

# Dokumentacija programskog jezika ***Plang***

Ideja jezika je da sve rezervisane reči počinju na slovo p. Realizovan je na sledeći način:

<b>Skelet programa i identifikatori.....</b>	<b>1</b>
Identifikator fajla.....	1
Sekcija promenljive.....	2
Tipovi podataka.....	3
Sekcija pocni.....	3
<b>Komunikacija sa korisnikom.....</b>	<b>4</b>
Ispisivanje vrednosti.....	4
Učitavanje vrednosti.....	4
<b>Aritmetičke operacije.....</b>	<b>5</b>
<b>Relacione i pojmovne operacije.....</b>	<b>5</b>
Relacione operacije.....	5
Pojmovne operacije.....	6
// da li nam trebaju ternarni i negacija.....	6
<b>Grananje.....</b>	<b>7</b>
<b>Ponovi petlja.....</b>	<b>7</b>
Prekini i produzi.....	8
<b>Procedure.....</b>	<b>9</b>
<b>Postoji.....</b>	<b>10</b>
<b>Popis.....</b>	<b>11</b>

# Skelet programa i identifikatori

## Identifikator fajla

Na početku je neophodno imenovati program naredbom **program**. Na kraju svake naredbe se mora dodati tačka zarez (;). Imena **programa**, **promenljivih**, **procedura** i **popisa** moraju da počinju slovom i mogu da sadrže samo slova, brojeve i donju crtu (\_).

```
program personalni_program;
```

## Sekcija promenljive

Sve promenljive koje koristimo u programu moramo deklarirati u **promenljive** bloku. Neophodno je koristiti reč **postavi** pri deklarisanju promenljive.

```
program personalni_program;  
  
promenljive {  
    postavi personalna_promenljiva;  
}
```

## Tipovi podataka

Prethodni kod se neće kompajlovati. Promenljivoj **personalna\_promenljiva** je neophodno dodeliti **tip podatka**. Mogući tipovi podataka su **podatak**, **plutajuci**, **pismo**, **poruka** i **pojam**. Tokom deklaracije moguće je dodeliti i vrednost koristeći **podesi**. Nizovi se oznacavaju uglastim zagradama ( [ ] ).

```
program personalni_program;

promenljive {
    postavi podatak personalni_podatak podesi 112;
    postavi plutajuci personalni_plutajuci podesi 112.0;
    postavi pismo personalno_pismo podesi 'p';
    postavi poruka personalna_poruka podesi "plang";
    postavi pojam personalni_pojam podesi pozitivno;
    postavi podatak[2] personalni_podatak2 podesi {112,0};
}
personalni_podatak2[1] podesi 80;
```

## Sekcija pocni

Ni prethodni kod se ne može kompajlovati. Sve naredbe moraju stajati u **pocni** bloku.

```
program personalni_program;

promenljive {
    postavi podatak personalni_podatak;
}
pocni {
    personalni_podatak podesi 112;
}
```

Ovaj kod bi se pokrenuo, ali se ništa ne bi dogodilo.

# Komunikacija sa korisnikom

## Ispisivanje vrednosti

Ako želimo da korisnik vidi nešto što smo uradili možemo da koristimo **pisi()**.

```
program personalni_program;  
  
promenljive {  
    postavi podatak personalni_podatak podesi 112;  
}  
pocni {  
    pisi(personalni_podatak);  
    pisi(80);  
    pisi('p');  
}
```

## Učitavanje vrednosti

Ako želimo da korisnik unese neke vrednosti koje treba da obradimo koristimo **procitaj()**.

```
program personalni_program;  
  
promenljive {  
    postavi podatak personalni_podatak;  
}  
pocni {  
    procitaj(personalni_podatak);  
    pisi(personalni_podatak);  
}
```

## Aritmetičke operacije

Do sada su nam mogućnosti bile ograničene na unos i ispis podataka. Sada možemo i da ih menjamo uvođenjem **aritmetičkih operacija**. Postojeće operacije su **plus**, **potkresi**, **puta**, **podeljeno**, **povrat** i **podigni**.

```
program personalni_program;

promenljive {
    postavi podatak personalni_podatak podesi 112;
}
pocni {
    pisi(personalni_podatak plus 1);      // ispisuje 113
    pisi(personalni_podatak potkresi 1);  // ispisuje 111
    pisi(personalni_podatak puta 2);      // ispisuje 224
    pisi(personalni_podatak podeljeno 2); // ispisuje 56
    pisi(personalni_podatak povrat 5);    // ispisuje 2
    pisi(personalni_podatak podigni 2);   // ispisuje 12544
    personalni_podatak podesi 80 plus 112;
}
```

# Relacione i pojmovne operacije

Da bismo napravili nešto korisnije od kalkulatora, moramo uvesti grananje (o kojem ćemo pričati kasnije), koje zahteva relacije i pojmovne operacije.

## Relacione operacije

Relacione operacije porede dva **podatka** ili **pojma**. Rezultat je uvek tipa **pojam** (**pozitivno** ili **pogresno**).

```
program personalni_program;

promenljive {
    postavi pojam personalni_pojam podesi pozitivno;
}
pocni {
    pisi(1<2);                // pozitivno
    pisi(personalni_pojam=pogresno); // pogresno
    pisi(1 plus 2 ≥ 3);        // pozitivno
    pisi(personalni_pojam=1);   // nije dozvoljeno
}
```

## Pojmovne operacije

Pojmovne operacije porede dva **pojma**. Rezultat je uvek tipa **pojam** (**pozitivno** ili **pogresno**).

```
program personalni_program;

promenljive {
    postavi pojam personalni_pojam podesi pozitivno;
}
pocni {
    pisi(1<2 && !pozitivno); // pogresno
    pisi(personalni_pojam || pogresno); // pozitivno
}
```

## Grananje

Grananjem možemo da izvršavamo različite stvari u zavisnosti od neke vrednosti. Za ovo se koristi **proveri**.

```
program personalni_program;

promenljive {
    postavi podatak personalni_podatak;
}
pocni {
    procitaj(personalni_podatak);
    prover_i(personalni_podatak povrat 2 == 0) pa
    {
        pisi("parno");
    }
    pak{ pisi("neparno"); }
}
```

Ukoliko u zagradi koja stoji uz prover\_i stoji pozitivno onda ulazimo u pa blok, u suprotnom ulazimo u pak blok.

## Ponovi petlja

**Ponovi** petlja služi da se neki kod izvršava više puta.

```
program personalni_program;

promenljive {
    postavi podatak p podesi 1;
}
pocni {
    ponovi(p; p ≤ 5; p podesi p plus 1;)
    {
        pisi(p);
        pisi(' ');
    }
} //ispisuje "1 2 3 4 5 "
```

Ponovi petlja se izvršava sve dok je promenljiva `p` manja ili jednaka broju 5. Podesi `p plus 1`; znači da se promenljiva `p` povećava za 1 svaki put kada se kod u telu petlje izvrši.

## Prekini i produzi

Naredba **prekini** odmah izlazi iz tela **ponovi** petlje.

```
program personalni_program;

promenljive {
    postavi podatak p podesi 1;
}
pocni {
    ponovi(p; p ≤ 5; p podesi p plus 1;)
    {
        proveriti(p = 4) pa prekini
        pak
        {
            pisi(p);
            pisi(' ');
        }
    }
} //ispisuje "1 2 3 "
```

U četvrtoj iteraciji nailazimo na **prekini** komandu i tu izlazimo iz petlje. Treba primetiti da se ne započinje peta iteracija **ponovi** petlje.

Naredba produzi odmah prelazi na sledeću iteraciju petlje.

```
program personalni_program;

promenljive {
    postavi podatak p podesi 1;
}
pocni {
    ponovi(p; p ≤ 5; p podesi p plus 1;)
    {
        proveriti(p = 4) pa produzi
        pak
        {
            pisi(p);
            pisi(' ');
        }
    }
} //ispisuje "1 2 3 5 "
```

## Procedure

Često se dešava da treba da isti kod izvršimo na više mesta. Najbolje rešenje je da taj deo izmestimo u **proceduru**.

```
program personalni_program;

procedura pisi_pismo_p() { pisi('p'); }
promenljive { }
pocni {
    pisi_pismo_p(); // ispisuje slovo p
}
```

**Procedure** mogu da imaju svoje promenljive i da ih koriste u svom kodu.

```
program personalni_program;

procedura pisi_proizvod_podataka(podatak p; podatak l)
promenljive { postavi podatak proizvod; }
{
    proizvod podesi p puta l;
    pisi(proizvod);
}
promenljive { }
pocni {
    pisi_proizvod_podataka(3;5); // ispisuje 15
}
```

Ako ipak želimo da sačuvmo vrednost koju **procedura** izračuna, moramo da dodelimo tip podatka **proceduri**. Vraćanjem vrednosti prekidamo **proceduru**. Za to nam treba **posalji**.

```
program personalni_program;

procedura podatak proizvod_podataka(podatak p; podatak l)
{
    poslji p puta l;
}
promenljive { postavi podatak p; }
pocni {
    p podesi proizvod_podataka(3;5); // p = 15
    pisi(proizvod_podataka(p;p)); // ispisuje 225
}
```

## Postoji

Ako želimo da napravimo novi tip podataka koristimo **postoji**.

```
program personalni_program;

postoji persona {
    poruka ime;
    poruke prezime;
    podatak godine;
}
promenljive { postavi persona p; }
pocni {
    p podesi { "Pavle";"Petrovic";80 };
    pisi(p.ime); // ispisuje pavle
}
```

## Popis

Ako želimo da vrednost neke promenljive bude iz predefinisanoog skupa koristimo **popis**.

```
program personalni_program;

popis ptice {
    patka;
    papagaj;
    pingvin;
}
promenljive { postavi poruka p; }
pocni {
    p podesi ptice.patka;
    pisi(p); // ispisuje patka
}
```