Objektno Orijentisano Programiranje 1

Metode

Funkcije ili 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

```
int saberi(int a, int b) {
  return a + b;
}
...
int x = 5, y = 6;
int c = saberi(x, y);
System.out.println(c);
...
```

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;
}
...
System.out.println(vratiSlucajanBroj());
...
```

Primer više return naredbi

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

Primer složene metode

- Napisati metodu sinus koja izračunava sinus razvojem u Tejlorov red, tačnosti 10^{-6}, po formuli:

  $sinus(x) = x/1! - x^3/3! + x^5/5! - x^7/7! + ...$

```
s = 0; sabirak = x; znak = 1; stepen = 1;
while (Math.abs(sabirak) > 1E-6)
{
  s = s + sabirak;
  znak = znak*-1;
  stepen = stepen + 2;
  sabirak = znak * Math.pow(x,stepen)/fakt(stepen);
}
```

Sinus.java

Lokalne promenljive

- Sve promenljive deklarisane 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 predefinisanu (*default*) vrednost!

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

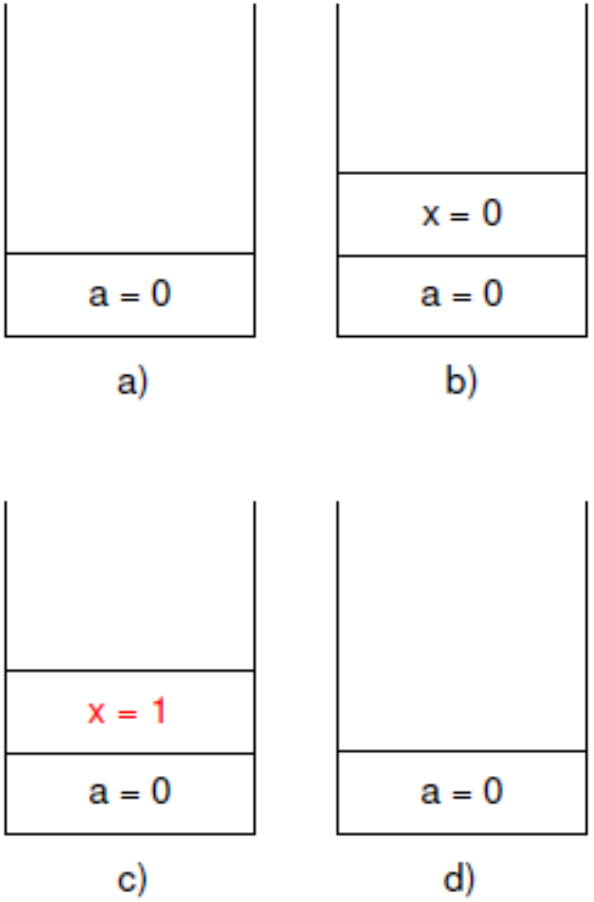
Vrednost.java

```
static void f(int x) {
   x = 1;
}
public static void main(String[] s) {
   int a = 0;
   f(a);
   System.out.println(a);
}
```

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

Šta će biti odštampano?

Prenos parametara po vrednosti



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!
- Nizovi se ne prosleđuju po vrednosti, tj. ne pravi se kopija niza!

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

  a.length

Nizovi.java

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;
}
```

Nizovi.java

Opseg vidljivosti promenljivih

- Promenljive deklarisane unutar metode se “vide” samo u metodi
  - to su lokalne promenljive
- Pomenljive deklarisane izvan metode se “vide” i u ostalim metodama
  - to su atributi

Opseg vidljivosti promenljivih

```
static void f(int i) {
  i = 3;
}

static int a;

static void f2(int i) {
  int b;
  a = 5;
  i = 3;
}

static void f3(int i) {
  int a;
  a = 5;
  i = 3;
}
```

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

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

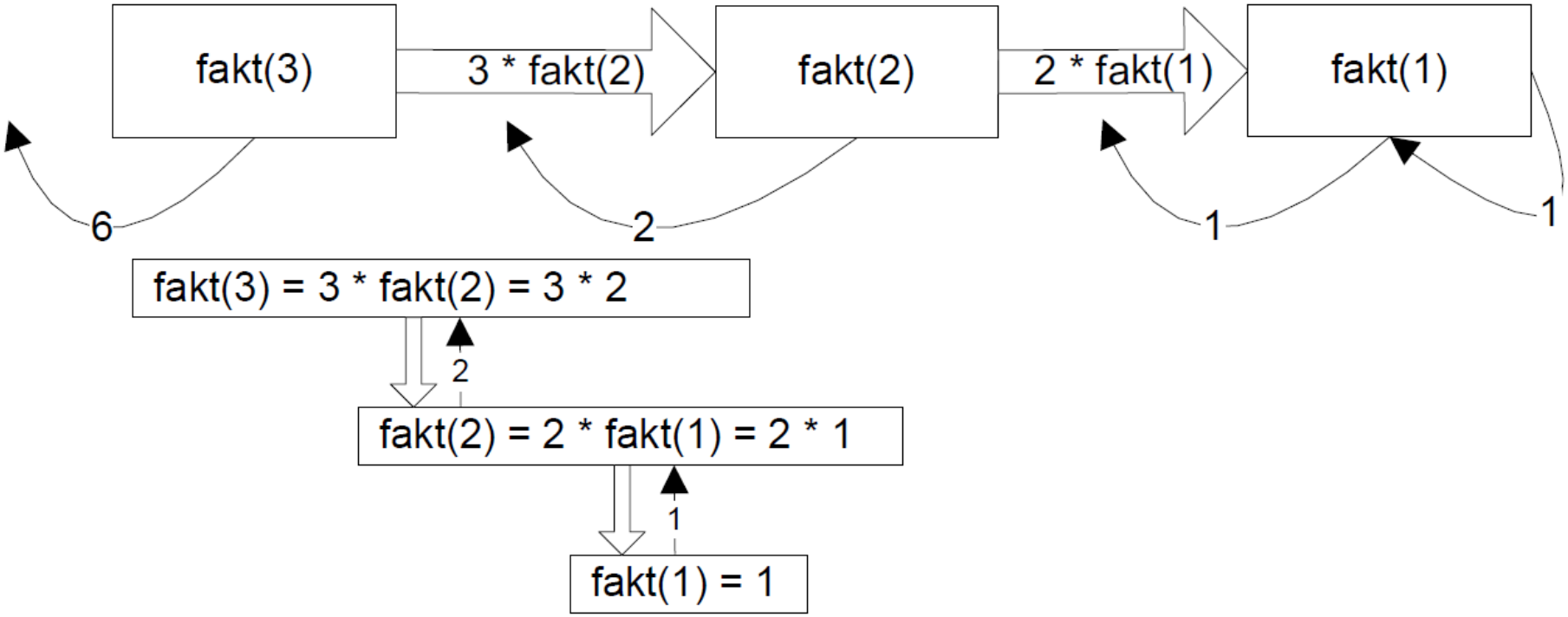
Fakt1.java

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

  return n * fakt(n-1);
}
```

Fakt2.java

Rekurzivne metode



Promenljiv broj parametara

- Ako želimo da metoda ima promenljiv broj parametara, definišemo je na sledeći način:

```
void f(int... params) {...}
```

- Parametrima se pristupa kao elementima niza:

```
int a = params[2];
```