

Polimorfizam i vrste polimorfizma

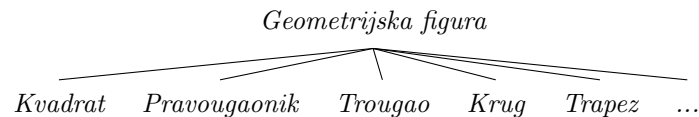
Bogosav Cvetković
mr16213@alas.matf.bg.ac.rs

06. decembar 2018.

1 Uvod

Polimorfizam (eng. *polymorphism*) je sposobnost imanja više formi, koje se različito ponašaju. Kao prirodna pojava u svakodnevnom životu, polimorfizam je lako našao mesto i u računarstvu.

Primer 1.1 U Euklidovoj geometriji površinu možemo dodeliti raznim figurama poput kvadrata, trougla, pravougaonika, trapeza, kruga, kružnog isečka... Ukoliko bismo koristili formalnu definiciju površine, nju bismo računali korišćenjem višestrukih integrala. Taj pristup teorijski radi ali postavlja se pitanje da li računar može, a ako može koliko bi to bilo efikasno i da li je to zapravo ono što smo želeli.



Kada bi svaki oblik imao svoj metod(funkciju) za računanje površine, to bi bio naporan i nelagodan posao za programere.

```
double površina_kvadrat(double a);  
double površina_pravougaonik(double a, double b);  
double površina_trougao(double a, double h);  
double površina_krug(double r);  
...
```

Stoga je poželjnije imati jednu funkciju koja će u zavisnosti od figure i uz pomoć pravih argumenata izračunati površinu na pravi način.

```
double površina(args);  
}
```

2 Vrste polimorfizma

- Statički polimorfizam

- Parametarski polimorfizam
- Ad-hoc polimorfizam
- Dinamički polimorfizam
 - Hijerarhijski polimorfizam
 - Implicitni polimorfizam

Pod statičkim polimorfizmom podrazumevamo da se on ostvaruje u toku prevođenja, dok se dinamički ostvaruje u toku izvršavanja samog programa.

Statički polimorfizam se izvršava brže, ali zahteva dodatnu podršku kompajlera. Statički polimorfizam omogućava veću statičku analizu: prilikom kompajliranja (posebno za optimizaciju), analizu izvornog koda i analizu koju vrše programeri. Dinamički polimorfizam je fleksibilniji, ali sporiji - na primer, dinamički polimorfizam omogućava kodiranje bez eksplicitno deklariranih tipova, a dinamički povezane biblioteke omogućavaju njihovo korišćenje nad objektima bez poznavanja njihovog punog sadržaja.[\[3\]](#)

3 Statički polimorfizam

3.1 Parametarski polimorfizam

Parametarski polimorfizam počiva na upotrebi tipskih promenljivih. Tipske promenljive se koriste za simboličko označavanje tipova vrednosti, promenljivih i izraza. Svaki put kada se upotrebi kod koji je napisan uz upotrebu parametarskog polimorfizma, simbolički tipovi se zamenjuju konkretnim tipovima i program se prevodi u nepolimorfnom obliku. Da bi prevođenje bilo uspešno, neophodno je da u svim delovima tog polimorfnog koda zamenjivanje simboličkih tipova konkretnim tipovima proizvodi ispravan programski kod. Određivanje konkretnih tipova se može izvršiti eksplicitno, navođenjem tipa ili implicitno na osnovu mehanizma samog programskog jezika.

Generičko programiranje je zasnovano na parametarskom polimorfizmu

Primer 3.1 U čvoru binarnog stabla možemo čuvati podatke različitih tipova. Mukotrpno bi bilo kada bismo za svaki tip pravili različito stablo.[\[2\]](#)

```
public class Tree<T>{
    private T value;
    private Tree<T> left;
    private Tree<T> right;

    public void replaceAll(T value){
        this.value = value;
        if(left != null)
            left.replaceAll(value);
        if(right != null)
            right.replaceAll(value);
    }
}
```

3.2 Ad-hoc polimorfizam

Ad-hoc polimorfizam se može posmatrati kao specijalan slučaj parametarskog polimorfizma. Ova vrsta polimorfizma nam dopušta višeznačnost imena funkcija (eng. *overloading*). Ova osobina se često koristi kod konstruktora u OOP.

Primer 3.2 *Umesto posebne funkcije za svaki tip, ad-hoc polimorfizam nam dozvoljava da definišemo funkciju saberi koja će se u zavisnosti od poziva adekvatno tumačiti.*

```
int saberi(int x, int y);
int saberi(int x, int y, int z);
int saberi(int x, int y, int z, int w);
```

umesto

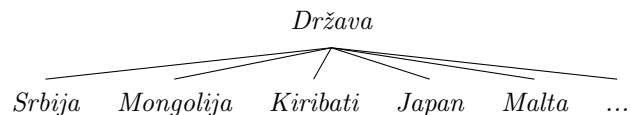
```
int saberi2(int x, int y);
int saberi3(int x, int y, int z);
int saberi4(int x, int y, int z, int w);
```

4 Dinamički polimorfizam

4.1 Hijerarhijski polimorfizam

Konceptualno zasnovan na hijerarhijskom odnosu između klasa u objektno-orijentisanom programiranju. Ukoliko natklasa (*superclass*) sadrži neki metod on se prenosi i na sve potklase lasa u objektno-orijentisanom programiranju. Ukoliko natklasa (*subclass*).

Primer 4.1 *Posmatrajmo klase koje su uređene na sledeći način*



Svaka od ovih država ima svoj glavni grad, stoga definišemo metod koji će vraćati ime odgovarajućeg grada

```
abstract class Drzava {
    abstract String glavni_grad();
}

static class Srbija extends Drzava {
    String glavni_grad() {
        return "Beograd";
    }
}

static class Mongolija extends Drzava {
    String glavni_grad() {
        return "Ulan_Bator";
    }
}
```

```

    }
}

static class Kenija extends Drzava {
    String glavni_grad() {
        return "Najrobi";
    }
}

void ispisiMesto(Drzava a) {
    System.out.println(a.glavni_grad());
}

class Main {
    public static void main(String[] args) {
        ispisiMesto(new Srbija());
        ispisiMesto(new Kenija());
        ispisiMesto(new Mongolija());
    }
}

```

4.2 Implicitni polimorfizam

Implicitni polimorfizam je najzastupljeniji u funkcionalnoj paradigmi. Može se smatrati uopštenjem parametarskog polimorfizma. Naziv je dobio zbog činjenice da nije potrebno navođenje tipova (možemo ih izostaviti) .[1]

Primer 4.2 U funkcionalnim jezicima možemo definisati funkciju bez da joj prosledimo tipove. Primer funkcije identiteta.

```

let id = fun(a) a
id(3)
id(true)

```

Primer 4.3 Makroi u programskom jeziku C su takođe neki vid implicitnog polimorfizma

```

#define kub(x) (x)*(x)*(x)

```

Literatura

- [1] dr Milena Vujošević Jančić. Materijali i slajdovi sa predavanja. http://www.programskijezici.matf.bg.ac.rs/DizajnProgramskihJezika.html#2_tab, 2018. Online; pristupljeno 06.12.2018.
- [2] Mike Mol. Rosseta Code - Polymorphism. https://rosettacode.org/wiki/Parametric_polymorphism. Online; pristupljeno 06.12.2018.

- [3] Wikipedia. Polimorfizam u računarstvu. [https://sr.wikipedia.org/wiki/%D0%9F%D0%BE%D0%BB%D0%B8%D0%BC%D0%BE%D1%80%D1%84%D0%B8%D0%B7%D0%B0%D0%BC_\(%D1%80%D0%B0%D1%87%D1%83%D0%BD%D0%B0%D1%80%D1%81%D1%82%D0%B2%D0%BE\)](https://sr.wikipedia.org/wiki/%D0%9F%D0%BE%D0%BB%D0%B8%D0%BC%D0%BE%D1%80%D1%84%D0%B8%D0%B7%D0%B0%D0%BC_(%D1%80%D0%B0%D1%87%D1%83%D0%BD%D0%B0%D1%80%D1%81%D1%82%D0%B2%D0%BE)). Online; pristupljeno 06.12.2018.