# Ulazno-izlazne naredbe

Razmenu podataka između programa i standardnog ulaza (tastatura, fajl na disku, ...) i izlaza (ekran, štampač, disk, ...) možemo izvršiti korišćenjem ulazno-izlaznih naredbi. Ulazne naredbe programskog jezika Pascal su **read** i **readln**, dok su izlazne naredbe **write** i **writeln**.

## Ulazne naredbe

Za rad većine programa potrebno je obezbediti neophodne ulazne podatke nad kojima će program obavljati određene aktivnosti. Iako je najčešći slučaj da se ulazni podaci definišu na početku rada programa, nisu retki slučajevi kada program tokom čitavog svog rada zahteva unos određenih podataka, koji su mu neophodni za nastavak.

Kada u programu deklarišemo određene promenljive, time smo samo obezbedili memorijsku lokaciju za smeštanje njihovih vrednosti. Kod većine kompajlera deklarisana promenljiva dobija vrednost koja je pre toga bila zapisana na toj lokaciji i koja najčešće nema nikakvog smisla za dati program. Sa druge strane, neki kompajleri prilikom deklaracije automatski postavljaju vrednost promenljive na neku inicijalnu vrednost (na pr. 0 za celobrojnu promenljivu), ali se na to nikada ne treba oslanjati, jer tako napisan program neće funkcionisati ukoliko pod svim kompajlerima. Iz tog razloga, pre korišćenja sve promenljive moraju dobiti inicijalnu vrednost u samom programu ili se njihove vrednosti moraju učitati sa nekog ulaznog uređaja.

Naredbe za učitavanje vrednosti sa standardnog ulaza u programskom jeziku Pascal su **read** i **readln**. Ove dve naredbe preuzimaju vrednosti sa standardnog ulaza i dodeljuju ih odgovarajućim promenljivama.

**Sintaksa**

```
read(X1, X2, ..., Xn);
readln(X1, X2, ..., Xn);
```

Ovako napisane naredbe promenljivama *X1*, *X2*, ..., *Xn* dodeljuju *n* vrednosti koje su unete sa tastature ili nekog drugog standardnog ulaza. Broj, tip i redosled promenljivih moraju da odgovaraju broju, tipu i redosledu podataka koji se unose. Na primer, ukoliko promenljivama *i*, *j*, *x* i *k*, gde su *i*, *j* i *k* celi, a *x* realan broj, treba dodeliti vrednosti 41, 7, 19.6 i 12 sa tastature, onda bi naredba izgledala ovako:

```
read(i,j,x,k);
```

a na tastaturi bismo uneli:

```
41 7 19.6 12↵
```

pri čemu znak ↵ označava pritisnut taster ENTER na tastaturi, odnosno kraj reda u ulaznoj datoteci (fajlu). Nakon unosa podataka promenljive imaju sledeće vrednosti:

| Promenljiva | Vrednost |
|---|---|
| i | 41 |
| j | 7 |
| x | 19.6 |
| k | 12 |

U slučaju upotrebe naredbe **read**, za razdvajanje brojeva prilikom unosa mogu se koristiti jedan ili više praznih mesta (taster SPACE), znak za tabulator → (taster TAB) ili znak za kraj reda ↵ (taster ENTER). Tako, na primer, ova četiri broja možemo uneti i na sledeće načine:

```
41↵
7↵
19.6↵
12↵
```

ili

```
41      7→19.6→12↵
```

ili pak

```
41→7↵
19.6      12↵
```

Napomenimo i to da ni jedan od ovih znakova nije vidljiv na ekranu, ali ih ovde prikazujemo kao oznaku da je potrebno pritisnuti odgovarajući taster.

Ukoliko se vrši učitavanje podataka tipa **char**, naredba **read** prazna mesta ne tretira kao separatore, već kao ravnopravne znakove. Tako, na primer, ako učitavamo realnu promenljivu *a* i znakovne promenljive *c1* i *c2* korišćenjem komande

```
read(a,c1,c2);
```

ulazna linija

```
2.54E+1 Pera
```

će dati sledeće vrednosti promenljivama:

| Promenljiva | Vrednost |
|---|---|
| a | 25.4 |
| c1 | ' ' |
| c2 | 'P' |

Promenljive tipa **string** se učitavaju tako što naredba **read** očitava sve karaktere do kraja reda. Ukoliko je maksimalna dužina navedena u deklaraciji stringa manja od dužine ulazne linije, onda se učitavanje vrši samo do maksimalne dužine stringa. Tako, ukoliko su promenljive deklarisane na sledeći način:

```
var  recenica1:string;
     recenica2:string[8];
     recenica3:string[30];
```

onda će komande

```
readln(recenica1);
readln(recenica2);
readln(recenica3);
```

u slučaju sledećeg unosa:

```
Mace jede jecam
Ana voli Milovana
Udovica baci vodu
```

kao rezultat dati:

| Promenljiva | Vrednost |
|---|---|
| recenica1 | 'Mace jede jecam' |
| recenica2 | 'Ana voli' |
| recenica3 | 'Udovica baci vodu' |

Podatke logičkog tipa nije moguće učitavati korišćenjem naredbe **read**.

Naredba **readln** učitava vrednosti i dodeljuje ih navedenim promenljivama na sličan način kao i naredba **read**, sa tom razlikom što komanda **readln** nakon učitavanja naznačenih promenljivih prelazi u sledeći red, bez obzira da li su u nastavku navedeni još neki podaci. Na primer, ukoliko se izvršavaju naredbe:

```
readln(i,j);
readln(x,y);
```

a na tastaturi se unese:

```
30   10   8.5   39↵
16.9   4.4   12   3.8↵
```

promenljive će dobiti sledeće vrednosti:

| Promenljiva | Vrednost |
|---|---|
| i | 30 |
| j | 10 |
| x | 16.9 |
| y | 4.4 |

## Izlazne naredbe

Nakon obrade podataka u programu, potrebno je rezultate prikazati korisniku ili ih zabeležiti na nekom medijumu. Da bi se to postiglo koriste se naredbe programskog jezika Pascal **write** i **writeln**, koje vrše ispisivanje navedenih podataka na standardnom izlazu (ekran, štampač, datoteka na disku, ...). Pored ispisa navedenih podataka, korišćenjem ovih naredbi moguće je definisati i format u kome se ti podaci ispisuju, kako bi prikaz bio čitljiviji za korisnika.

Naredbe **write** i **writeln** ispisuju na standardnom izlazu vrednosti koje su navedene unutar zagrada.

**Sintaksa**

```
write(X1,X2, ..., Xn);
writeln(X1,X2, ..., Xn);
```

Na primer, naredba:

```
write(1,2,3,4,5);
```

prikazuje na ekranu

```
1     2     3     4     5
```

Isti efekat možemo postići i ukoliko pozovemo nekoliko naredbi **write** uzastopno. Na primer:

```
write(1,2);
write(3);
write(4,5);
```

Naredba **writeln** funkcioniše na sličan način kao i naredba **write**, sa tom razlikom što naredba **writeln** nakon ispisa navedenih podataka prelazi u sledeći red. Ukoliko bismo prethodni primer napisali koristeći naredbu **writeln**:

```
writeln(1,2);
writeln(3);
writeln;
writeln(4,5);
```

onda bismo kao rezultat dobili sledeći prikaz na ekranu:

```
1     2
3

4     5
```

Kao što se vidi iz prethodnog primera, za ispisivanje praznog reda može se koristiti komanda **writeln** bez argumenata.

Prilikom ispisivanja podataka na ekranu ili bilo kom drugom izlaznom uređaju, treba voditi računa o tome da ti podaci budu napisani pregledno kako bi bili lako čitljivi za korisnika. Naredbe **write** i **writeln** omogućavaju ispisivanje podataka u određenom formatu i to:

- definisanje širine polja u kome se ispisuje podatak
- definisanje broja decimalnih cifara pri ispisivanju realnih brojeva

Definisanje širine polja u kome se ispisuje podatak se vrši tako što se iza podatka unutar naredbi **write** ili **writeln** navodi znak **dve tačke**, a zatim broj karaktera koji će zauzimati taj podatak na ekranu. Ukoliko

podatak koji se ispisuje zauzima manje mesta od navedene širine polja, onda se suvišna polja popunjavaju prazninama ispred podatka. Ako je navedena širina polja nedovoljna za ispis podatka, onda se polje proširuje za potreban broj pozicija.

**Primer**

```
program Format;
   const
      i=428;
      a=-8475.542;
      b=56.45E+2;
      p=true;
      c='a';
begin
   writeln(i:8);
   writeln(a:10:2);
   writeln(b:11);
   writeln(p:5,c:5);
end.
```

Nakon izvršenja ovog programa biće ispisan sledeći rezultat na ekranu:

```
|   |   |   |   |   | 4 | 2 | 8 |

|   |   | - | 8 | 4 | 7 | 5 | . | 5 | 4 |

|   |   | 5 | . | 6 | 4 | 5 | E | + | 0 | 3 |

|   | t | r | u | e |   |   |   |   | a |
```