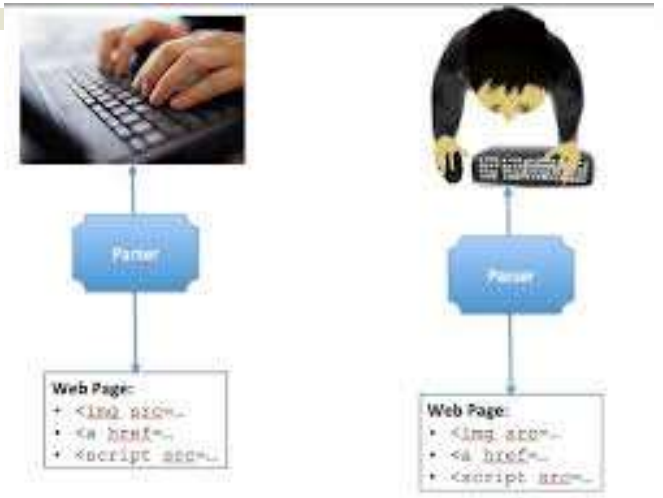
Programiranje



Nastava: prof.dr.sc. Dražena Gašpar

Datum: 17.04.2018.

Za Predavanje 10.04.2018.

Za Test 2:
- Poglavlje 4
- Poglavlje 5

DOMAĆA ZADAĆA:

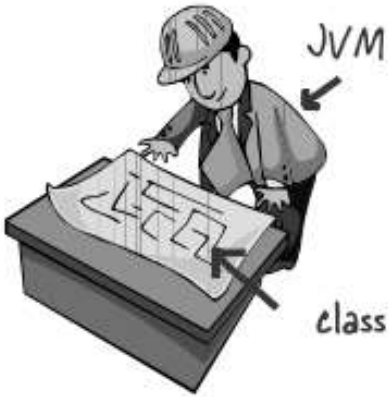
Kreirati dijagram toka i napraviti program koji izračunava najmanju prodaju za svaku prodavaonu (smješta ih u polje M) te prosječnu prodaju po prodavaonicama (smješta ih u polje P)

PRODAJA	Artikl 1	Artikl 2	Artikl 3	Artikl 4
Prodavnica 1	14	12	14	15
Prodavnica 2	5	4	4	4
Prodavnica 3	23	33	31	24

KLASA

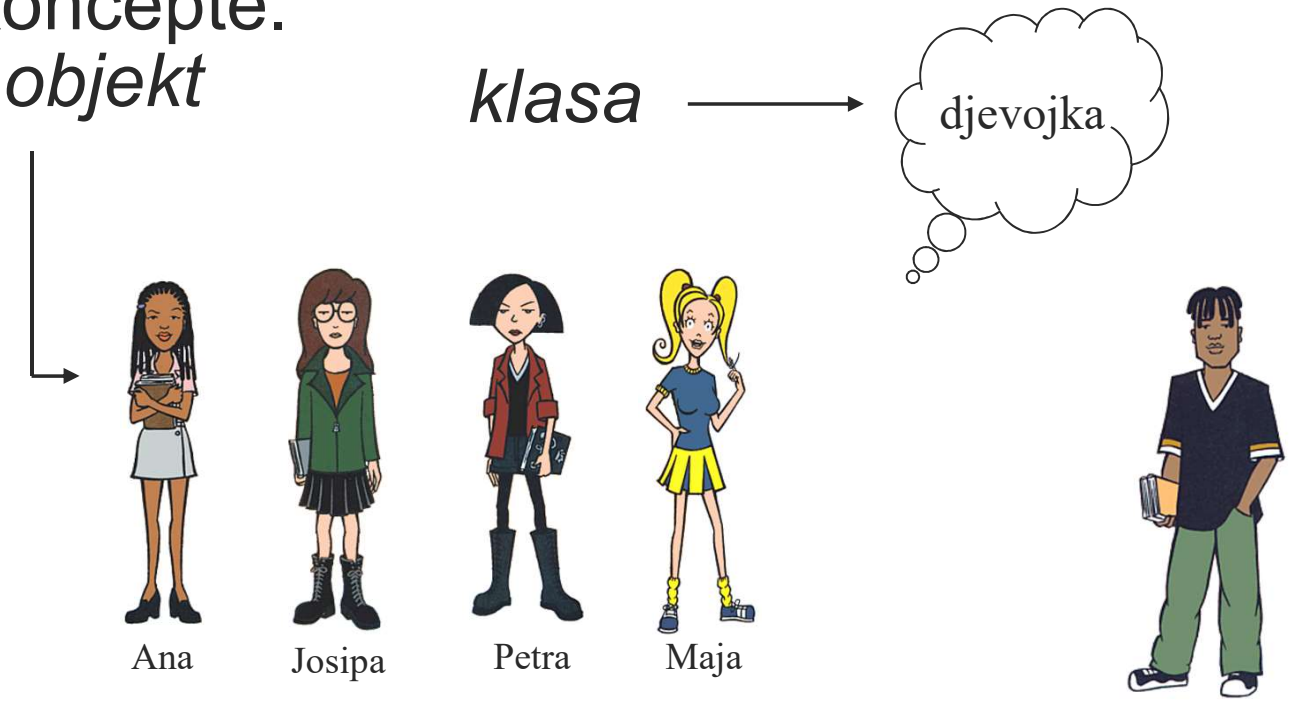
KLASA je apstrakcija (generički opis) za skup objekata s istim atributima i ponašanjem (operacijama)

KLASA je predložak za kreiranje objekata.



Objekt vs Klasa

- Klase predstavljaju koncepte, objekti predstavljaju instance koje utjelovljuju te koncepte.



KLASA

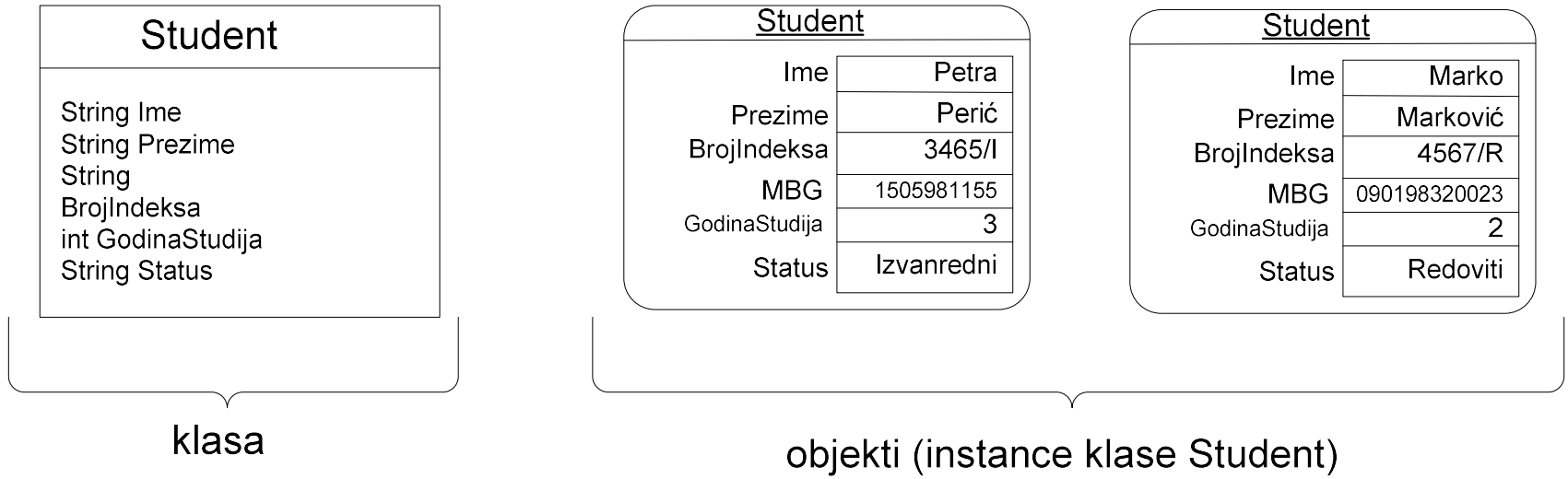
Promatrano s aspekta tipova, klasa je složeni tip podataka.

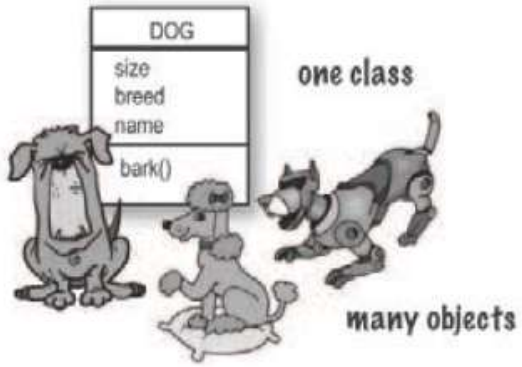
Definicija klase kreira instance klase.

Definicija klase prevodi atribute i ponašanje stvarnog objekta u simulaciju tog objekta unutar programa.

Ponašanja su naredbe (instrukcije) koje izvršavaju specifične zadatke a nazivaju se metode

KLASA


Student
String Ime
String Prezime
String
BrojIndeksa
int GodinaStudija
String Status
Student
Ime Petra
Prezime Perić
BrojIndeksa 3465/I
MBG 1505981155
GodinaStudija 3
Status Izvanredni
Student
Ime Marko
Prezime Marković
BrojIndeksa 4567/R
MBG 090198320023
GodinaStudija 2
Status Redoviti
klasa
objekti (instance klase Student)


DOG
size
breed
name
bark()
one class
many objects

KLASE

- **Pisanje JAVA programa podrazumijeva pisanje kolekcije definicija klasa.**
- **Definicija klase opisuje instancu klase i specificira podatke kojima upravlja objekt koji je instanca te klase, kao i funkcionalnost (upite i naredbe) koje objekt podržava.**
- **Definicija klase podrazumijeva slijedeće:**
  - **definiranje varijabli instance**
  - **definiranje varijabli klase**
  - **definiranje upita i naredbi – objedinjeno pod zajedničkim nazivom metode.**

KLASA

Varijabla instance (engl. instance variable) je imenovani memorijski prostor koji se koristi za pohranjivanje podataka objekta i dodjeljuje se objektu u trenutku njegovog kreiranja.

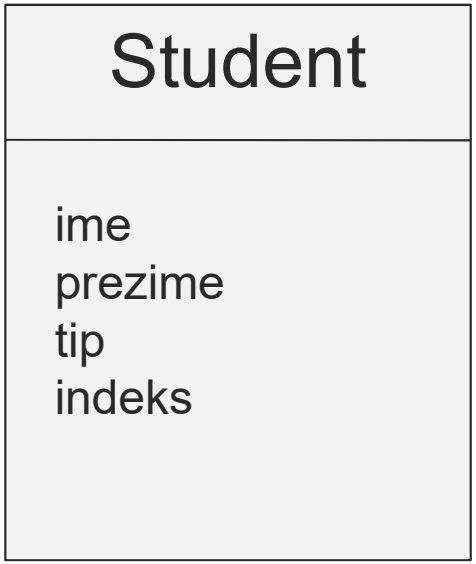
Svaka instanca klase (svaki objekt klase) ima svoj vlastiti primjerak te varijable.

KLASA

Varijabla klase (engl. class variable) podrazumijeva da klasa ima samo jedan primjerak te varijable koji sve instance te klase mogu koristiti.

Varijabla klase je element klase koji se koristi neovisno o instancama (objektima) klase.

KLASA




```
class student {
        String ime;
        String prezime;
        String tip;
        String indeks;
         }
```

KLASA

Instanca klase *Student* - konkretni objekt te klase, deklarira se kao:

Student Ana

Varijable instance i metode su članovi klase.

Članovima instance klase se pristupa tako što se koristi ime klase, operator točka i ime člana.

Operator točke (Dot operator)

- **Omogućava pristup varijablama instanci**
- **I ponašanju objekta (metode)**

KLASA

Članu klase – varijabli instance tip instance Ana klase s*tudent* pristupa se na sljedeći način:

Ana.tip = „redoviti“

Metodi *prikaziTip()* pristupa se na sljedeći način:

Ana.prikaziTip()

KLASA

Varijabla klase je element klase koji se koristi neovisno o instancama (objektima) klase.

static - ključna riječ za označavanje varijable klase

static tipVarijable nazivVarijable

Static metoda

- ⇒Ne radi s objektima
- ⇒ne postoji “this” za tu konkretnu metodu
- ⇒ne mogu se pozivati ne-static metode unutar static metoda (obrnuto je moguće)
- ⇒može se pozvati static metoda za samu klasu, bez i jednog objekta
- ⇒ekvivalent za global funkciju
- ⇒neki smatraju da static narušava objektnu orijentaciju (ne šalje se poruka objektu)

Main metoda

⇒Static metoda se može pozvati bez i jednog objekta i to je razlog što je Main static metoda

⇒Svaka klasa može imati Main metodu koja se u tom slučaju rabi za testiranje pojedinačnih klasa

Main metoda

```
public class Aplikacija {
  public static void main(String[] args) {
     …. // formiranje objekata
 }}
class djelatnik {
  public djelatnik(String n, double s,…)
    ….
public static void main(String[] args) // testiranje
  …
}}
```

Obje klase imaju main metodu

2 načina poziva:

java djelatnik ILI java Aplikacija

Opći oblik JAVA definicije klase

Definicija klase se obično sastoji od (redoslijed nije bitan):

- Modifikatora pristupa – definira raspoloživost klase iz drugih klasa
- Ključne riječi class – znak Javi da slijedeći blok definira klasu
- Varijable instance – Sadrži varijable i konstante koje koriste objekti klase
- Konstruktora – metoda koje imaju isti naziv kao i klasa, a koje se koriste za kontrolu inicijalnog stanja bilo kojeg kreiranog objekta klase
- Varijable klase – sadrže varijable i konstante koje pripadaju klasi i koje dijele svi objekti klase
- Metoda klase – metode koje se koriste za kontrolu vrijednosti varijabli klase, odnosno za implementiranje ponašanja klase

Opći oblik JAVA definicije klase

```
modifikatorPristupa class imeklase {
        tip varijabla1;
        tip varijabla2;
        tip varijabla3;
          ….
        tip varijablaN;

        tip imemetode1(lista parametara) {
              …. tijelo metode (programski kod)
        }
        tip imemetode2(lista parametara) {
              …. tijelo metode (programski kod)
        }

        ….
        tip imemetodeN(lista parametara) {
              …. tijelo metode (programski kod)
        }
}
```

Objekt

OBJEKT - bilo što ako je to sposobno osigurati ograničeni skup korisnih servisa - usluga.

Razumijevanje objekta se bazira na njegovoj javnoj prezentaciji: njegovom „obiteljskom imenu“ (klasi) i na „objavljenoj listi servisa“ (protokol) koje je voljan osigurati.

Svaki objekt ima svoje vlastito ime (naziv) koje ga jedinstveno identificira, dok „obiteljsko ime“ označava klasu sličnih objekata kojoj konkretni objekt pripada.

Objekt

Memorijski prostor za pohranjivanje podataka o objektu dodjeljuje se u trenutku kreiranja objekta i naziva se varijabla instance (engl. *instances variables*).

Varijabla instance se dodjeljuje za svako svojstvo objekta i sadrži vrijednost tog svojstva.

Varijabla instance je varijabla koja je stalni dio objekta, a memorijski prostor za nju se dodjeljuje pri kreiranju objekta

Objekt - stanje objekta

Objekti klase

Za kreiranje instanci klase u JAVA jeziku, odnosno objekata klase koriste se:

- **operator new i**
- **konstruktor.**

Primjeri:

```
String znakovniNiz = new String();
Random r = new Random();
Brojač prviBrojač = new Brojač(10);
```

…. Program Student1.java

Objekti klase

Pri kreiranju objekta potrebno je deklarirati varijablu čiji tip odgovara klasi, što nije ništa drugo do kreiranje novog tipa podatka koji se može koristiti za deklariranje objekata tog tipa.

Tako deklarirana varijabla ne definira objekt već samo pokazuje (referencira) na njega.

Stvarna, fizička kopija objekta se dodjeljuje toj varijabli pomoću operatora new. Operator new dinamički (tj. u trenutku izvršavanja programa) dodjeljuje memoriju za objekt i programu vraća referencu na njega. Ova referenca predstavlja adresu objekta stvorenog operatorom new u memoriji.

Objekti klase

B I T N O

u varijablu se smješta referenca na objekt, a ne objekt. Iz ovoga proizlazi da JAVA svim objektima klasa memoriju mora dodjeljivati dinamički.

Objekti klase

Primjer:

Brojač prviBrojač = new Brojač(10);

Može se rastaviti u dva koraka:

```
Brojač  prviBrojač;        // deklariranje reference na objekt
prviBrojač = new Brojač(10);  // dodjeljivanje memorije
                              objektu Brojač
```

Objekti klase

Opći oblik new operatora je:

NE POSTOJI
Varijabla tipa
OBJEKT
Ovo je referenca

```
varijabla = new NazivKlase();
```

Gdje je varijabla tip klase koja se pravi, a NazivKlase označava klasu čiji se objekt (instanca) pravi.

Operator new kreira novu instanca za klasu i dodjeljuje joj memoriju.

Slijedeći korak je poziv specijalne metode za inicijaliziranje objekta i postavljanje odgovarajućih početnih (inicijalnih) vrijednosti. Ta specijalna metoda se naziva konstruktor i ona kreira i inicijalizira nove instance klase. Obično se konstruktor izričito definira unutar definicije klase. Međutim, ako se konstruktor ne naznači eksplicitno, JAVA automatski osigurava tzv. podrazumijevani (engl. default) konstruktor.

Objekti – 3 osnovna koraka

1 3 2

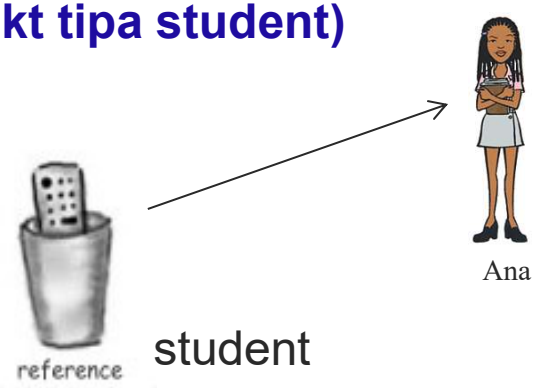
```
student  Ana  =  new student();
```

1. Deklariranje varijable koja sadrži referencu
(JVM dodjeljuje memoriju za referentnu varijablu)

student

2. Kreiranje objekta
(JVM dodjeljuje memoriju za novi objekt tipa student)

3. Povezivanje objekta i reference
(dodjeljuje referencu na objekt
Referentnoj varijabli)



student

Objekti klase

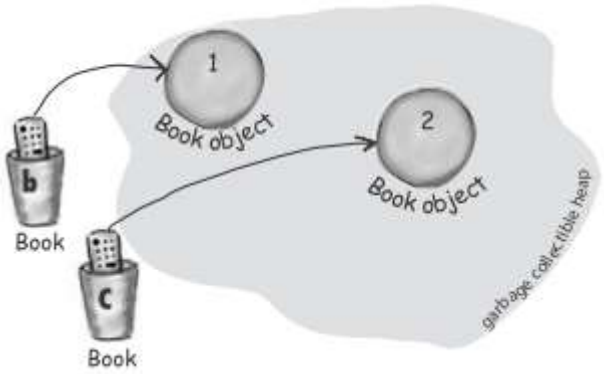
Kod dodjeljivanja referenci na objekte varijablama, odnosno kada se jedna referentna varijabla dodjeljuje drugoj, tada se ne pravi kopija objekta, već se samo kopira referenca na objekt:

```
Brojač prviBrojač = new Brojač(10);
Brojač drugiBrojač = prviBrojač;
```

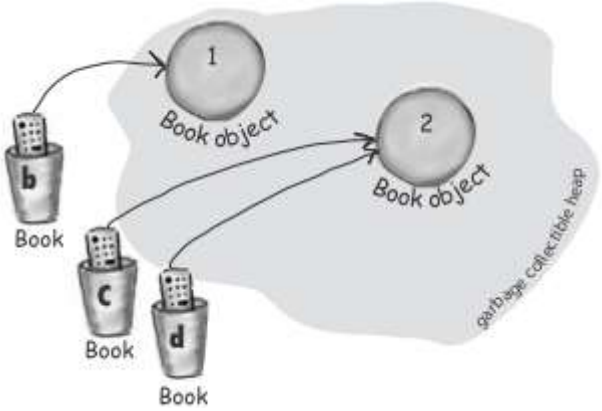
Objekti klase

Book b = new Book();

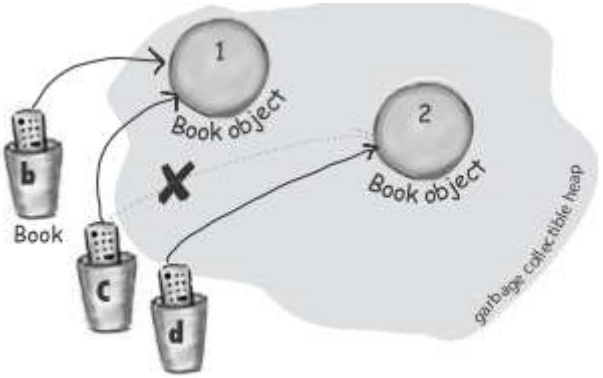
Book c = new Book();



Book d= c;



c=b;



Objekti klase

Da bi se moglo raditi s objektima potrebno je

1. Napraviti (konstruirati) objekt
2. Definirati početno (inicijalno) stanje
3. Pridružiti metode objektu

JAVA rabi konstruktore za pravljenje (konstruiranje) i inicijalizaciju objekata.

Inicijalni konstruktor

```
import java.util.Scanner;
public class Student2
{
 public static void main(String[] args)
 {
  Scanner unos = new Scanner(System.in);
  student Ana = new student();
  System.out.print("Ime studenta ");
  Ana.ime = unos.next();
  System.out.print("Prezime studenta ");
  Ana.prezime = unos.next();
  System.out.print("Status ");
  Ana.tip = unos.next();
  System.out.print("Broj indeksa ");
  Ana.indeks = unos.next();
  System.out.println(Ana.ime+" "+Ana.prezime+" "+Ana.tip+" "+Ana.indeks);
} }

class student
 {
   String ime;
   String prezime;
   String tip;
   String indeks;
   }
```

Program: Student2.java

Primjer konstruktora

```
class student
    { public student(String i, String p, String t, String n)
        { ime = i;
          prezime = p;
          tip = t;
          indeks = n; }

      public String uzmiime()
        { return ime; }

      public String uzmiprezime()
        { return prezime; }

      public String uzmitip()
        {return tip; }

      public String uzmiindeks()
        {return indeks; }

        private String ime, prezime, tip, indeks; }
```

KONSTRUKTOR

Metoda uzmiime

Metoda uzmiprezime

Metoda uzmitip

Metoda uzmiindeks

Varijable instance

Konstruktor

Pseudo metoda koja kreira objekt. To su instance metoda s UVIJEK istim imenom kao i njihove klase.

Zadaća konstruktora jeste inicijaliziranje objekta u procesu njegovog stvaranja, tako da se nakon naredbe “new” ima odmah spreman objekt za uporabu.

Objekti klase

Da bi se moglo raditi s objektima potrebno je

1. Napraviti (konstruirati) objekt
2. Definirati početno (inicijalno) stanje
3. Pridružiti metode objektu

JAVA rabi konstruktore za pravljenje (konstruiranje) i inicijalizaciju objekata.

Konstruktor

Pseudo metoda koja kreira objekt. To su instance metoda s UVIJEK istim imenom kao i njihove klase.

Zadaća konstruktora jeste inicijaliziranje objekta u procesu njegovog stvaranja, tako da se nakon naredbe “new” ima odmah spreman objekt za uporabu.

Metoda

Metoda je naziv dan dijelu (bloku) programskog koda koji se izvršava kao odgovor na specifičnu poruku, gdje je poruka formalna komunikacija poslana od jednog objekta drugom s ciljem izvršenja određenog servisa (usluge).

Metoda

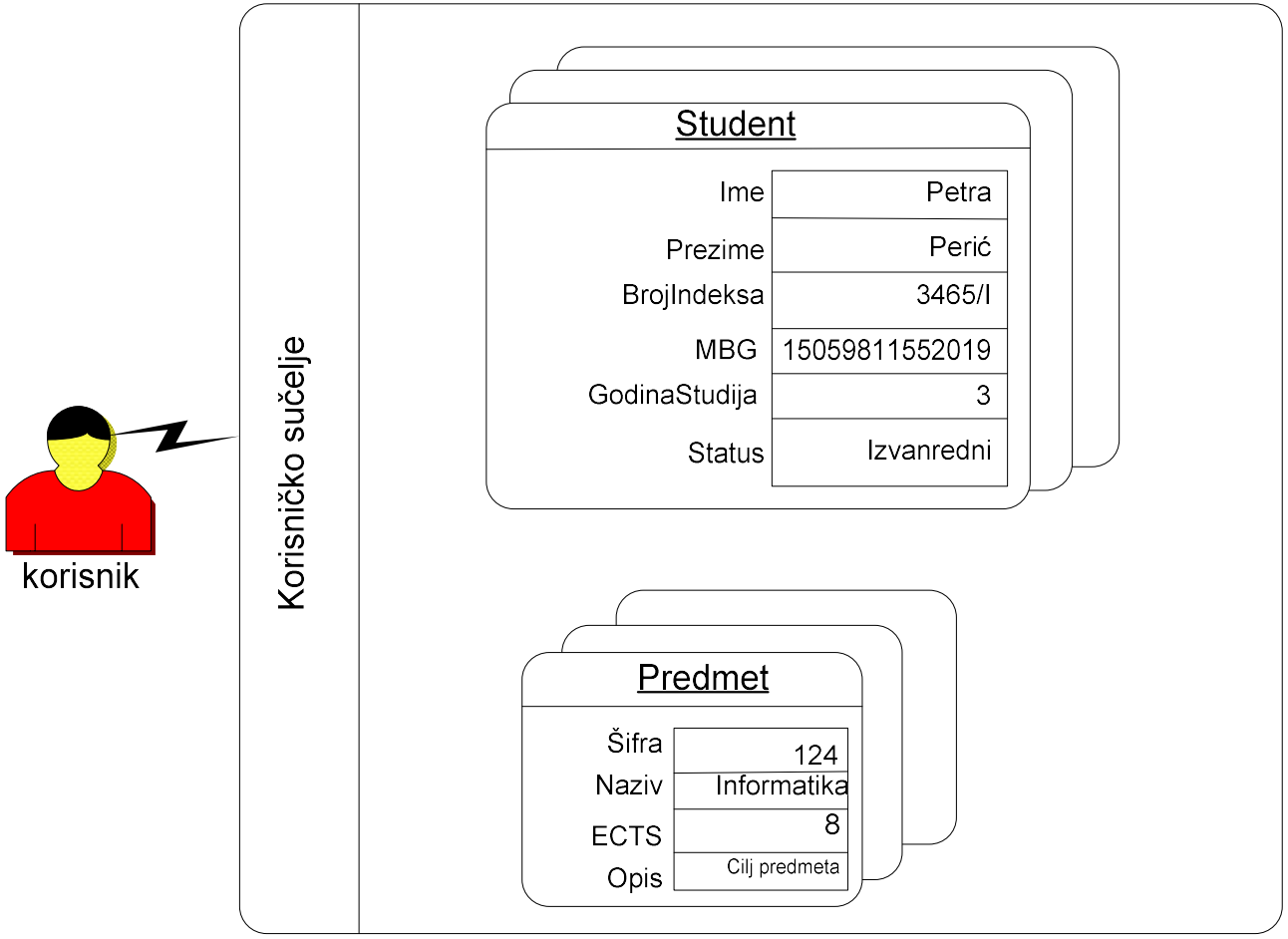
Dva su tipa poruka na koje objekt može odgovoriti:

- Zahtjev za podacima koji se naziva upit (engl.*query*)
- Zahtjev za promjenom stanja koji se naziva naredba (engl. *command*).

Skup upita i naredbi na koje će dani objekt odgovoriti naziva se sposobnost (mogućnost) objekta i određuje prigodom dizajniranja objekta, a u objektnoj paradigmi se implementira pomoću metode.

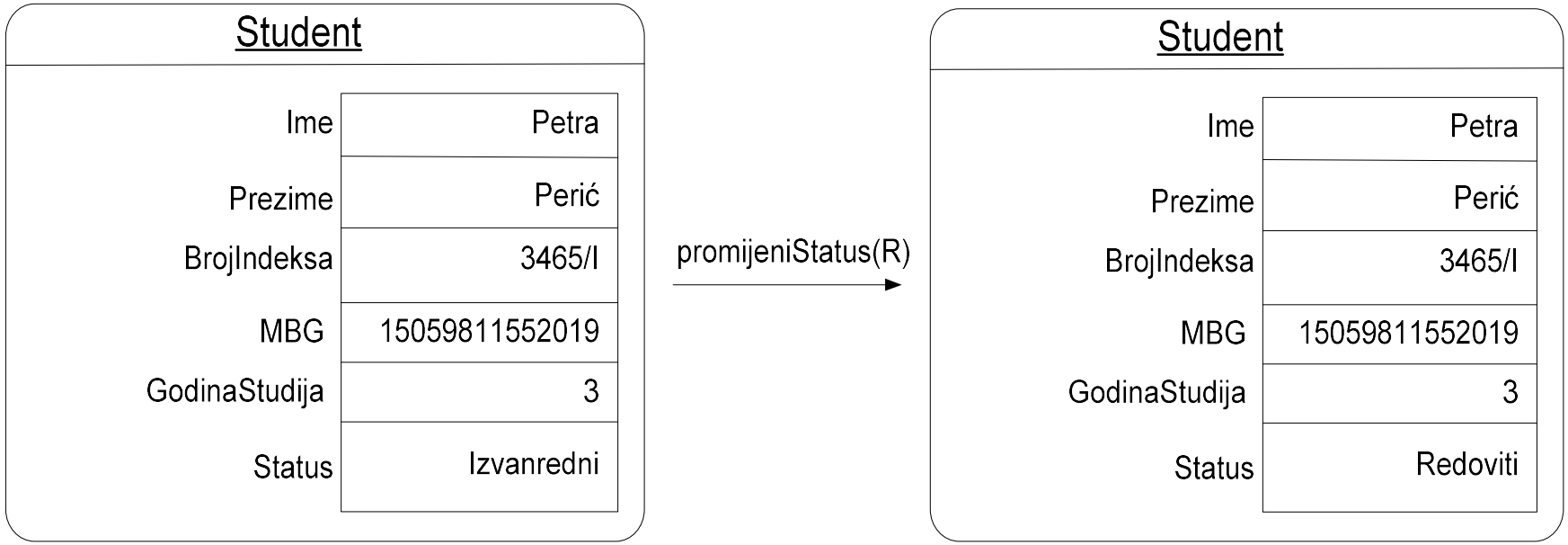
Metoda

Upit:


Korisničko sučelje
Student
Ime
Petra
Prezime
Perić
BrojIndeksa
3465/I
MBG
15059811552019
GodinaStudija
3
Status
Izvanredni
Predmet
Šifra
124
Naziv
Informatika
ECTS
8
Opis
Cilj predmeta
korisnik

Metoda

Promjena stanja objekta:



Metoda

Skup svih sposobnosti (funkcionalnosti) objekta, onako kako se vidi od strane klijenta naziva se *specifikacija*

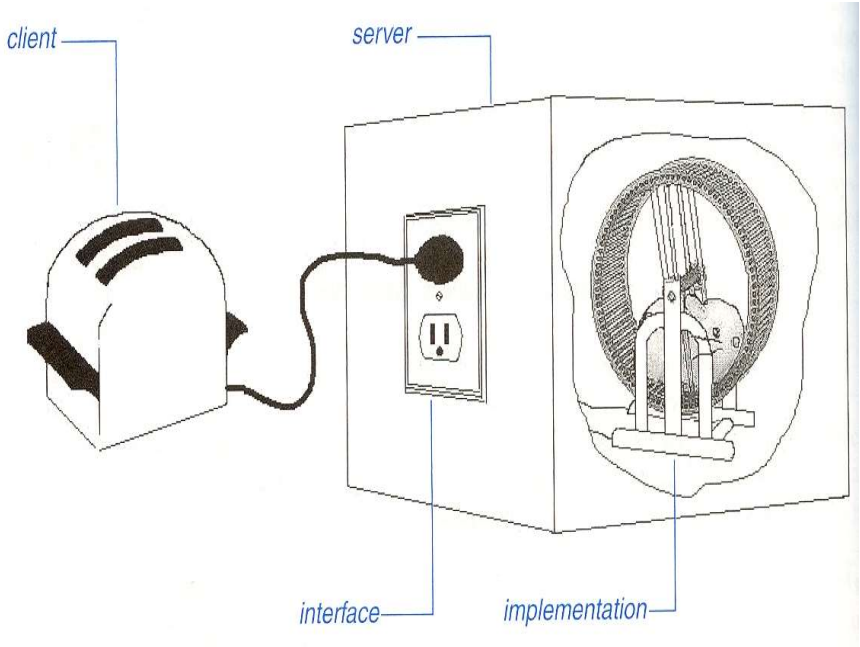
Pojam *implementacija* koristi se za opis interne strukture objekta koja ustvari čini sposobnost tj. funkcionalnost objekta

Funkcionalnost se naziva još i sučelje (engl. *interface*), ali se taj termin ne koristi da se ne bi pomiješao s Java sučeljem.

Metoda

Primjer: standardna utičnica za struju

Specifikacija jamči, ako se neki od električnih uređaja uključi u utičnicu, da će taj uređaj biti snabdjeven strujom odgovarajućeg napona. Na koji način se struja proizvodi, te kako dolazi do utičnice (implementacija) nije nešto što bi bilo koji električni uređaj trebao poznavati, ili o tome voditi računa



Metoda

Sintaksa programskog jezika Java ne dopušta odvajanje specifikacije klase od njene deklaracije.

Klasa se definira pomoću deklariranja ili definiranja klase, a sadrži i specifikacijske i implementacijske mogućnosti

```
public class Student {
    …    ←specifikacija

    public int
GodinaStudija() {
        … ←implementacija
    }
}
```

Metode

Opći oblik deklariranja metode je:

```
modifikator  tip naziv(lista parametara)
pristupa
{
        // tijelo metode
      return izraz
   }
```

Metode

Lista parametara sadrži niz parova sastavljen od tipa podatka i imena parametra, razdvojenih zarezima.

Parametri su varijable koje prihvaćaju vrijednosti argumenata proslijeđenih metodi u trenutku njenog poziva.

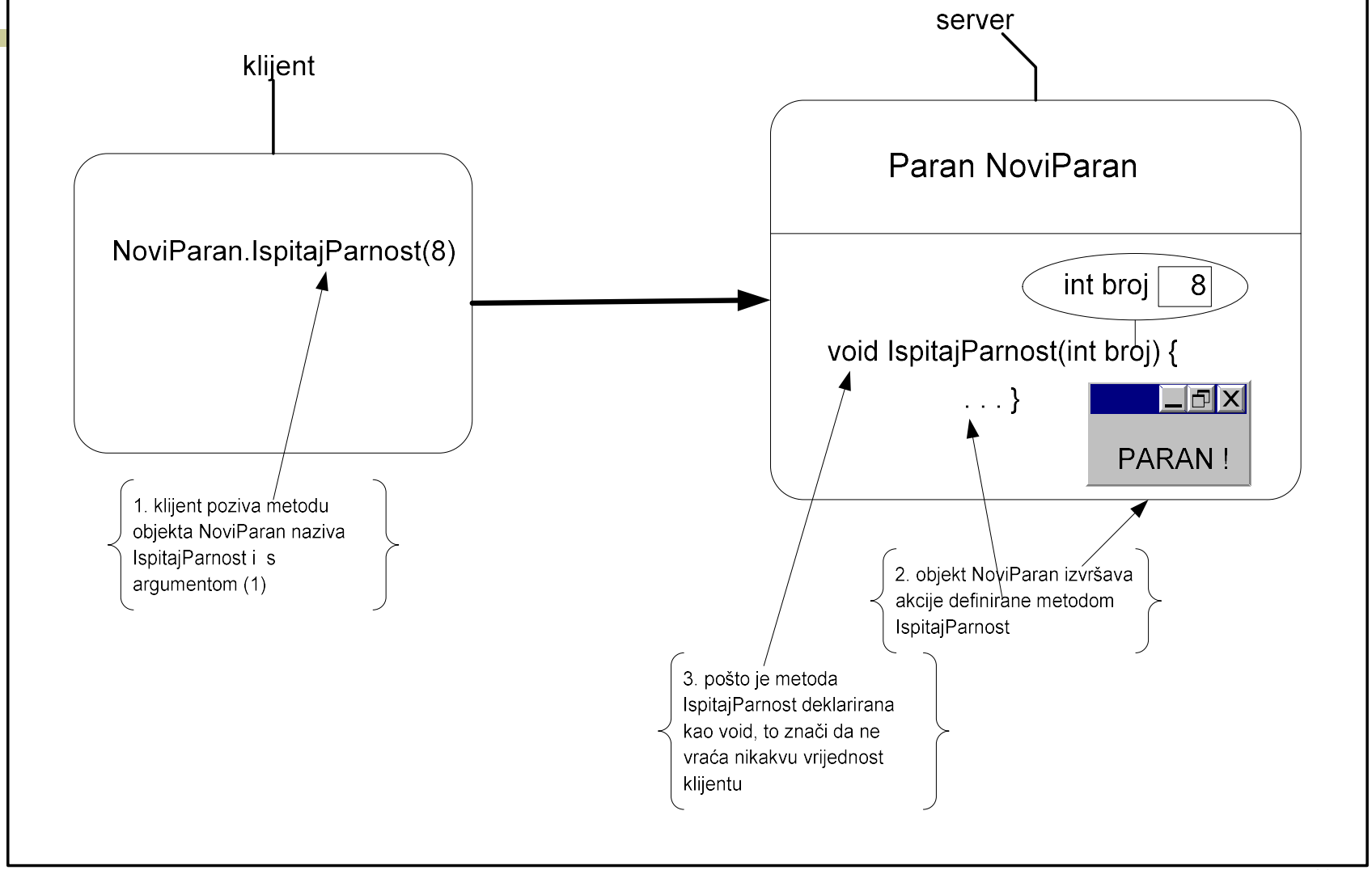
Ako metoda nema parametara, lista parametara je prazna.

Metode

Način rada objekata s metodama opisuje se općenito kao:

NazivObjekta.NazivMetode(argumenti);

Poziva se NazivMetode objekta NazivObjekta, odnosno metoda NazivMetode koja se odnosi na objekt NazivObjekta, s argumentima


server
klijent
Paran NoviParan
NoviParan.IspitajParnost(8)
int broj 8
void IspitajParnost(int broj) {
. . . }
PARAN !
1. klijent poziva metodu objekta NoviParan naziva IspitajParnost i s argumentom (1)
2. objekt NoviParan izvršava akcije definirane metodom IspitajParnost
3. pošto je metoda IspitajParnost deklarirana kao void, to znači da ne vraća nikakvu vrijednost klijentu

Metode

```
class Paran {
        void ispitajParnost (int broj)
{
         System.out.println("Broj je "+broj);
                  if (broj % 2 = = 0)
         System.out.println("PARAN !");
    }
      public static void main (String args[]) {
                  Paran NoviParan = new Paran();
                  NoviParan.ispitajParnost(1);
                  NoviParan.ispitajParnost(8);
                  NoviParan.ispitajParnost(93);
                  NoviParan.ispitajParnost(1432);
 }}
```

Zadatak: doraditi program tako da korisnik izravno unosi broj i onda se provjerava parnost. (Paran.java)

Primjer kreiranja objekta

Program: Student4.java

```
import java.util.Scanner;

public class Student4
{ public static void main(String[] args)
 { Scanner unos = new Scanner(System.in);
   System.out.print("Ime studenta ");
   String imes = unos.next();
   .....
 student student1 = new
 student(imes,prezimes,tips,indekss);
  System.out.println("Student1 "+student1.uzmiime()+"
"+student1.uzmiprezime()+" "+student1.uzmitip()+"
"+student1.uzmiindeks());  }}
```

Primjer kreiranja objekta

Program: Student4.java

```
class student
 { public student(String i, String p, String t, String n)
  { ime = i;
    prezime = p;
    tip = t;
    indeks = n;
  }
    public String uzmiime()
  { return ime; }
  public String uzmiprezime()
  { return prezime; }
  public String uzmitip()
  {return tip; }
  public String uzmiindeks()
  {return indeks; }
  private String ime, prezime, tip, indeks;   }
```

“Static” varijabla

⇒Ako se varijabla definira kao “static” to znači da postoji samo jedna takva za klasu, inače svaki objekt ima svoju kopiju svih varijabli instanci.

```
class djelatnik {
        ....
        private int id;
        private static int nextId = 1;
   }
```

“Static” varijabla

```
Public void setId() {
        id = nextId;
        nextId++;
    }

Mate.setId();
Mate.id = …;
djelatnik.nextId++;
```

Domaća zadaća za 24.04.2018.

Program: Student4.java

Nadograditi s automatskim izračunom šifre za studenta i prikazom sljedeće slobodne šifre.

Rezultat: Student5.java

PITANJA