Ulaz i izlaz podataka

Ulaz i izlaz podataka

- Nakon odslušanog bit ćete u stanju:
  - navesti sintaksu naredbi za unos/ispis
    - znakova
    - znakovnih nizova
    - cijelih brojeva
    - realnih brojeva jednostruke i dvostruke preciznosti
  - primijeniti navedene naredbe.

Funkcije `getchar` i `putchar`

```
int getchar(void);
int putchar(int);
```

- Funkcija `getchar` čita jedan znak sa standardnog ulaza. Sintaksa poziva je c_var = `getchar();`
- Kada funkcija naiđe na kraj ulaznih podataka vraća vrijednost `EOF` (**E**nd **O**f **F**ile).
- Obje funkcije kao i simbolička konstanta `EOF` definirane su u `stdio.h`.

Funkcije `getchar` i `putchar` (2)

- Funkcija `putchar` šalje jedan znak na standardni izlaz. Sintaksa poziva je `putchar(`c_var`);`
- Funkcija uzima jedan argument (znak koji treba ispisati) i vraća cjelobrojnu vrijednost ili `EOF` ukoliko ispis znaka nije uspio.

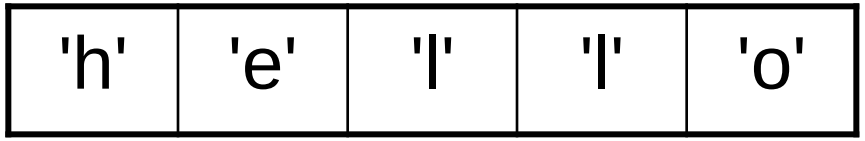
Funkcije putchar i getchar (3)

Primjer:

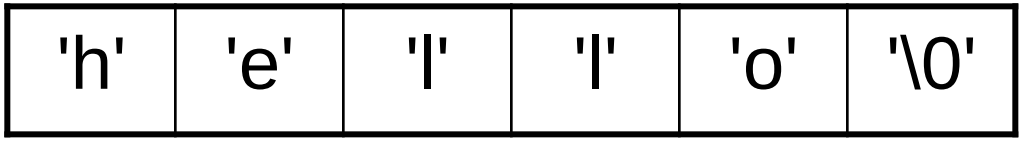
```
#include <stdio.h>

int main (){
    int c;

    while ((c = getchar( )) != EOF)
        putchar(c);
    return 0;
}
```

Znakovni nizovi (stringovi)

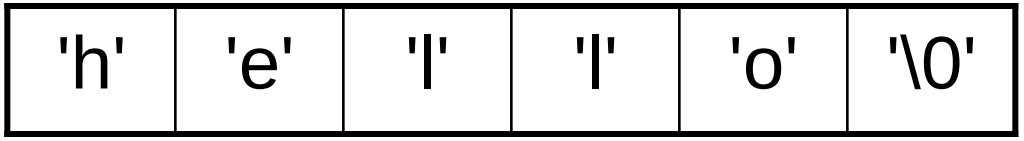
`char` niz_znakova[5] = {'h', 'e', 'l', 'l', 'o'};



`char` string[6] = {'h', 'e', 'l', 'l', 'o', '\0'};



`char` string[ ] = "hello";



Funkcije `gets` i `puts`

```
char *gets(char *s);
int puts(const char *s);
```

- Funkcija `gets` učitava znakove sa standardnog ulaza.
- Kada naiđe na kraj linije zamjenjuje ga s nul znakom '\0' i vraća pokazivač na `char` koji pokazuje na učitani znakovni niz.
- Funkcija vraća `NULL` ukoliko se pojavila greška.

Funkcije `gets` i `puts` (2)

- Funkcija `puts` uzima kao argument znakovni niz koji će biti ispisan na standardnom izlazu.
- Funkcija vraća nenegativni cijeli broj ako je ispis uspio te `EOF` ako ispis nije uspio.
- Prije ispisa: '\0' → '\n'
- Obje funkcije kao i simbolička konstanta `NULL` definirane su u `stdio.h`.

Funkcije `gets` i `puts` (3)

Primjer:

```
#include <stdio.h>
#include <string.h>

main( ){
    char niz[80];

    gets(niz);
    puts(niz);
    printf("%d\n", strlen(niz));
    printf("%c\n", niz[4]);
}
```

Funkcija `scanf`

```
scanf(kontrolni_niz,arg_1,arg_2,…,arg_n);
```

- `kontrolni_niz` sadrži informacije o vrijednostima koje se učitavaju u argumente.
- `kontrolni_niz` je konstantni znakovni niz koji se sastoji od grupa znakova; svaka grupa pridružena je jednom argumentu.
- Grupa započinje znakom postotka (%) kojeg slijedi znak konverzije (npr. %d, %c itd.).

Najčešće korišteni znakovi konverzije

znak konverzije	tip podatka koji se učitava
d	decimalni cijeli broj (`int`)
i	dec, heks. ili okt. cijeli broj (`int`)
h	kratki cijeli broj (`short`)
u	cijeli broj bez predznaka (`unsigned`)
o	oktalni cijeli broj (`int`)
x	heksadecimalni cijeli broj (`int`)
e, f, g	broj s pokretnim zarezom (`float`)
c	jedan znak (`char`)
s	string (`char` *)
p	pokazivač (`void` *)

Unos cijelih brojeva

- Unutar kontrolnog niza grupe znakova mogu (ali ne moraju) biti odvojene bjelinama.
- Argumenti funkcije `scanf` mogu biti samo pokazivači na varijable. Ukoliko podatak treba učitati u neku varijablu, onda `scanf` uzima kao argument adresu te varijable.

  Primjer:

```
int x, y, z, n;
……
n = scanf("%i %i %i",&x, &y, &z); /* ulaz:  13   015      0xd */
```

Unos realnih brojeva

- Znakovi konverzije e, f, g služe za učitavanje varijable tipa `float`. Za učitavanje varijable tipa `double` treba koristiti prefiks l (le, lf, ili lg).

  Primjer:

```
float x;
double y;
.....
scanf("%f %lf", &x, &y);
```

Unos pojedinačnih znakova

- Znak konverzije c učitava jedan znak u varijablu bez obzira je li on bjelina ili ne.
- Ako je prvi znak konverzije c potrebno je ispred njega staviti jednu bjelinu kako ne bi pročitao eventualni znak za prijelaz u novi red.

Primjer:

Kontrolni niz " %c%c%c" čita tri znaka. Započet će s prvim znakom koji nije bjelina, a onda će učitati još dva uzastopna znaka bili oni bjeline ili ne.
Ako želimo učitati samo znakove bez bjelina treba koristiti " %c %c %c".

Unos znakovnih nizova

- Znak konverzije s učitava znakovni niz. Niz završava prvom bjelinom u ulaznom nizu znakova.

  Primjer:

```
char string[128];
int x;
.....
scanf("%s%d", string, &x);
```

- Budući da se svako polje (niz) kao argument funkcije interpretira kao pokazivač na prvi element polja, ispred varijable string ne stavlja se adresni operator.

Primjer:

```
int i; float x;
char string[50];

scanf("%d%f%s",&i, &x, string);
```

ulaz: 25 54.32E-1 abc

```
scanf("%2d%f%*d %[0-9]s", &i, &x, string);
```

ulaz: 56789 0123 56a72

```
scanf("%s", string);
```

ulaz: Programski jezik C

Funkcije `scanf` i `gets`

Primjer:

```
char s[30]; int i;
scanf("%d", &i);

gets(s);
puts(s);
printf("%c\n", s[0]);
```

Ulaz_1:
10
abc
Ulaz_2: 10 abc

Funkcija `printf`

```
printf(kontrolni_niz,arg_1,arg_2,…,arg_n);
```

- `kontrolni_niz` sadrži informacije o formatiranju ispisa argumenata.
- Kontrolni znakovni niz ima istu formu i funkciju kao kod funkcije `scanf`.
- Pojedine grupe znakova unutar kontrolnog niza mogu se nastavljati jedna na drugu ili biti odvojene bjelinama ili nekim drugim znakovima.

Najčešće korišteni znakovi konverzije

znak konverzije	tip podatka koji se ispisuje
d,i	decimalni cijeli broj (`int`)
u	cijeli broj bez predznaka (`unsigned`)
o	oktalni cijeli broj (`int`)
x	heksadecimalni cijeli broj (`int`)
e, f, g	broj s pokretnim zarezom (`double`)
c	jedan znak (`char`)
s	string (`char` *)
p	pokazivač (`void` *)

Ispis cijelih brojeva

Primjer:

```
int x = 13;
char c = '1';

printf("%10d%4o %x\n", x, x, x);
printf("%%10d\n", x);

printf("c = %c, c = %d\n", c, c);
```

Ispis realnih brojeva

- Brojeve tipa `float` i `double` možemo ispisati pomoću znakova konverzije f, g i e. U konverziji tipa f broj se ispisuje bez eksponenta, a u konverziji tipa e s eksponentom. U konverziji tipa g način ispisa (s eksponentom ili bez) ovisi o vrijednosti koja se ispisuje.

  Primjer:

```
#include <math.h>

.....
double pi = 4*atan(1.0);
printf("%f:%e:%g\n", pi, pi, pi);
```

Preciznost ispisa realnih brojeva

%a.bf ili %a.be ili %a.bg

- a – minimalna širina ispisa
- b – broj decimala koje će biti ispisane (preciznost)

Primjer:

```
#include <math.h>
.....
double pi = 4*atan(1.0);
printf("%5f:%12.8f:%8.12f:%5.4f\n", pi, pi, pi, pi);

/* 3.141593:  3.14159265:3.141592653590:3.1416 */
```

Primjer:

```
#include <stdio.h>

int main( ){
   double x = 123.4567;

   printf("%f\n", x);        printf("%20.12f\n", x);
   printf("%g\n", x);        printf("%10.8e\n", x);
   printf("%E\n", x);

   x = 123456789.;
   printf("%g\n", x);
   printf("%G\n", x);
   return 0;
   }
```

Dinamičko zadavanje širine i preciznosti

- Iznos širine (preciznosti) u formatu zamjenjuje *.
- Stvarni iznos širine (preciznosti) određuje cjelobrojna varijabla na odgovarajućem mjestu.

  Primjer:

```
double pi = 4*atan(1.0);
int i = 10;
printf("%*f:%*.*f:%5.*f:%15.*f\n",12, pi,16,14, pi, i, pi, i, pi);
```

/*….3.141593:3.14159265358979:3.1415926536:…3.1415926536 */

Ispis znakovnih nizova

Primjer:

```
char naslov[ ] = "Programski jezik C";
……
printf("%s\n", naslov);
printf("%5.10s\n", naslov);
printf("%30.40s\n", naslov);
```