VEŽBA BR. 4

Nizovi, višedimenzionalni nizovi (matrice)

Cilj vežbe: Upoznavanje za nizovima i matricama

Nizovi su promenljive koje mogu da uskladište više vrednosti odjednom. Deklaracija niza se vrši na sledeći način:

```
tip_podataka[] nazivPromenljive;
```

Deklaracija je skoro identična kao za običnu promenljivu, jedini dodatak predstavljaju uglaste zagrade koje se navode posle tipa podataka. Pošto nizovi mogu da uskladište više vrednosti odjednom, ove vrednosti se zovu **elementi niza**. Svi elementi jednog niza su uvek istog tipa podataka. Da bi neki niz mogao da se koristi on prvo mora da se **inicijalizuje**.

Inicijalizacija se vrši na sledeći način:

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

Promenljiva koja predstavlja niz ustvari nije niz, već je samo **pokazivač na niz.** Inicijalizacijom se alocira memorija za niz i tek tada se niz može koristiti. Broj koji se nalazi u zagradi mora da bude neki pozitivan ceo broj. On predstavlja **kapacitet niza** tj. koliko će elemenata niz moći maksimalno da sadrži. Na primer, niz kapaciteta 100 može da primi 100 celih brojeva. Jednom kada se kapacitet odredi, on ostaje **fiksiran** i ne može se **smanjiti niti povećati.** Niz se može ponovo inicijalizovati (ako je potrebno da se poveća ili smanji kapacitet), ali se tada brišu elementi niza.

Promenljiva koja predstavlja niz ima samo jedan naziv, a odnosi se na više elemenata, pa se postavlja pitanje kako se pristupa pojedinačnim elementima niza.

Svaki niz ima svoj **indeks.** Indeks elementa je ceo broj koji predstavlja redni broj elementa u nizu. **Indeksi elemenata počinju od nule**, a ne od jedan. Indeks pevog elementa niza kapaciteta 10, je 0 a indeks poslednjeg elementa niza je 9.

Pozivanje pojedinačnih elemenata niza uz pomoć indeksa se vrši na sledeći način:

```
nazivPromenljive[indeks]
```

Ako je potrebno dobiti vrednost kapaciteta niza (želimo da proverimo da li je niz dovoljno dugačak) to se radi na sledeći način:

```
nazivPromenljive.length
```

For petlja

U praksi se nizovi često koriste u kombinaciji sa For petljom. Tada se brojač For petlje koristi da simulira indeks elemenata niza. Postavlja se da brojač petlje ima početnu vrednost nula (jer indeksi niza počinju od nule), a petlja se izvršava sve dok brojač ne dostigne maksimalnu vrednost indeksa.

Primer 1. Napraviti klasu ***MesecniProfiti*** koja ima:

- Atribut profiti koji predstavlja niz od 12 realnih brojeva. Svaki element niza predstavlja profit za određeni mesec (januar, februar, …, decembar).
- Metodu koja kao parametre prima realan broj koji predstavlja profit i ceo broj koji predstavlja redni broj meseca na koji se taj profit odnosi (1 – januar, 2-februar, …, 12 - decembar). Metoda unosi odgovarajuću vrednost profita za taj mesec u niz.
- Metodu koja na ekranu ispisuje profit za svaki mesec.

Napraviti klasu ***TestMesecniProfiti*** koja kreira jedan objekat klase ***MesecniProfiti***, unosi profit za januar koji iznosi 223.42 i ispisuje sve mesečne profite na ekranu.

Kod primera:

```
package mesecniProfiti;

class MesecniProfiti {
	double[] profiti = new double[12];

	void unesiProfit(double profit, int mesec) {
		profiti[mesec-1] = profit;
		//Ovde stoji mesec-1 zbog toga sto
		//indeksi niza krecu od 0 pa je indeks
		//za mesec januar 0 iako je to prvi mesec, a
		//indeks za mesec decembar bi bio 11.
	}

	void ispisi() {
		for(int i=0; i<profiti.length; i++) {
			System.out.println(profiti[i]);
		}
	}
}
public class testMesecniProfiti {

	public static void main(String[] args) {
		MesecniProfiti mp = new MesecniProfiti();

		mp.unesiProfit(223.42, 1);

		mp.ispisi();

	}


	}
```

Šta predstavlja atribut ***profiti?***

Koje parametre prima metoda ***unesiProfit?***

Šta radi metoda ***ispisi***? (objasniti detaljno)

Niz koji je dat u prethodnom zadatku se uvek koristi do maksimalnog kapaciteta. Šta to znači? Niz ima dvanaest elemenata i uvek se koriste svi elementi (svaki predstavlja iznos profita za neki mesec u toku godine). Međutim, moguće su i situacije u kojima se ne koristi pun kapacitet niza.

Primer 2. Napraviti klasu ***OceneNaIspitnomRoku*** koja ima:

- Atribut ***ocene*** koji predstavlja ocene studenata na ispitnom roku. Zna se da ispit može maksimalno da polaže 100 studenata.
- Atribut ***brojac*** koji predstavlja trenutni broj elemenata u nizu. Početna vrednost neka bude 0.
- Metodu ***unesiOcenu*** koja kao parametar dobija ocenu na ispitu i unosi je u niz i to na prvo slobodno mesto.
- Metodu koja na ekranu ispisuje sve ocene na ispitnom roku.

Napraviti klasu ***TestOceneNaIspitnomRoku*** koja kreira jedan objekat klase ***OceneNaIspitnomRoku,*** unosi u njega ocene 6, 9, 7, 10, 9, 9, 8, 5, 7 i ispisuje sve ocene na ekranu.

Kod primera:

```
package ocenenaispitnomroku;

class OceneNaIspitnomRoku {
	int[] ocene = new int[100];
	int brojac = 0;

	void unesiOcenu(int ocena) {

		ocene[brojac] = ocena;

		//novi element je dodat u niz
		//pa je potrebno uvecati brojac
		//za jedan.
		brojac++;
	}

	void ispisi() {

		for(int i=0; i<brojac; i++) {
			System.out.println(ocene[i]);
		}
```

```
	}
}
public class TestOceneNaIspitnomRoku {

	public static void main(String[] args) {

		OceneNaIspitnomRoku oir = new OceneNaIspitnomRoku();

		oir.unesiOcenu(6);
		oir.unesiOcenu(9);
		oir.unesiOcenu(7);
		oir.unesiOcenu(10);
		oir.unesiOcenu(9);
		oir.unesiOcenu(9);
		oir.unesiOcenu(8);
		oir.unesiOcenu(5);
		oir.unesiOcenu(7);
		oir.ispisi();
	}


	}
```

Šta predstavlja promenljiva ***brojac*** u okviru metode ***unesiOcenu?***

Po čemu se metoda ***ispisi*** razlikuje u odnosu na metodu iz prethodnog primera?

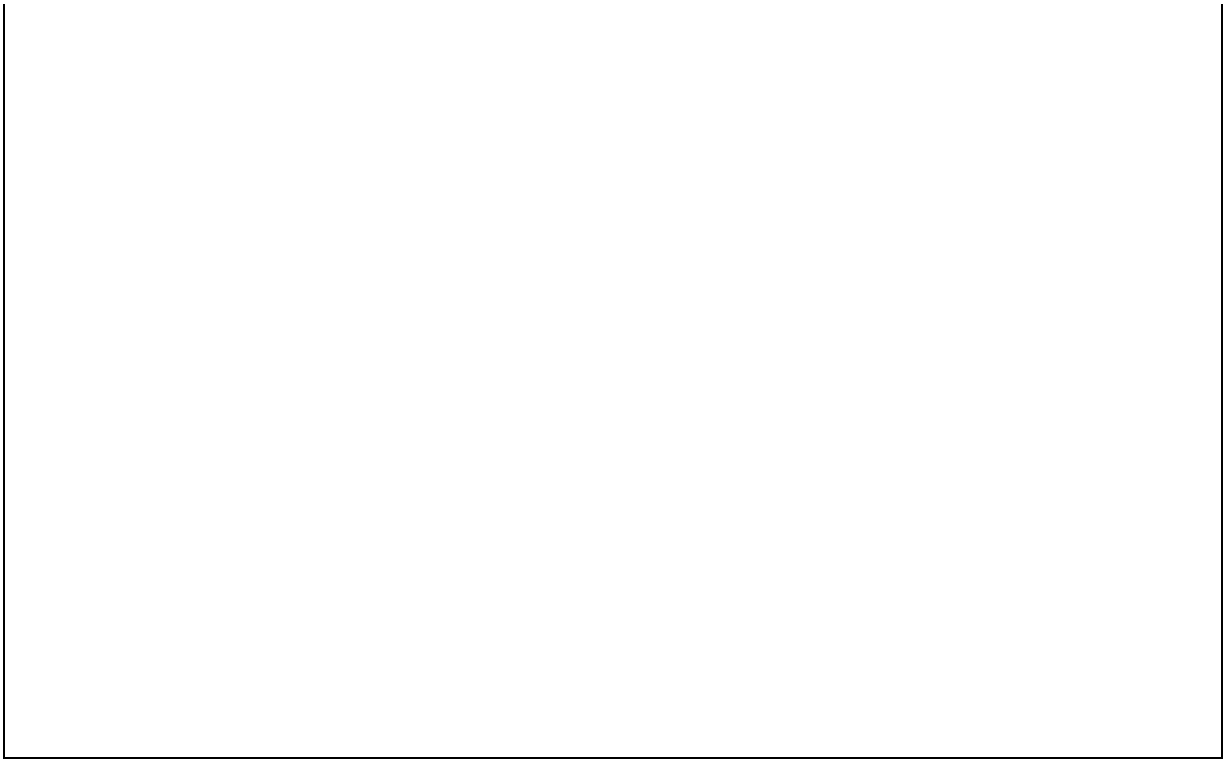
Zadaci za samostalni rad studenata

Zadatak 1. Potrebno je napraviti klasu ***Autobus*** koja ima:

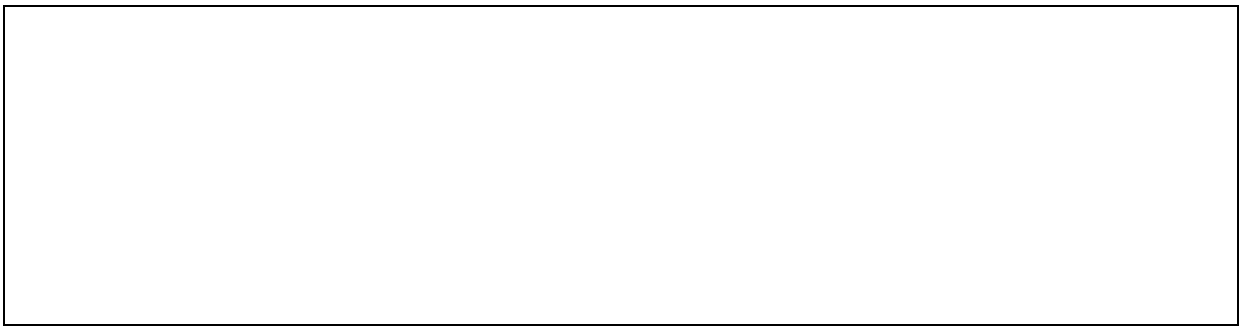
- Atribut koji predstavlja niz sedišta u autobusu. Svako sedište može da bude slobodno ili zauzeto. Ako je slobodno, vrednost odgovarajućeg lementa niza je TRUE, a ako je zauzeto, onda je FALSE. Autobus ima tačno 50 sedišta.
- Konstruktor koji predstavlja vrednost svih sedišta iz niza na slobodna TRUE.
- Metodu ***uvediPutnika***za uvođenje putnika u autobus. Ova metoda prima kao parametar broj sedišta na koje bi trebalo uvesti putnika (brojevi sedišta su od 0 do 49). Ako je sedište slobodno TRUE, sedište postaje zauzeto FALSE, a ako je već bilo zauzeto ispisuje se poruka o grešci.
- Metodu ***imaSlobodnihMesta*** koja proverava da li ima slobodnih mesta u autobusu. Metoda vraća TRUE ako ima slobodnih mesta, u suprotnom FALSE.
- Metodu ***brojSlobodnihMesta***koja vraća broj slobodnih mesta u autobusu.
- Metodu ***brojZauzetihMesta*** koja vraća broj zauzetih mesta u autobusu.
- Metodu ***ispisiStatusMesta*** koja ispisuje status svakog sedišta iz autobusa u obliku "***Sedište broj __ je slobodno***" ili "***Sedište broj __ je zauzeto***".

Potrebno je napraviti klasu ***TestAutobus*** koja kreira jedan objekat klase ***Autobus*** i uvodi u njega četiri putnika: na drugo, trideseto, četrdeset drugo i pretposlednje mesto u autobusu. Posle toga potrebno je ispisati statuse svih mesta u autobusu.

!Napomena: Svaki student je dužan da za primer uzme nasumični broj mesta na koja će da rasporedi putnike!



Izlaz iz programa:



Zadatak 2. Napraviti klasu ***NizCena*** koja predstavlja niz cena raznih proizvoda i ima:

- Atribut koji predstavlja niz cena proizvoda (primer cene: 70.4 dinara). Maksimalni kapacitet niza je uvek 100 elemenata.
- Atribut koji predstavlja brojač elemenata niza. Brojač na početku ima vrednost nula, jer je niz prazan.
- Metodu ***dodajCenu*** za dodavanje nove cene u nizu. Dodavanje se vrši samo ako je nova cena veća od nule i ako u nizu ima mesta (brojač je manji od maksimalnog kapaciteta). U suprotnom potrebno je ispisati poruku o grešci. Ako se ubacivanje izvrši, potrebno je brojač uvećati za jedan.
- Metodu ***prosecnaCena*** koja izračunava i vraća prosečnu cenu proizvoda. Ukoliko je niz prazan, ispisuje se poruk da je niz prazan i vraća se nula.
- Metodu ***najnizaCena*** koja vraća najnižu cenu proizvoda. Ukoliko je niz prazan, ispisuje se poruka da je niz prazan i vraća se nula.
- Metodu ***najvisaCena*** koja vraća najvišu cenu proizvoda. Ukoliko je niz prazan, ispisuje se poruka da je niz prazan i vraća se nula.

- Metodu ***razlikaMaxMin*** koja vraća razliku između najniže i najviše cene proizvoda. Ukoliko je niz prazan ispisuje poruku da je niz prazan i vraća se nula.
- Metodu ***ispisiCeneVeceOd*** koja ispisuje samo one cene proizvoda koje su veće od neke zadate vrednosti. Ta vrednost se unosi u metodu kao parametar. Ukoliko je niz prazan, ispisuje se poruka o tome.

Napisati klasu ***TestNizCena*** koja kreira jedan objekat klase ***NizCena***. Svaki student je dužan da testira sve metode i na kraju ispiše rezultate u deo predviđen za to.

Izlaz iz programa:

Višedimenzionalni nizovi (matrice)

Višedimenzionalni podaci (npr. matrice) mogu da se predstave u Javi korišćenjem **višedimenzionalnih nizova. Višedimenzionalni niz** je veoma sličan jednodimenzionalnom nizu u svakom pogledu, a njegova deklaracija se vrši na sledeći način (**stavlja se onoliko uglastih zagrada koliko podaci imaju dimenzija**):

```
tip_podataka[][] nazivPromenljive;
```

Ovo je deklaracija dvodimenzionalnog niza. Od svih višedimenzionalnih nizova se, u principu, najčešće koriste dvodimenzionalni nizovi i to u onim slučajevima kada je potrebno predstaviti neku matricu ili tabelu sa podacima.

I ovde je potrebna inicijalizacija niza, sa tim što se pri inicijalizaciji **navodi kapacitet niza po svakoj dimenziji:**

```
nazivPromenljive = new tip_podataka[ceo_broj_1][ceo_broj_2];
```

Broj u prvoj zagradi predstavlja **broj redova** matrice, a broj u drugoj zagradi predstavlja **broj kolona** iste. Pristup elementima dvodimenzionalnog niza se takođe vrši preko indeksa, ali svaka dimenzija ima svoj indeks pa se, u slučaju dvodimenzionalnog niza, **obavezno navode oba indeksa (prvi za red a drugi za kolonu).**

```
nazivPromenljive[indeks_1][indeks_2];
```

U radu sa dvodimenzionalnim nizovima se načešće koriste dve ugnježdene For petlje, pri čemu brojač svake petlje glumi indeks jedne dimenzije niza.

Primer 3. Napraviti klasu ***Matrica*** koja ima:

- Atribut matrica koji predstavlja dvodimenzionalni niz celih brojeva.
- Konstruktor koji kao parametar prima broj redova i broj kolona koji matrica treba da ima i inicijalizuje atribut matrica na te kapacitete.
- Matodu koja pretvara matricu u nula matricu – matricu čiiji su svi elementi jednaki nuli.
- Metodu koja matricu pretvara u jediničnu matricu tj. podešava vrednosti elemenata matrie tako da elementi koji se nalaze na glavnoj dijagonali budu jednaki jedinici, a ostali elementi budu jednaki nuli. Metoda prvo proverava da li je matrica kvadratna (da li ima sti broj redova i kolona), ako nije ispisuje poruku o grešci.
- Metodu koja na ekranu ispisuje vrednosti elemenata matrice i to tako da se u jednom redu na ekranu ispišu svi elementi koji pripadaju istom redu matrice.

Napraviti klasu ***TestMatrica*** koja kreira jedan objekat klase ***Matrica*** dimenzija 5x5 i poziva njegove metode.

```java
package Matrica;

class Matrica{
	int[][] matrica;

	Matrica(int brojRedova, int brojKolona){
		matrica = new int[brojRedova][brojKolona];
	}

	void nulaMatrica() {
		for(int i=0; i<matrica.length; i++) {
			for(int j=0; j<matrica[0].length; j++) {
				matrica[i][j]=0;
			}
		}
	}

	void jedinicnaMatrica() {
		if(matrica.length == matrica[0].length) {
			for(int i = 0; i<matrica.length; i++) {
				for(int j=0; j<matrica[0].length; j++) {
					if(i == j) {
						matrica[i][j]=1;
					}else {
						matrica[i][j]=0;
					}
				}
			}
		}else {
			System.out.println("Greska, matrica nije kvadratna!");
		}
	}

	void ispisi() {
		for(int i=0; i<matrica.length; i++) {
			for(int j=0; j<matrica[0].length; j++) {
				System.out.print(matrica[i][j] + " ");
			}
			System.out.println();
		}

	}
}
public class TestMatrica {

	public static void main(String[] args) {
		Matrica m = new Matrica(5,5);

		m.nulaMatrica();
		m.ispisi();

		System.out.println("---------");

		m.jedinicnaMatrica();
		m.ispisi();


		}
```

Kako se dobija broj redova, a kako broj kolona matrice?

Zaključak

U Nišu _____

Potpis _____