8. Nizovi

Sadržaj

- Nizovi primitiva
- Nizovi referenci na objekte

Nizovi

- Nizovi omogućavaju upravljanje skupom promenljivih istog tipa korišćenjem jedne reference
- Promenljiva niza je uvek referenca
- Nizovi u Javi su fiksne dužine

Nizovi primitiva

- Deklaracija promenljive niza

  ```
  int[] arrayOfInts;
  ```
- Promenljiva niza je referenca na objekat niza vrednosti primitiva
- Bez inicijalizacije ona ima vrednost `null`
  - Pokušaj korišćenja neinicijalizovane promenljive niza će za posledicu imati pojavu izuzetka `NullPointerException`
- Inicijalizacija promenljive niza - kreiranje niza primitiva

  ```
  int[] anotherArrayOfInts = new int[3];
  ```
- Kreirani niz primitiva ima sve elemente niza inicijalizovane na podrazumevanu vrednost za taj tip

Inicijalizacija elemanata niza

- Ukoliko su prilikom kreiranja niza poznati njegovi elementi, moguće je u istoj naredbi izvršiti kreiranje i inicijalizaciju elemenata niza

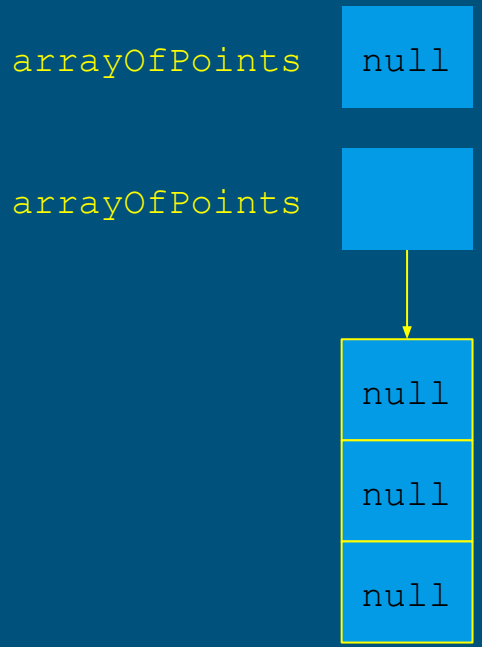
```
int[] arrayOfInts = new int[] {3, 6, 0, 4};
```

- Elementima niza se pristupa pomoću indeksa elementa
- Elementi su indeksirani od 0 do n-1

```
int someVar = arrayOfInts[0];      // 3
int anotherVar = arrayOfInts[3];   // 4
int niceTry = arrayOfInts[4];      // greska
arrayOfInts[1] = 7;                // {3 , 7, 0, 4}
```

Nizovi referenci na objekte

- Ukoliko se ne inicijalizuje promenljiva niza je `null`
  `Point[] arrayOfPoints;`
- Nakon kreiranja niza, inicijalno su svi elementi niza `null`
  `arrayOfPoints = new Point[3];`
- Da bi se mogli koristiti, potrebno je da se elementi niza inicijalizuju
  `arrayOfPoints[0] = new Point(6, 7);`



Rad sa elementima niza referenci na objekte

- Elementi niza mogu da se tretiraju kao posebne promenljive datog tipa

```
Circle[] olympicCircles = new Circle[5];
olympicCircles[0] = new Circle(new Point(100, 100), 30);
…
if (olympicCircles[3].getRadius() < 30)
  …
olympicCircles[2].setCenter(new Point(300, 100));
```

- Mogu da se porede sa `null`

```
if (olympicCircles[4] == null)
  olympicCircles[4] = new Circle(...);
```

Prolazak kroz niz pomoću `for` petlje

- Svaki niz ima predefinisano obeležje `length`
- Oubičajen način za prolazak kroz sve elemente niza je pomoću `for` petlje

```
Circle[] circles = new Circle[10];
…
double totalArea = 0;
for (int i = 0; i < circles.length; i++) {
  if (circles[i] != null)
    totalArea = totalArea + circles[i].area();
}
System.out.println("total area = " + totalArea);
```

Prevazilaženje problema fiksne dužine niza

- Kreiranje novog niza i prepisivanje svih elemenata starog

```
Point[] points = new Point[10];
Point newPointToAdd = new Point(55, 44);
if (points[9] != null) {                        // popunjen niz
  Point[] temp = new Point[20];
  for (int i = 0; i < points.length; i++)
    temp[i] = points[i];
  points = temp;
  points[10] = newPointToAdd;
}
```

- Klasa `ArrayList` obezbeđuje rad sa skupovima referenci varijabilne dužine

Rezime

- Nizovi u Javi su fiksne dužine
- Svaka promenljiva tipa niz je referenca na objekat
- Svaki niz ima predefinisano obeležje `length`
- Elementi nizova primitiva se prilikom kreiranja niza inicijalizuju na podrazumevane vrednosti za dati tip
- Elementi nizova referenci na objekte se prilikom kreiranja niza inicijalizuju na `null` vrednosti