

# Uvod u objektno programiranje

Metode

# Metode

---

- Motivacija:
  - ponavljanje koda
  - dekompozicija na manje celine
- Osnovni elementi:
  - definicija
  - poziv

# Definicija

---

- Opšta sintaksa:

```
povratni_tip ime_metode(parametri) {  
    ...  
}
```

- Povratni tip je bilo koji tip podatka ili *void* ako metoda ne vraća vrednost.
  - metoda vraća najviše jednu vrednost!
- Parametri (argumenti) se deklarišu na isti način kao i promenljive.
  - ako metoda nema parametara ostave se prazne zagrade.
- Ako metoda vraća vrednost, to se postiže *return* naredbom:
  - `return a;`
  - `return (a);`

# Primer

---

- Mogu pozivati (izvršiti) u bilo kom trenutku u programu, potrebno je samo da se navede ime funkcije i njeni pozivajući parametri (ukoliko postoje).

```
/*  
Pozeljno je iznad funkcije napisati kratak komentar  
kojim se objasjava programska logika i svrha funkcije  
*/  
// ispis Hello World teksta  
static void pozdrav(){  
    System.out.println("Hello World");  
}
```

Primer00.Zad01

```
public static void main(String[] args) {  
    pozdrav();  
}
```

# Parametri i rezultat metoda

---

- Parametri mogu biti:
  - primitivni tipovi
  - reference na objekte
- Rezultat može biti:
  - primitivni tip
  - referenca na objekat
- Metoda vraća vrednost naredbom:  
return vrednost;  
ili  
return (vrednost);

# Primer metode bez parametara

---

```
double vratiSlucajanBroj() {  
    return Math.random() * 100;  
}  
  
... //poziv funkcija u mainu  
System.out.println(vratiSlucajanBroj());  
...
```

# Primer

---

```
//izracunavanje kvadrata hipotenuze pravouglog trougla
//ulazni parametri su duzine kateta a i b
static double vrednostHipotenuzePravouglogTrougla(double a, double b){
    double c = 0;
    c = Math.sqrt(a*a + b*b);
    return c;
}

public static void main(String[] args){

    double vrednost = 0;
    vrednost = vrednostHipotenuzePravouglogTrougla(3, 4);
    System.out.println("Vrednost hipotenuze je:"+vrednost);
}
```

# Primer više return naredbi

---

```
int max(int a, int b) {  
    if (a > b)  
        return a;  
    else  
        return b;  
}  
... //poziv funkcija u mainu  
int m = max(4, 5);  
...
```



# Primer greške

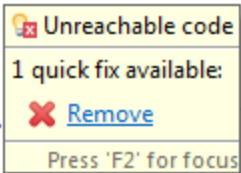
- Ukoliko se tip povratne vrednost iz tela funkcije ne poklopi sa tipom povratne vrednosti navedenim u naslovu funkcije ili ako implicitna konverzija tipova nije moguća, alat Eclipse će prijaviti grešku (konverzija sa šireg na uži tip nije dozvoljena).

```
static int vrednostHipotenuzePravouglogTrougla(double a, double b)
{
    double c = 0;
    c = Math.sqrt(a*a + b*b);
    return c;
}
```

# Primer greške

- Posle napisane naredbe `return` pisanje bilo kakvog java koda nema smisla tj. taj kod se neće nikada izvršiti.

```
// ----- JAVO -----  
static double vrednostHipotenuzePravouglogTrougla(double a){  
    double c = 0;  
    c = Math.sqrt(a*a + a*a);  
  
    //    a = 5;  
    return c;  
    a = 5;  
}  
public void main(String[] args){
```



The image shows a code editor with a Java method `vrednostHipotenuzePravouglogTrougla` that returns a value. After the `return` statement, there is a line `a = 5;` which is unreachable. An IDE tooltip is displayed over this line, indicating the error and offering a 'Remove' quick fix.

# Lokalne promenljive

---

- Sve promenljive deklarisanе unutar metoda su lokalne promenljive.
- Takve promenljive se ne vide u drugim metodama i one se svaki put prave kada program pozove metodu, i uništavaju se kada se metoda završi.
- Lokalne promenljive nemaju predefinisano (*default*) vrednost!

```
4 public static void main(String[] args) {  
5     int a, b = 1, c;  
6     c=a+b;  
7 }
```

# Parametri metoda

---

- Parametri metoda se u nekim programskim jezicima prenose po vrednosti ili po referenci
  - u programskom jeziku Java, prenos je **isključivo** po vrednosti

# Prenos parametara po vrednosti

---

- Prenos parametara po vrednosti:
  - prave se kopije parametara i te kopije se prosleđuju metodi
  - posledica: nije moguće promeniti prosleđenu promenljivu iz metode

# Prenos parametara po vrednosti

```
static void f(int x) {  
    x = 1;  
}
```

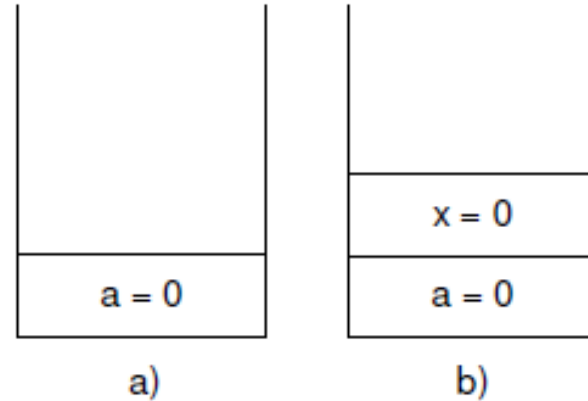
Ne vredi ni da  
promenljivu 'x'  
nazovemo 'a'

```
public static void main(String[] s) {  
    int a = 0;  
    f(a);  
    System.out.println(a);  
}
```

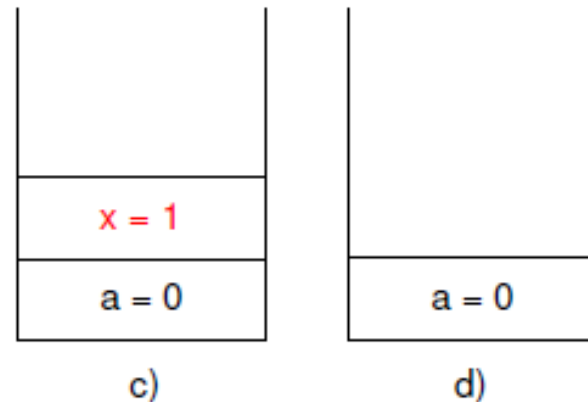
Šta će biti  
odštampano?

# Prenos parametara po vrednosti

- Pri pozivu metode



- Pri povratku iz metode



# Nizovi kao parametri metoda

- U listi parametara metoda ne navode se dimenzije.
- Primer:

```
void f(int a[]) {  
    ...  
    a[3] = 5;  
}
```

**Ovo će promeniti a[3] u pozivajućoj metodi!**

POSLEDICA:

- Elementi niza se mogu promeniti iz metode!
- Pri proseđivanju niza (kao i objekta) ne pravi se novi niz tj. ne pravi se kopija niza!
- Pri proseđivanju niza (kao i objekta) se pravi kopija reference na niz tj. referenca se proseđuje po vrednosti!

- Ako je potrebna veličina niza, ona se može saznati iz atributa length:  
a.length



# Višedimenzionalni nizovi kao parametri metoda

---

- U listi parametara metode se ne navode dimenzije
  - ta informacija se može saznati iz same promenljive
    - `a.length` – broj vrsta
    - `a[0].length` – broj kolona

- Primer:

```
void f(int a[][]) {  
    ...  
    a[3][3] = 5;  
}
```

# Opseg vidljivosti promenljivih

---

- Promenljive deklarisanе unutar metode se “vide” samo u metodi
  - to su lokalne promenljive
- Promenljive deklarisanе izvan metode (na nivou klase) se “vide” i u ostalim metodama te klase
  - to su atributi

# Rekurzivne metode

---

- Rekurzija: metoda poziva samu sebe
- Svaka rekurzivna metoda mora da ima uslov za izlaz iz rekurzije!
- Pozitivno:
  - razumljivije
  - ponekad i jedino moguće (Akermanova funkcija)
- Mana:
  - opterećuje stek
  - brzina

$$A(m, n) = \begin{cases} n + 1 & \text{if } m = 0 \\ A(m - 1, 1) & \text{if } m > 0 \text{ and } n = 0 \\ A(m - 1, A(m, n - 1)) & \text{if } m > 0 \text{ and } n > 0. \end{cases}$$

# Rekurzivne metode

---

//bez rekurzije

```
int fakt(int n) {  
    int i, f=1;  
    for (i = 1; i <= n; i++) {  
        f *= i;  
    }  
    return f;  
}
```

//sa rekurzijom

```
int fakt(int n) {  
    if (n < 2)  
        return 1;  
  
    return n * fakt(n-1);  
}
```