Leksion 6

Referenca this

Deklarimi i vektorëve dhe matricave në Java

1. Referenca *this*

Fjala çelës this është një variabël reference në Java, që mund të përdoret brenda çdo metode për tiu referuar objektit aktual.

this është një referencë mbi objektin mbi të cilin është thirrur metoda.

Variablat lokalë janë variablat e përcaktuar brenda metodave, konstruktorëve ose blloqeve. Variablat lokalë deklarohen dhe inicializohen brenda metodave dhe shkatërrohen kur metoda të ketë përfunduar. *Variablat e instancës* janë variablat që përcaktohen brenda një klase por jashtë çdo metode. Ato inicializohen kur ngarkohet klasa. Këta variabla mund të aksesohen brenda çdo metode, konstruktori ose blloku të një klase të veçantë. *Variablat e klasës* janë variablat që deklarohen në një klasë, jashtë çdo metode me fjalën çelës static.

1.1 Përdorimi i this për tiu referuar variablave të instancës të klasës aktuale.

Kur një variabël lokal ka emër të njëjtë me një variabël instance, variabli lokal e fsheh variablin e instancës.

Shembull:

```
public class Test {
    public static void main(String[] args){
        Objekt obj1 = new Objekt();
        obj1.shfaqNr();
    }
}
class Objekt{
    int nr = 1; //variabel instance
    public void shfaqNr()
    {
        int nr = 2; //variabel lokal
        System.out.println(nr);
    }
}
```

Afishon: 2

Shembull

```
public class Test {
    public static void main(String[] args){
        Objekt obj1 = new Objekt();
        obj1.shfaqNr();
    }
}
class Objekt{
    int nr = 1; //variabel instance
    public void shfaqNr()
    {
        int nr = 2; //variabel lokal
        System.out.println(this.nr);
    }

}
```

1

Afishon : 1

this.shfaqNr ka kuptimin “thirr objektin aktual që nëpërmjet metodës shfaqNr të afishojë variablin e tij nr”.

Emri i një variabli mund të përdoret si emër parametri në një metodë. Në këtë rast variabli është i fshehur në metodë. Ju duhet ti referoheni emrit të variablit në metodë në mënyrë që ti caktoni atij një vlerë. Një variabël statik i fshehur mund të aksesohet duke përdorur referencën

EmriKlasës.emriVariablitStatik.

Një variabël instance i fshehur mund të aksesohet duke përdorur fjalën çelës this.

```
class Studenti{
        int id;
        String emri;

        Studenti(int id,String emri){
                id = id;
                emri = emri;
        }
        void shfaq(){
           System.out.println(id + " " + emri);
        }

        public static void main(String args[]){
        Studenti st1 = new Studenti(111,"Emi");
        Studenti st2 = new Studenti(321,"Ana");
        st1.shfaq();
        st2.shfaq();
        }
}
```

Afishon:

0 null

0 null

Në shembullin e mësipërm, parametrat formalë dhe variablat e instancës janë të njëjtë. Për ti dalluar ata nga njëri-tjetri përdoret referenca this:

```
class Studenti{
        int id;
        String emri;

        Studenti(int id,String emri){
                this.id = id;
                this.emri = emri;
        }
        void shfaq(){
           System.out.println(id + " " + emri);
        }

        public static void main(String args[]){
        Studenti st1 = new Studenti(111,"Emi");
        Studenti st2 = new Studenti(321,"Ana");
        st1.shfaq();
        st2.shfaq();
        }
}
```

Afishon:

```
111   Emi
321   Ana
```

this.id = id; do të thotë ti caktohet vlera id (e parametrit), fushës së të dhënës id të objektit. Nëse emrat e variablave të instancës dhe të parametrave (variablave lokalë) janë të ndyshëm, përdorimi i fjalës çelës this është i panevojshëm. Deri tani kemi parë shumë raste të tilla.

1.2 Thirrja e kostruktorit *this()* për të thirrur konstruktorin e klasës

Përdoret zakonisht kur keni disa konstruktorë në klasë dhe ju doni ti ripërdorni ata.

Shembull:

```
public class Studenti {
        int id;
        String emri;

        Studenti(){
           System.out.println("eshte thirrur konstruktori default");

        }

        Studenti(int id,String emri){
                    this();
                    this.id = id;
                    this.emri = emri;
        }
        void shfaq(){
            System.out.println(id + " " + emri);
        }

        public static void main(String args[]){
        Studenti st1 = new Studenti(111,"Emi");
        Studenti st2 = new Studenti(321,"Ana");
        st1.shfaq();
        st2.shfaq();
        }
}
```

Afishon:

```
eshte thirrur konstruktori default
eshte thirrur konstruktori default
111   Emi
321   Ana
```

Shembull:

```
public class Studenti {
        int id;
        String emri;
        int mosha;

        Studenti (int id, String emri){
                this.id = id;
                this.emri = emri;
        }

        Studenti(int id,String emri, int mosha){
                this(id, emri); //nuk nevojitet te inicializohen id dhe emri
```

```
                                        //therret kontruktorin e pare
this.mosha = mosha;
        }
        void shfaq(){
                System.out.println(id + " " + emri + " " + mosha);
        }

        public static void main(String args[]){
        Studenti st1 = new Studenti(111,"Emi");
        Studenti st2 = new Studenti(321,"Ana", 29);
        st1.shfaq();
        st2.shfaq();
        }
}
```

Afishon:

```
111 Emi 0
321 Ana 29
```

Kujdes! *this() duhet të jetë statementi i parë në konstruktor.*

2. Deklarimi i vektorëve dhe matricave në Java

2.1 Vektorët (ArrayList-at)

Vektorët në Java janë tabela të cilat mund ta ndryshojnë madhësinë e tyre. ArrayList është një klasë në libraritë standarde të Java që mund të mbajë çdo tip objekti. Një ArrayList është një objekt që mund të zmadhohet ose zvogëlohet ndërkohë që programi ekzekutohet. Në mënyrë që të përdorim klasën Arraylist, fillimisht ajo duhet të importohet duke vendosur statement-in e mëposhtëm në krye të skedarit tuaj:

```
import java.util.ArrayList;
```

Një vektor (ArrayList) stringjesh, të cilit i referohet një variabël reference i quajtur emrat deklarohet në këtë mënyrë:

```
ArrayList<String> emrat = new ArrayList<String>();
```

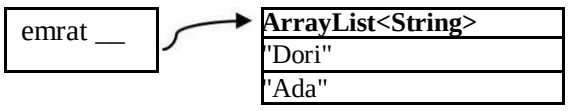
Objekti i krijuar është i tipit ArrayList<String>. Kur ju ndërtoni një objekt ArrayList, ai e ka madhësinë 0. Por ju mud të përdorni një vektor të një tipi tjetër. Për shembull, nëse Lojtarët është një klasë që përfaqëson lojtarët e një loje, ne mund të krijojmë një vektor të lojtarëve nëpërmjet statement-it:

```
ArrayList<Lojtarët> listaLojtarëve = new ArrayList<Lojtarët>(5);
```

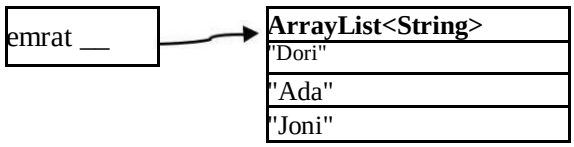
Ky vektor e ka gjatësinë 5. Ju përdorni metodën add për të shtuar një objekt në fund të vektorit:

```
emrat.add("Dori");//tashmë vektori e ka madhësinë 1 dhe ka elementin “Dori” emrat.add("Ada"); //vektori ka madhësinë 2 dhe elementët “Dori” dhe “Ada” emrat.add("Joni"); //vektori ka madhësinë 3
```

Përpara shtimit:



Pas shtimit:



Për të marrë vlerën e një elementi të vektorit, përdoret metoda get dhe jo operatori []. Si në rastin e tabelave, vlera e indeksit nis nga 0. Për shembull emrat.get(2) kthen elementin me indeks 2, pra elementin e tretë në vektor:

```
String emri = emrat.get(2);
```

Si në rastin e tabelave, është gabim të aksesohet një element që nuk ekziston. Indeksi i fundit në vektor është emrat.size() – 1. Për ti caktuar një elementi në vektor një vlerë të re, përdoret metoda set.

```
emrat.set(2, "Era");
```

Vendos vlerën "Era" tek pozicioni 2 në vektorin emrat, duke mbishkruar çdo vlerë që ka qënë më parë aty. Metoda set mund të mbishkruajë vetëm vlerat ekzistuese. Është ndryshe nga metoda add, e cila shton një objekt të ri në fund të vektorit. Ju mund të fusni një objekt në mes të vektorit.

```
emrat.add(1,"Blerta"); //shton stringun "Blerta" ne indeksin 1
```

Gjithë elementët me indeks 1 ose më të madh se 1 l ëvizin me një pozicion dhe stringu "Blerta" shtohet tek indeksi 1. Pas çdo thirrje të metodës add, madhësia e vektorit rritet me 1. Metoda remove fshin nj ë element në indeksin e dhënë dhe i lëviz gjithë elementet pas elementit të fshirë tek indeksi një më i vogël se indeksi ku ndodhen, dhe redukton madhësinë e vektorit me 1.

```
emrat.remove(1); //fshin elementin në indeksin 1
emrat.remove("Blerta");//fshin elementin "Blerta" nga vektori
```

Për të gjetur madhësinë e vektorit përdoret metoda size().

```
emrat.size(); //kthen numrin e elementëve në vektor
```

Metoda indexOf përdoret për të kërkuar për një element në vektor, nëse elementi gjendet në vektor kthen indeksin ku ai gjendet, në të kundërt kthen -1.

```
emrat.indexOf("Blerta"); //kërkon për elementin Blerta në vektorin emrat dhe kthen numrin e pozicionit ku ai ndodhet.
```

Metoda boolean-e contains përdoret për të kontrolluar nëse një element gjendet në vektor.

```
emrat.contains("Blerta"); //kontrollon nëse elementi "Blerta" gjendet në vektor,
                          //nëse gjendet kthen true, në të kundërt kthen false
```

Për të fshirë të gjithë elementët e një vektori përdoret metoda clear().

```
emrat.clear(); //fshin gjithë elementët e vektorit me variabël reference vlerat.
```

Për të afishuar elementët e vektorit emrat përdoret cikli for:

```
for ( int i = 0; i < emrat.size(); i++ ) {
String emri = emrat.get(i);
System.out.println(emri);
}
```

Ju mund të përdorni dhe ciklin for-each me vektorët, kështu elementët e emrat mund të afishohen:

```
for ( String s: emrat ) {

System.out.println(s);

}
```

Variabli s merr vlerën e çdo elementi në vektor.

2.2 Matricat (ArrayList-at dy dimensionale)

Një matricë (ArrayList dy-dimensionale) është një vektor (ArrayList) i vektorëve(Arraylist-ave). Deklarimi i një variabli matrice, caktimi i tij dhe krijimi i matricës bëhet nëpërmjet statement-it:

ArrayList<ArrayList<String>> emrat = new ArrayList<ArrayList<String>>(); //madhësia e matricës është 0.

Ushtrime

Ushtrimi 1

Shkruani nj ë program që ndërton dy objekte punonjës, ku për secilin prej tyre të afishojë pagën dhe bonusin që merr secili, si dhe të llogarisë pagën totale të secilit dhe ta afishojë atë.

Mënyra e parë:

```
public class Punonjes {

	double paga;
	double bonus;
	//konstruktori mungon, nje konstruktor default krijohet nga kompilatori

	double llogaritPagenTotale(){
		return paga + bonus;
	}
}


public class TestPunonjes {

	public static void main(String[] args) {
		Punonjes aleks = new Punonjes();
		aleks.paga = 40000.0;
		aleks.bonus = 1200.0;
		System.out.println("Aleksi paguhet " + aleks.paga +
		  "   leke dhe ka marre nje bonus prej " + aleks.bonus + " lekesh."); System.out.println ("Paga
		totale e Aleksit eshte " +
					aleks.llogaritPagenTotale() + " leke.");

		Punonjes megi = new Punonjes();
		megi.paga = 45000.0;
		megi.bonus = 1400.0;
		System.out.println("Megi paguhet " + megi.paga +
		  "   leke dhe ka marre nje bonus prej " + megi.bonus + " lekesh."); System.out.println ("Paga
		totale e Megit eshte " +
					megi.llogaritPagenTotale() + " leke.");

	}


}
```

Programi printon:

```
Aleksi paguhet 40000.0 leke dhe ka marre nje bonus prej 1200.0 lekesh.
Paga totale e Aleksit eshte 41200.0 leke.
Megi paguhet 45000.0 leke dhe ka marre nje bonus prej 1400.0 lekesh.
Paga totale e Megit eshte 46400.0 leke.
```

Mënyra e dytë

```
class Punonjes {

        double paga;
        double bonus;
        double pagaTotale;

        Punonjes(){
        }

        double llogaritPagenTotale(){
        return pagaTotale = paga + bonus;
        }
}
```

Kurse klasa TestPunonjës mbetet siç ishte në mënyrën e parë. Krijohet një konstruktor pa parametra, i cili thirret me operatorin new për të krijuar objektet.

Mënyra e tretë

```
class Punonjes {

        double paga;
        double bonus;
        double pagaTotale;

        Punonjes(){
                paga = 40000.0;
                bonus = 1200.0;
        }

        Punonjes(double p, double b){
                paga = p;
                bonus = b;
        }

        double llogaritPagenTotale(){
             pagaTotale = paga + bonus;
             return pagaTotale;
        }
}

public class TestPunonjes {

       public   static void main(String[] args) { Punonjes aleks = new Punonjes();
                System.out.println("Aleksi paguhet " + aleks.paga +
                  "   leke dhe ka marre nje bonus prej " + aleks.bonus + " lekesh."); System.out.println ("Paga
                totale e Aleksit eshte " +
                                aleks.llogaritPagenTotale() + " leke.");

                Punonjes megi = new Punonjes(45000.0, 1400.0);
                System.out.println("Megi paguhet " + megi.paga +
                  "   leke dhe ka marre nje bonus prej " + megi.bonus + " lekesh."); System.out.println ("Paga
                totale e Megit eshte " +
                                megi.llogaritPagenTotale() + " leke.");

        }
}
```

Mënyra e katërt

```
class Punonjes {

	double paga;
	double bonus;
	double pagaTotale;

	Punonjes(double p, double b){
		paga = p;
		bonus = b;
	}

	double llogaritPagenTotale(){
	    pagaTotale = paga + bonus;
	    return pagaTotale;
	}
}

public class TestPunonjes {

public static void main(String[] args) {

Punonjes aleks = new Punonjes(40000.0, 1200.0); System.out.println("Aleksi paguhet " + aleks.paga + leke dhe ka
marre nje bonus prej " + aleks.bonus + " lekesh.");
System.out.println ("Paga totale e Aleksit eshte " + aleks.llogaritPagenTotale() + " leke.");

Punonjes megi     = new     Punonjes(45000.0, 1400.0);

System.out.println("Megi paguhet " + megi.paga + leke dhe ka marre nje bonus prej " + megi.bonus + " lekesh.");
System.out.println ("Paga totale e Megit eshte " + megi.llogaritPagenTotale() + " leke.");
}
}
```

Mënyra e pestë

```
public class Punonjes {

	public static void main(String[] args) {
Punonjes aleks = new Punonjes(40000.0, 1200.0); System.out.println("Aleksi paguhet " + aleks.paga + leke dhe ka
marre nje bonus prej " + aleks.bonus + " lekesh.");
System.out.println ("Paga totale e Aleksit eshte " +aleks.llogaritPagenTotale() + " leke.");
Punonjes megi = new Punonjes(45000.0,      1400.0);
System.out.println("Megi paguhet " + megi.paga + leke dhe ka marre nje bonus prej " + megi.bonus + " lekesh.");
System.out.println ("Paga totale e Megit eshte " + megi.llogaritPagenTotale() + " leke.");
	}

	double paga;
	double bonus;
	double pagaTotale;

	Punonjes(double p, double b){
		paga = p;
		bonus = b;
	}
```

```
        double llogaritPagenTotale(){
            pagaTotale = paga + bonus;
            return pagaTotale;

        }
}
```

Ushtrimi 2

Çfarë afishon programi i mëposhtëm?

```
import java.util.ArrayList;

public class Vektor1 {

        public   static void main(String[] args) {
        ArrayList<String> list = new ArrayList<String>(); //kur krijohet vektori 'list' ka 0
                elemente
        list.ensureCapacity(20); //tani 'list' ka 20 elemente, 20 caktohet si kapaciteti i
                vektorit

                    list.add("NJE");

                    list.add("DY");

                    list.add("TRE");

                    list.add("KATER");

      //redukton kapacitetin aktual ne madhesine aktuale te nje ArrayList, nga 20 nr. I elementeve behet 4
                    list.trimToSize();
                    System.out.println(list);
                    System.out.println("Madhesia e vektorit eshte " + list.size());
        }
}
```

Përgjigje:

[NJE, DY, TRE, KATER]

Madhesia e vektorit eshte 4