element po jedan objekat**. Rad sa nizovima objekata je skoro potpuno isti kao i rad sa običnim nizovima. Jedine razlike se svode na to da se svaki pojedinačni element tretira malo drugačije jer je u pitanju objekat.

Deklaracija niza objekata potpuno ista kao i za običan niz:

```
tip_podatka[] nazivPromenljive;
```

Sa obzirom na to da su elementi niza objekti neke klase, tip podatka je upravo naziv te klase:

```
NazivKlase[] nazivPromenljive;
```

Inicijalizacija niza je i u ovom slučaju potrebna, i vrši se na isti način kao i za običan niz:

```
nazivPromenljive = new NazivKlase[ceo_broj];
```

Prva razlika u odnosu na obične nizove se može videti pri inicijalizaciji. Kod inicijalizacije niza čiji su elementi prostog tipa, sama inicijalizacija niza rezerviše sav potreban memorijski prostor za svaki element niza i sa nizom se odmah može raditi. Kada je u pitanju niz objekata, inicijalizacija niza samo rezerviše memorijski prostor za pokazivače na objekte, ali **ne inicijalizuje same objekte**. Drugim rečima, **posle inicijalizacije niza objekata, svaki element ima vrednost “null” i sa njim se ne može raditi dok se ne inicijalizuje**. U nekim situacijama je potrebno izvršiti pojedinačnu inicijalizaciju elemenata odmah na početku, a nekad se samo unose već inicijalizovani objekti na odgovarajuća mesta u nizu (u tom slučaju se smatra da je mesto u nizu “prazno” ako ima “null” vrednost).

Pristup elementima niza se vrši preko indeksa, a maksimalni kapacitet niza se dobija pozivanjem komande “length”:

```
nazivPromenljive[indeks]
```

```
nazivPromenljive.length
```

Druga razlika u odnosu na obične nizove se sastoji u tome što se nad elementima niza objekata direktno mogu pozivati metode. Svaki element je po jedan objekat, pa se pozivi njegovih metoda mogu izvršiti ovako:

```
nazivPromenljive[indeks].nazivMetode(...ulazni parametri...);
```

### Primer 1

Napraviti javnu klasu **TemperaturaMesta** koja ima:

- Privatni atribut naziv koji predstavlja naziv mesta. Početna vrednost za ovaj atribut je “nepoznat”.
- Privatni atribut temperatura (ceo broj) koji predstavlja dnevnu temperaturu za to mesto.
- Odgovarajuće javne get i set metode za ova dva atributa.

Napraviti javnu klasu **DnevnaPrognoza** koja ima:

- Privatni atribut *temperature* koji predstavlja niz objekata klase *TemperaturaMesta*.
- Javni konstruktor koji kao ulazni argument prima broj koji predstavlja maksimalan broj mesta na koje se prognoza odnosi. Ako je uneti broj veći od nule, potrebno je inicijalizovati atribut *temperature* na taj kapacitet. Ako je uneti broj nula ili manji od nule, kapacitet se postavlja na 10 i ispisuje se poruka o grešci na ekranu. U svakom slučaju je potrebno inicijalizovati svaki element niza.
- Javnu metodu *imaSlobodnihMesta* koja vraća *TRUE* ako u nizu ima slobodnih mesta za unos temeprature, a *FALSE* ako nema. Mesto u nizu je slobodno ako umesto naziva mesta na koje se odnosi prognoza stoji String "nepoznat".
- Javnu metodu *unesi* koja kao ulazne parametre prima naziv mesta i temperaturu i unosi te podatke na prvo slobodno mesto u nizu. Ako u nizu nema slobodnih mesta, metoda ispisuje poruku o tome na ekranu.
- Javnu metodu *izbaci* koja kao ulazi parametar prima naziv mesta i izbacuje podatke o tom mestu i njegovoj temperaturi iz niza. Izbacivanje se vrši tako što se kao naziv mesta postavlja reč "nepoznat" a temperatura dobija vrednost 0. Ako u nizu nema mesta sa unetim nazivom, ništa se ne dešava.
- Javnu metodu *ispisi* koja na ekranu ispisuje celokupnu dnevnu prognozu.

Napraviti klasu **TestDnevnaPrognoza** koja kreira jedan objekat klase *DnevnaPrognoza* kapaciteta 3 mesta i u njega unosi podatke o temperaturama za tri mesta: "Beograd" (17C), "Novi Sad" (13C), "Nis" (16C). Ispisati podatke o svim gradovima i njihovim temperaturama.

```
public class TemperaturaMesta {

    private String naziv = "nepoznat";
    private int temperatura;

    public String getNaziv() {
        return naziv;
    }
    public void setNaziv(String naziv) {
        this.naziv = naziv;
    }
    public int getTemperatura() {
        return temperatura;
    }
    public void setTemperatura(int temperatura) {
        this.temperatura = temperatura;
    }
}

public class DnevnaPrognoza {

    private TemperaturaMesta[] temperature;

    public DnevnaPrognoza(int brojMesta) {
        if (brojMesta > 0)
            temperature = new TemperaturaMesta[brojMesta];
        else {
            temperature = new TemperaturaMesta[10];
        }
    }
}
```

```

        System.out.println("Greska");
    }

    // Pojedinačna inicijalizacija svakog
    // elementa niza.
    for (int i = 0; i < temperature.length; i++)
        temperature[i] = new TemperaturaMesta();
}

public boolean imaSlobodnihMesta() {
    for (int i = 0; i < temperature.length; i++)
        if (temperature[i].getNaziv().equals("nepoznat"))
            return true;

    return false;
}

public void unesi(String naziv, int temperatura) {
    if (!imaSlobodnihMesta())
        System.out.println("Nema slobodnih mesta");
    else
        for (int i = 0; i < temperature.length; i++)
            if (temperature[i].getNaziv().equals("nepoznat")) {
                temperature[i].setNaziv(naziv);
                temperature[i].setTemperatura(temperatura);
                break;
            }
}

public void izbaci(String naziv) {
    for (int i = 0; i < temperature.length; i++)
        if (temperature[i].getNaziv().equals(naziv)) {
            temperature[i].setNaziv("nepoznat");
            temperature[i].setTemperatura(0);
            break;
        }
}

public void ispisi(){
    for (int i = 0; i < temperature.length; i++)
        System.out.println(
            "Mesto: "+temperature[i].getNaziv()+
            " Temperatura: "+
            temperature[i].getTemperatura());
}

}

public class TestDnevnaPrognoza {

    public static void main(String[] args) {

        DnevnaPrognoza d = new DnevnaPrognoza(3);
    }
}

```

```

        d.unesi("Beograd", 17);
        d.unesi("Novi Sad", 13);
        d.unesi("Nis", 16);

        d.ispisi();
    }
}

```

*Konstruktor klase DnevnaProгноza inicijalizuje atribut "temperature" koji predstavlja niz objekata klase TemperaturaMesta. Pored inicijalizacije niza, ovaj konstruktor inicijalizuje i svaki element niza (objekat) pojedinačno. To se vrši pozivanjem konstruktora klase TemperaturaMesta za svaki element. Tek posle ove inicijalizacije se elementi niza mogu koristiti.*

*Metoda "imaSlobodnihMesta" proverava da li je neko mesto u nizu slobodno na taj način što poredi da li atribut "naziv" objekta koji se nalazi na tom mestu u nizu ima vrednost "nepoznat". Tu se može videti da se vrednost njegovog atributa "naziv" dobija pozivanjem metode "getNaziv" ("temperature[i].getNaziv()"). Sa obzirom na to da ova metoda vraća naziv (String) potrebno ga je uporediti sa vrednošću "nepoznat" što se vrši pozivanjem "equals" metode. Ceo logički izraz u okviru IF komande zbog toga izgleda relativno složeno pa glasi: "temperature[i].getNaziv().equals("nepoznat)".*

## Zadaci

### Zadatak 1

Napraviti javnu klasu **ParkingMesto** koja ima:

- Privatni atribut slobodno koji predstavlja indikator zauzetosti parking mesta. Ovaj indikator ima vrednost TRUE ako je mesto slobodno a FALSE ako nije.
- Privatni atribut registarskiBroj koji predstavlja registarski broj vozila koje se nalazi na tom parking mestu (String).
- Odgovarajuće javne get i set metode za ova dva atributa.

Napraviti javnu klasu **Parking** koja ima:

- Privatni atribut mesta koji predstavlja niz objekata klase ParkingMesto.
- Javni konstruktor koji kao ulazni argument prima broj koji predstavlja kapacitet parkinga tj. ukupan broj parking mesta. Ako je uneti broj veći od nule, potrebno je inicijalizovati atribut mesta na taj kapacitet. Ako je uneti broj nula ili manji od nule, kapacitet parking mesta se postavlja na 40 i ispisuje se poruka o grešci na ekranu. U svakom slučaju je potrebno inicijalizovati svako parking mesto i postaviti ga da bude slobodno.
- Javnu metodu koja ispisuje na ekranu registarski broj kola koja se nalaze na prvom parking mestu. Ako je parking mesto slobodno, ispisuje se poruka o tome.
- Javnu metodu koja ispisuje na ekranu registarski broj kola koja se nalaze na poslednjem parking mestu. Ako je parking mesto slobodno, ispisuje se poruka o tome.
- Javnu metodu koja proverava da li na parkingu ima slobodnih mesta i vraća TRUE ako ima, a FALSE ako nema.
- Javnu metodu koja vraća broj slobodnih parking mesta.
- Javnu metodu koja kao ulazni argument prima registarski broj vozila i proverava da li se to vozilo nalazi na parkingu. Ako se nalazi, metoda vraća TRUE a suprotnom FALSE.
- Javnu metodu za "uvođenje" vozila na parking. Ova metoda kao ulazni argument dobija registarski broj vozila. Prvo je potrebno proveriti da li na parkingu ima slobodnih mesta. Ako ima, potrebno je uvesti vozilo na prvo slobodno mesto. Ako slobodnih mesta nema, ispisati poruku o tome na ekranu.
- Javnu metodu za "izvođenje" vozila sa parkinga. Ova metoda kao ulazni argument dobija registarski broj vozila. Prvo je potrebno proveriti da li se vozilo sa tim registarskim brojem nalazi na parkingu. Ako se nalazi, potrebno ga je izvesti, tako da parking mesto ponovo postane slobodno. Izvođenje vozila podrazumeva da se mesto na kome je bilo označi kao slobodno i da se podatak o njegovom registarskom broju ukloni.
- Javnu metodu koja na ekranu ispisuje registarske brojeve svih vozila koja se nalaze na parkingu i broj parking

- mesta na kome se nalaze. Naravno, ispisivanje se vrši samo za ona parking mesta koja su zauzeta.
- Javnu metodu koja na ekranu ispisuje registarske brojeve svih vozila koja se nalaze na parkingu a imaju beogradske tablice.

Napraviti klasu **TestParking** koja kreira jedan objekat klase Parking kapaciteta 50 mesta i u njega unosi kola sa tablicama "BG 123-456" i "NS 234-56". Ispisati registarske tablice svih vozila koja su na parkingu, a onda i registarske tablice vozila iz Beograda.

#### Rešenje:

```
public class ParkingMesto {

    private boolean slobodno;

    private String registarskiBroj;

    public String getRegistarskiBroj() {
        return registarskiBroj;
    }

    public void setRegistarskiBroj(String registarskiBroj) {
        this.registarskiBroj = registarskiBroj;
    }

    public boolean isSlobodno() {
        return slobodno;
    }

    public void setSlobodno(boolean slobodno) {
        this.slobodno = slobodno;
    }

}

public class Parking {

    private ParkingMesto[] mesta;

    public Parking (int kapacitet){
        if (kapacitet > 0){
            mesta = new ParkingMesto[kapacitet];
        }
        else{
            System.out.println("Greska");
            mesta = new ParkingMesto[40];
        }

        for (int i=0; i<mesta.length;i++){
            mesta[i] = new ParkingMesto();
            mesta[i].setSlobodno(true);
        }
    }

    public void ispisiPrvo(){
        if (mesta[0].isSlobodno())
            System.out.println("Prvo mesto je slobodno");
        else
            System.out.println("Registarski broj: "+
                                mesta[0].getRegistarskiBroj());
    }

    public void ispisiPoslednje(){
        if (mesta[mesta.length-1].isSlobodno())
            System.out.println("Poslednje mesto je slobodno");
    }
}
```

```

        else
            System.out.println("Registarski broj: "+
                               mesta[mesta.length-1].getRegistarskiBroj());
    }

    public boolean imaSlobodnih(){
        for (int i=0; i<mesta.length;i++)
            if (mesta[i].isSlobodno()) return true;

        return false;
    }

    public int brojSlobodnih(){
        int brojac = 0;
        for (int i=0; i<mesta.length;i++)
            if (mesta[i].isSlobodno()) brojac++;

        return brojac;
    }

    public boolean daLiJeNaParking (String regBr){
        for (int i=0; i<mesta.length;i++)
            if (!mesta[i].isSlobodno()){
                String regBr1 = mesta[i].getRegistarskiBroj();
                if (regBr1.equals(regBr)) return true;
            }

        return false;
    }

    public void uvediNaParking (String regBr){
        if (!imaSlobodnih())
            System.out.println("Nema slobodnih mesta");
        else{
            for (int i=0; i<mesta.length;i++)
                if (mesta[i].isSlobodno()){
                    mesta[i].setSlobodno(false);
                    mesta[i].setRegistarskiBroj(regBr);
                    break;
                }
        }
    }

    public void izvediSaParkinga (String regBr){
        if (!daLiJeNaParking(regBr))
            System.out.println("To vozilo se ne nalazi na parkingu");
        else{
            for (int i=0; i<mesta.length;i++)
                if (!mesta[i].isSlobodno() &&
                    mesta[i].getRegistarskiBroj().equals(regBr)){
                    mesta[i].setSlobodno(true);
                    mesta[i].setRegistarskiBroj(null);
                    break;
                }
        }
    }

    public void ispisi(){
        for (int i=0; i<mesta.length;i++)
            if (!mesta[i].isSlobodno())
                System.out.println("Mesto br."+i+" Reg. br. "+
                                    mesta[i].getRegistarskiBroj());
    }

```

```

        public void ispisiBG() {
            for (int i=0; i<mesta.length;i++)
                if (!mesta[i].isSlobodno()) {
                    String grad = mesta[i].getRegistarskiBroj().substring(0, 2);
                    if (grad.equals("BG"))
                        System.out.println("Mesto br."+i+" Reg. br. "+
                                           mesta[i].getRegistarskiBroj());
                }
        }
    }

}

public class TestParking {

    public static void main(String[] args) {

        Parking p = new Parking (50);

        p.vediNaParking("BG 123-456");
        p.vediNaParking("NS 234-56");

        p.ispisi();

        p.ispisiBG();

    }

}

```

## Zadatak 2

Napisati javnu klasu **MestoUAvionu** koja ima:

- Privatni atribut slobodno koji predstavlja indikator zauzetosti sedišta (mesta). Ovaj indikator ima vrednost TRUE ako je mesto slobodno a FALSE ako nije. Početna vrednost za ovaj atribut je TRUE.
- Privatni atribut imePrezime koji predstavlja ime i prezime putnika koji sedi na tom mestu dato u formatu "IME PREZIME". Početna vrednost za ovaj atribut je null.
- Privatni atribut starost koji predstavlja broj godina putnika koji sedi na tom mestu. Početna vrednost za ovaj atribut je 0.
- Odgovarajuće javne get i set metode za ova tri atributa.
- Redefinisani metodu toString klase Object koja vraća String sa svim podacima o mestu u avionu. Ako je mesto slobodno, vraća se String koji sadrži reč "Slobodno", a ako nije vraća se String sa imenom i prezimenom i godinama starosti putnika koji se nalazi na tom mestu.

Napisati javnu klasu **Avion** koja ima:

- Privatni atribut mesta koji predstavlja niz objekata klase MestoUAvionu.
- Javni konstruktor koji inicijalizuje atribut mesta i kreira svako pojedinačno mesto u avionu. Avion ima tačno 120 mesta.
- Javnu metodu uvediPutnika koja kao ulazni argument dobija redni broj mesta, String sa imenom i prezimenom putnika i godine starosti putnika. Ako je mesto sa tim brojem već zauzeto ili je uneti broj mesta van opsega (0-119), potrebno je ispisati poruku o grešci, a u suprotnom "uvesti" putnika na dato mesto i postaviti da mesto nije slobodno.
- Javnu metodu izvediPutnika koja kao ulazni argument dobija redni broj mesta. Ako je mesto sa tim brojem već slobodno ili ako je broj mesta van opsega (0-119), potrebno je ispisati poruku o grešci, a u suprotnom "izvesti" putnika sa datog mesta i postaviti mesto da bude slobodno.
- Javnu metodu daLiJeUAvionu koja kao ulazni argument dobija String sa imenom i prezimenom putnika i njegove godine starosti. Metoda vraća TRUE ako se putnik nalazi u avionu, a u suprotnom FALSE.
- Javnu metodu koja izračunava i vraća procentualnu zauzetost sedišta u avionu. Ako je avion popunjen, metoda vraća 100.0%, a ako je prazan 0.0%.
- Javnu metodu koja proverava da li u avionu ima slobodnih sedišta. Ako ima, metoda vraća TRUE, a u suprotnom FALSE.
- Javnu metodu koja izračunava i vraća prosečnu starost putnika kao ceo broj.
- Javnu metodu koja vraća broj godina najstarijeg putnika.
- Javnu metodu koja vraća broj godina najmlađeg putnika.

- Javnu metodu koja na ekranu ispisuje redni broj i podatke o svakom mestu u avionu.
- Javnu metodu koja na ekranu ispisuje podatke o svim putnicima u avionu koji se prezivaju "Jovanovic".

Napraviti klasu **TestAvion** koja kreira jedan objekat klase Avion. Potrebno je u avion uvesti putnike: "Jovan Jovanovic" (53 godine), "Milos Milosevic" (23 godine) i "Ana Jovanovic" (22 godine) na 21, 31 i 41 mesto. Ispisati sve podatke o putnicima, ispisati samo podatke o putnicima koji se prezivaju Jovanovic i ispisati prosečnu starost putnika.

#### Rešenje:

```
public class MestoUAvionu {

    private boolean slobodno = true;;
    private String imePrezime = null;
    private int starost = 0;

    public String getImePrezime() {
        return imePrezime;
    }
    public void setImePrezime(String imePrezime) {
        this.imePrezime = imePrezime;
    }
    public boolean isSlobodno() {
        return slobodno;
    }
    public void setSlobodno(boolean slobodno) {
        this.slobodno = slobodno;
    }
    public int getStarost() {
        return starost;
    }
    public void setStarost(int starost) {
        this.starost = starost;
    }

    public String toString(){
        if (slobodno)
            return "Slobodno";
        else
            return "Ime putnika "+imePrezime+" Starost: "+starost;
    }

}

public class Avion {

    private MestoUAvionu[] mesta;

    public Avion(){
        mesta = new MestoUAvionu[120];
        for (int i=0; i<120;i++)
            mesta[i] = new MestoUAvionu();
    }

    public void uvediPutnika(int brojMesta, String imePrezime, int starost){
        if (brojMesta<0 || brojMesta>119 || !mesta[brojMesta].isSlobodno())
            System.out.println("Greska");
        else{
            mesta[brojMesta].setSlobodno(false);
            mesta[brojMesta].setImePrezime(imePrezime);
            mesta[brojMesta].setStarost(starost);
        }
    }

    public void izvediPutnika(int brojMesta){
```



```

        if (brojMesta<0 || brojMesta>119 || mesta[brojMesta].isSlobodno())
            System.out.println("Greska");
        else{
            mesta[brojMesta].setSlobodno(true);
            mesta[brojMesta].setImePrezime(null);
            mesta[brojMesta].setStarost(0);
        }
    }

    public boolean daLiJeUAvionu(String imePrezime, int starost){
        for (int i=0; i<120;i++)
            if (!mesta[i].isSlobodno() &&
                mesta[i].getImePrezime().equals(imePrezime) &&
                mesta[i].getStarost() == starost) return true;

        return false;
    }

    public double procentualnaZauzetost(){
        int brojZauzetih = 0;
        for (int i=0; i<120;i++)
            if (!mesta[i].isSlobodno()) brojZauzetih++;

        return (brojZauzetih*100.0)/120.0;
    }

    public boolean daLiImaSlobodnih(){
        for (int i=0; i<120;i++)
            if (mesta[i].isSlobodno()) return true;

        return false;
    }

    public int prosečnaStarost(){
        int suma = 0;
        int brojZauzetih = 0;
        for (int i=0; i<120;i++)
            if (!mesta[i].isSlobodno()){
                suma = suma + mesta[i].getStarost();
                brojZauzetih++;
            }
        return suma/brojZauzetih;
    }

    public int najstarijiPutnik(){
        int maxGodine = 0;
        for (int i=0; i<120;i++)
            if (!mesta[i].isSlobodno() &&
                mesta[i].getStarost() > maxGodine)
                maxGodine = mesta[i].getStarost();
        return maxGodine;
    }

    public int najmladjiPutnik(){
        int minGodine = 200;
        for (int i=0; i<120;i++)
            if (!mesta[i].isSlobodno() &&
                mesta[i].getStarost() < minGodine)
                minGodine = mesta[i].getStarost();
        return minGodine;
    }

    public void ispisi(){
        for (int i=0; i<120;i++)

```

```

        System.out.println("Sediste "+i+" "+mesta[i]);
    }

    public void ispisiJovanovice() {
        for (int i=0; i<120;i++)
            if (!mesta[i].isSlobodno()) {
                String imePrezime = mesta[i].getImePrezime();
                String prezime =
                    imePrezime.substring(imePrezime.indexOf(' ')+1);
                if (prezime.equals("Jovanovic"))
                    System.out.println(mesta[i]);
            }
    }
}

public class TestAvion {

    public static void main(String[] args) {

        Avion a = new Avion();

        a.vediPutnika(20, "Jovan Jovanovic", 53);
        a.vediPutnika(30, "Milos Milosevic", 23);
        a.vediPutnika(40, "Ana Jovanovic", 22);

        a.ispisi();
        a.ispisiJovanovice();

        System.out.println("Prosečna starost putnika je "+
            a.prosečnaStarost()+" godina");
    }
}

```

### Zadatak 3

Napisati javnu klasu **Kontakt** koja ima:

- Privatni atribut imePrezime koji predstavlja ime i prezime osobe. Početna vrednost za ovaj atribut je null.
- Privatni atribut adresa. Početna vrednost za ovaj atribut je null.
- Privatni atribut telefon. Početna vrednost za ovaj atribut je nula.
- Odgovarajuće javne get i set metode za ova tri atributa. Nedoovoljene vrednosti za imePrezime i adresu su null String-ovi, a uneti telefon mora biti broj veći od nule. U slučaju prekoračenja bilo kojeg od ovih ograničenja, potrebno je ispisati poruku na ekranu o tome koje ograničenje je prekoračeno i zašto.
- Redefinisati metodu toString klase Object koja vraća String sa svim podacima o kontaktu.

Napisati javnu klasu **Adresar** koja ima:

- Privatni atribut kontakti koji predstavlja niz objekata klase Kontakt.
- Javni konstruktor koji inicijalizuje atribut kontakti na maksimalno 250 kontakata. Smatra se da je mesto u adresaru slobodno ako element na tom mestu ima vrednost NULL.
- Javnu metodu imaSlobodnih koja vraća TRUE ako u adresaru ima slobodnih mesta, a FALSE ako nema. Mesto je slobodno ako element na tom mestu ima vrednost NULL.
- Javnu metodu unesiUAdresar koja kao ulazni parametar dobija objekat klase Kontakt i unosi ga na prvo slobodno mesto u adresaru. Ako u adresaru nema slobodnih mesta, potrebno je ispisati poruku o tome na ekranu.
- Javnu metodu unesiUAdresar koja kao ulazne parametre dobija ime i prezime (jedan String), adresu i telefon kontakta i unosi te podatke na prvo slobodno mesto u adresaru. Ako u adresaru nema slobodnih mesta, potrebno je ispisati poruku o tome na ekranu.
- Javnu metodu izbaciIzAdresara koja kao ulazni parametar dobija ime i prezime osobe (jedan String) i izbacuje tu osobu iz adresara. Ako te osobe nema u adresaru, nije potrebno ništa uraditi.
- Javnu metodu ispisi koja ispisuje čitav sadržaj adresara, ali tako da se ne ispišu prazna mesta u adresaru.

Napraviti klasu **TestAdresar** koja kreira jedan objekat klase Adresar. Potrebno je uneti podatke o četiri osobe: "Pera

Peric", "Oblakovska 1", 123456, "Mika Mikic", "Strahinjica Bana 1", 654321, "Laza Lazic", "Kneza Milosa 1", 567890 i "Zika Zikic", "Karadjordjeva 1", 987654. Uneti prva dva kontakta pozivanjem prve metode za unos, a druga dva kontakta pozivanjem druge metode za unos. Ispisati sve podatke iz adresara na ekranu.

#### Rešenje:

```
public class Kontakt {

    private String imePrezime = null;
    private String adresa = null;
    private int telefon = 0;

    public String getImePrezime() {
        return imePrezime;
    }
    public void setImePrezime(String imePrezime) {
        if (imePrezime == null)
            System.out.println("Ime i prezime ne mogu biti null");
        else
            this.imePrezime = imePrezime;
    }
    public String getAdresa() {
        return adresa;
    }
    public void setAdresa(String adresa) {
        if (adresa == null)
            System.out.println("Adresa ne moze biti null");
        else
            this.adresa = adresa;
    }
    public int getTelefon() {
        return telefon;
    }
    public void setTelefon(int telefon) {
        if (telefon<=0)
            System.out.println("Uneti telefon mora biti veci od nule");
        else
            this.telefon = telefon;
    }

    public String toString(){
        return "Ime i prezime: "+imePrezime+
            " Adresa: "+adresa+
            " Telefon: "+telefon;
    }

}

public class Adresar {

    private Kontakt[] kontakti;

    public Adresar() {
        kontakti = new Kontakt[250];
    }

    public boolean imaSlobodnih() {
        for (int i = 0; i < kontakti.length; i++)
            if (kontakti[i] == null)
                return true;

        return false;
    }

}
```

```

public void unesiUAdresar(Kontakt k) {
    if (!imaSlobodnih())
        System.out.println("Nema mesta u adresaru");
    else
        for (int i = 0; i < kontakti.length; i++)
            if (kontakti[i] == null) {
                kontakti[i] = k;
                break;
            }
}

public void unesiUAdresar(String imePrezime, String adresa, int telefon) {
    if (!imaSlobodnih())
        System.out.println("Nema mesta u adresaru");
    else
        for (int i = 0; i < kontakti.length; i++)
            if (kontakti[i] == null) {
                kontakti[i] = new Kontakt();
                kontakti[i].setImePrezime(imePrezime);
                kontakti[i].setAdresa(adresa);
                kontakti[i].setTelefon(telefon);
                break;
            }
}

public void izbaciIzAdresara(String imePrezime) {
    //Pre proveriti jedankosti imena i prezimena,
    //potrebno je proveriti da li je taj element
    //niza uopste inicijalizovan (!=null). Tek
    //onda se bezbedno mogu pozivati metode npr.
    //"getIme". Element adresara se "brise" tako
    //sto mu se dodeli "null" vrednost.
    for (int i = 0; i < kontakti.length; i++)
        if (kontakti[i] != null
            && kontakti[i].getImePrezime().equals(imePrezime)) {
            kontakti[i] = null;
            break;
        }
}

public void ispisi(){
    for (int i = 0; i < kontakti.length; i++)
        if (kontakti[i] != null)
            System.out.println(kontakti[i]);
}

}

public class TestAdresar {

    public static void main(String[] args) {

        Adresar a = new Adresar();

        Kontakt k1 = new Kontakt();
        k1.setImePrezime("Pera Peric");
        k1.setAdresa("Oblakovska 1");
        k1.setTelefon(123456);

        a.unesiUAdresar(k1);

        Kontakt k2 = new Kontakt();
        k2.setImePrezime("Mika Mikic");
        k2.setAdresa("Strahinjica Bana 1");
    }
}

```

```
k2.setTelefon(654321);  
  
a.unesiUAdresar(k2);  
  
a.unesiUAdresar("Laza Lazic", "Kneza Milosa 1", 567890);  
a.unesiUAdresar("Zika Zikic", "Karadjordjeva 1", 987654);  
a.ispisi();  
  
}  
  
}
```