

Ugrađene funkcije

Ugrađene matematičke funkcije

```
#include <math.h>

int abs (int x);           |x| za int
long labs (long x);       |x| za long int
double fabs (double x);   |x| za double
double sin (double x);    sin x
double cos (double x);    cos x
double tan (double x);    tan x
double asin (double x);   arcsin x
double acos (double x);   arccos x
double atan (double x);   arctan x
```

2

Ugrađene matematičke funkcije

```
double sinh (double x);   sh x
double cosh (double x);   ch x
double tanh (double x);   th x
double exp (double x);     ex
double log (double x);     ln x
double log10 (double x);   log x
double pow (double x, double y); xy;
double sqrt (double x);    √x
double fmod (double x, double y); x mod y
double ceil (double x);    zaokr. na gore
double floor (double x);   zaokr. na dolje
```

3

Ugrađene posebne funkcije

```
#include <stdlib.h>
void exit (int status);
void randomize (void);
ili
void srand (unsigned int seed);
int rand (void);    [0, RAND_MAX] ≡ [0, 32767]
```

Kako jednoliko preslikati cijele brojeve iz intervala [a, b] u interval [c, d]?

$$y = \underbrace{x / (b - a + 1) * (d - c + 1)}_{\text{skaliranje}} + \underbrace{c}_{\text{translacija}}$$

4

Primjer: Načiniti funkciju koja simulira bacanje kocke. Baciti kocku zadani broj puta. Ispisati frekvenciju pojavljivanja svih brojeva.

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
int kocka () {
    return (float) rand () / (RAND_MAX+1) * 6 + 1;
}
int main () {
    int brojac[6] = {0};
    int i, n;
    printf ("RAND_MAX = %d\n", RAND_MAX);
    printf ("Unesite broj bacanja kocke >");
    scanf ("%d", &n);
    /* time() vraća broj sekundi od ponoći, 1. siječnja 1970. */
    srand ((unsigned) time(NULL));
```

5

Primjer: Bacanje kocke, nastavak

```
for (i = 0; i < n; i++) {
    ++brojac[kocka()-1];
}
for (i = 0; i < 6; i++) {
    printf ("%d %5d\n", i+1, brojac[i]);
}
return 0;
}
```

Primjer izvođenja programa:

```
RAND_MAX = 32767
Unesite broj bacanja kocke >1000000
1 166579
2 166446
3 166802
4 167009
5 166714
6 166450
```

6

typedef deklaracija

```
typedef stari_tip novi_tip;
```

npr.

```
typedef unsigned size_t;
typedef int redni_broj;
redni_broj i, j;
size_t velicina;
```

7

Znakovni niz

- Definicija kao polje znakova:

```
#define DULJINA_NIZA 8
char ime_niza[DULJINA_NIZA + 1];
```

npr.

```
char grad[] = "Zagreb";
char niz[80+1];
```

- Sve operacije nad nizovima obavljaju se s pomoću funkcija.

8

Ugrađene funkcije (<string.h>)

- Kopiranje niza:

```
char *strcpy(char *dest, const char *src);
```

Primjer:

```
strcpy (niz, grad);  
printf ("%s %s\n", grad, niz);  
↓  
Zagreb Zagreb
```

9

Ugrađene funkcije (<string.h>)

- Kopiranje dijela niza:

```
char *strncpy(char *dest,  
              const char *src,  
              size_t maxlen);
```

Primjer:

```
strncpy (niz, grad, 4);  
printf ("%s %s\n", grad, niz);  
↓  
Zagreb Zagreb  
Ne postavlja '\0'!  
Trebalo bi: niz[4] = 0;
```

10

Ugrađene funkcije (<string.h>)

- Konkatencija:

```
char *strcat(char *dest, const char *src);
```

Primjer:

```
strcat (niz, grad);  
printf ("%s %s\n", grad, niz);  
↓  
Zagreb ZagrebZagreb
```

11

Ugrađene funkcije (<string.h>)

- Duljina niza:

```
size_t strlen(const char *s);
```

Primjer:

```
printf("%d %d\n", strlen(grad), strlen(niz));  
↓  
6 12
```

12

Ugrađene funkcije (<string.h>)

- Pretvorba u velika ili mala slova:

```
char *strlwr(char *s);
char *strupr(char *s);
```

Primjer:

```
strlwr(niz);
printf ("%s\n", niz);
strupr(niz);
printf ("%s\n", niz);
```

↓

```
zagrebzagreb
ZAGREBZAGREB
```

13

Ugrađene funkcije (<string.h>)

- Usporedba nizova

```
int strcmp(const char *s1, const char *s2);
int strcmpi(const char *s1, const char *s2);
int stricmp(const char *s1, const char *s2);
```

Primjer:

```
printf ("%d\n", strcmp(grad, niz));
niz[6] = '\0';
printf ("%d\n", strcmp(grad, niz));
printf ("%d\n", stricmp(grad, niz));
```

↓

```
32   usporedba Zagreb i ZAGREBZAGREB
32   usporedba Zagreb i ZAGREB
0    kao prethodno, ali se mala i velika slova tretiraju isto
```

14

Ugrađene funkcije (<string.h>)

- Usporedba dijela nizova:

```
int strncmp(const char *s1,
            const char *s2,
            size_t maxlen);

int strncmpi(const char *s1,
             const char *s2,
             size_t maxlen);

int strnicmp(const char *s1,
             const char *s2,
             size_t maxlen);
```

15

Ugrađene funkcije (<string.h>)

- Traženje znaka u nizu

```
char *strchr(const char *s, int c);
Ako se znak ne pronađe vraća se pokazivač na null
```

Primjer:

```
printf ("%s\n", strchr(grad, 'g'));
printf ("%d\n", strchr(grad, 'g') - grad+1);
```

↓

```
greb
3
```

16

Ugrađene funkcije (<string.h>)

- Traženje podniza

```
char *strstr(const char *s1, const char *s2);
```

Primjer:

```
printf ("%s\n", strstr(niz, "AGR"));
```

```
printf ("%d\n", strstr(niz, "AGR") - niz+1);
```

↓

```
AGREB
```

```
2
```

17

Funkcije nad znakom (<ctype.h>)

- Pretvorba u veliko slovo

```
int toupper(int ch);
```

- Pretvorba u malo slovo

```
int tolower(int ch);
```

18

Makro nad znakom(<ctype.h>)

```
int isdigit(int c); znamenka (0-9)
```

↓

```
#define isdigit(c) (c >= '0' && c <= '9')
```

```
int isalpha(int c); slovo (A-Z ili a-z)
```

```
int isalnum(int c); slovo (A-Z ili a-z) ili znamenka (0-9)
```

```
int isprint(int c); znak koji se može ispisati (0x20-0x7E)
```

```
int iscntrl(int c); kontrolni znak (0x7F ili 0x00-0x1F)
```

```
int isspace(int c); praznina
```

```
int islower(int c); slovo (a-z)
```

```
int isupper(int c); slovo (A-Z)
```

19

Izdvajanje rečenica iz unešenog teksta

Zadatak: Učitati s tipkovnice stranicu teksta (veličine do 4096 znakova). Tekst se sastoji od rečenica odvojenih točkama. Unos teksta se završava unosom riječi GOTOVO. U programu treba ispisati tekst tako da svaka rečenica počne u novom retku.

20

Rješenje - deklaracijski blok

```
Recenice
#include <string.h>
#include <stdio.h>
#define MAXTEXT 4096
#define MAXRIJEC 80
#define MAXRECEN 512
int main () {
    char tekst[MAXTEXT+1];
    char rijec[MAXRIJEC+1];
    char recenica[MAXRECEN+1];
    char *pocrec, *tocka;
    int lrijec, ltekst;
```

21

Rješenje - postupak

```
/* Učitavanje ulaznih podataka */
strcpy (tekst, "");
ltekst = 0;
while (1) {
    scanf ("%s", rijec);
    if (strcmp (rijec, "GOTOVO") == 0) break;
    lrijec = strlen (rijec);
    if (ltekst + lrijec + 1 > MAXTEXT)break;
    strcat (tekst, rijec);
    strcat (tekst, " ");
    ltekst += lrijec + 1;
}
```

22

Primjer zadavanja teksta

```
prva recenica.   nova.
recenica.       GOTOVO.
```

Nakon unosa tekst će biti pohranjen u memoriji računala na slijedeći način:

```
p r v a   r e c e n i c a .   n o v a   r e c e n i c a .   0
```

23

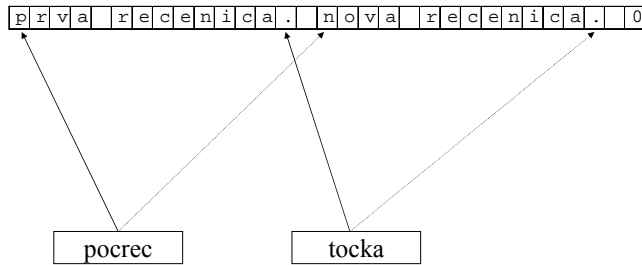
Rješenje - analiza niza i ispis rezultata

```
/* analiza niza i ispis */
pocrec = tekst;
do {
    tocka = strchr (pocrec, '.');
    if (!tocka) {
        if (strlen (pocrec) > 0) {
            printf ("%s\n", pocrec);
        }
    } else {
        strncpy (recenica, pocrec, tocka-pocrec+1);
        recenica[tocka-pocrec+1] = '\0';
        printf ("%s\n", recenica);
        pocrec = tocka+2;
    }
} while (tocka);

return 0;
}
```

24

Prikaz promjene pokazivača



25

Alternativno rješenje učitavanja - znak po znak:

```

ReceniceZnakPoZnak
#include <ctype.h>
...
int i; char c;
i = 0;
do {
    c = getche ();
    if (isprint (c)) tekst[i++] = c;
    else if (c == '\b' && i > 0) i--;
    if (i >= 6 && strcmp (&tekst[i-6],
        "GOTOVO", 6) == 0) break;
} while (i < MAXTEXT);
tekst[i-6] = '\0';
    
```

Backspace

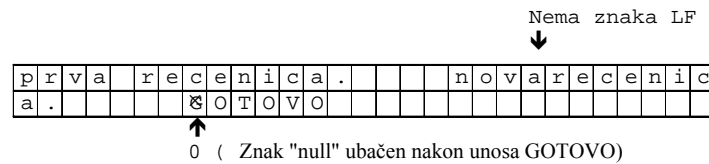
26

Ulazni podaci:

```

prva recenica.  nova
recenica.      GOTOVO
    
```

tekst u memoriji računala:



27

Modifikacija da se izbjegne spajanje riječi:

```

...
c = getche ();
if (c == 0x0d) {
    c = ' '; printf ("\n");
}
...
    
```

Return

Ispis:

```

prva recenica.
nova recenica.
    
```

28

Modifikacija da se izbjegnu suvišne praznine:

- a) kod ispisa

```
...
for (i=0; recenica[i] == ' '; i++);
printf ("%s\n", &recenica[i]);
...
```

29

Modifikacija da se izbjegnu suvišne praznine (ne uzima u obzir \b):

- b) kod unosa

```
...
int preskoci; /* zastavica - flag */
i = 0; preskoci = 1;
do {
    c = getche ();
    if(isprint(c) && (!preskoci || c != ' '))
        tekst[i++] = c;
    preskoci = (c == ' ' || c == '.');
    if (i >= 6 && strcmp (&tekst[i-6],
        "GOTOVO", 6) == 0) break;
} while (i < MAXTEXT);
```

30

Učitavanje i ispis podataka

Učitavanje:

`getchar` `scanf` `gets`

Ispis:

`putchar` `printf` `puts`

31

Funkcija `getchar`

- Funkcija `int getchar(void);` iz `<stdio.h>` koristi se za unos znak po znak. Uspješno pročitani znak pretvara u cijeli pozitivni broj, a nakon nailaska na kraj datoteke (^Z za DOS, ^D za Unix) ili na pogrešku vraća EOF (tj. -1).
- Funkcija se poziva na sljedeći način:
`znakovna_varijabla = getchar();`

32

Primjer čitanja znak po znak

```
IOGetchar
char c;
. . .
while(1) {
    c = getchar();
    printf("%d %c ", c, c);
    if (c == EOF) break;
}
```

Rezultat izvođenja programa:

```
aA↓
↓
97 a 65 A 10
/↓
↓
32 47 / 10
<Ctrl+Z>
↓
-1
```

33

Funkcija putchar

- Funkcija `int putchar(int ch);` iz `<stdio.h>` koristi se za ispis znak po znak

- Funkcija se poziva na sljedeći način:
`putchar(znakovna_varijabla);`

Primjer:

```
char c;
. . .
putchar(c);
```

34

Primjer: Učitati niz slova, a ispisati ih kao velika

```
IOPutchar
#include <stdio.h>
int main() {
    char slovo[80];
    int i, n;
    /* učitavanje linije teksta */
    for(i=0; (slovo[i] = getchar()) != '\n'; i++);
    /* broj procitanih znakova */
    n = i;
    /* ispis linije ali velikim slovima */
    for(i = 0; i < n; ++i)
        putchar(toupper(slovo[i]));

    return 0;
}
```

35

Izvođenje programa

```
Mali auto↓ (U niz je unesen i znak '\n' !)
```

↓

```
MALI AUTO
```

36

Modifikacija: Kriterij za zaustavljanje petlje je EOF

```
for(i = 0; (slovo[i] = getchar()) != EOF; i++);  
n = i;  
for(i = 0; i < n; ++i) putchar(toupper(slovo[i]));
```

Ulazni niz podataka:

```
Mali auto,␣  
dobro,␣  
vozi.␣  
<Ctrl+Z>
```

Ispis:

```
MALI AUTO  
DOBRO  
VOZI.
```

Znak '\n' ostaje sačuvan u nizu
i uzrokuje prijelaz u novi red kod ispisa

37

Funkcija scanf

```
#include <stdio.h>  
...  
int scanf(const char *format, arg1, arg2, ..., arg n);
```

`scanf` vraća broj uspješno obrađenih ulaznih polja koja povezuje s navedenim argumentima. Argumenti moraju odgovarati po broju, redosljedu i tipu formatskim specifikacijama. S obzirom da su argumenti pokazivači, za polje se navodi njegovo ime (pokazivač na nulti član), a za obične varijable se navodi njihova adresa (npr. `&x`).

38

Primjer korištenja funkcije scanf

```
#include <stdio.h>  
int main() {  
    char naziv[20];  
    int sifra;  
    float cijena;  
    scanf("%s %d %f", naziv, &sifra, &cijena);  
    ...  
}
```

Ulazni podaci bi se mogli zadati, na primjer, na sljedeće načine:

```
Indeks 32145 10.5
```

```
Indeks  
32145  
10.5
```

```
Indeks  
32145 10.5
```

```
Indeks 32145  
10.5
```

39

Primjer: Učitavanje niza uz kontrolu postojanja znaka u skupu znakova.

```
char slovo[80];  
scanf("%[ ABCDEFGHIJKLMNOPQRSTUVWXYZ]", slovo);  
printf("%s\n", slovo);
```

Ulazni niz podataka:

```
MALI Auto dobro vozi,␣
```

Ispis:

```
MALI A Znak 'u' nije u definiranom skupu i izazvat će prekid
```

40

Primjer: Učitavanje znakovnog niza sve dok se ne pojavi znak '\n'

```
char slovo[80];  
...  
scanf("%[^\\n]", slovo);
```

Čita dok ne bude unešen znak \n

41

Izgled formatske specifikacije kod funkcije **scanf**

%[širina][modifikator]tip
[širina]
n broj mjesta predviđenih za ulazni podatak

[modifikator]
F argument je udaljeni pokazivač
N argument je bliski pokazivač
h cjelobrojni argument je `short`
l cjelobrojni argument je `long`

42

Izgled formatske specifikacije kod funkcije **scanf**

tip

- c jedan znak
- d cijeli broj s predznakom
- e, f, g broj s pomičnim zarezom
- o oktalni broj
- s znakovni niz ('\\0' se automatski dodaje na kraj)
- x heksadecimalni broj
- u cijeli broj bez predznaka
- [...] znakovni niz u kojemu može biti uključena i praznina

43

Odvajanje formatskih specifikacija kod **scanf**

Pojedinačne formatske specifikacije se mogu pisati neposredno jedna za drugom ili se mogu razdvojiti **prazninom**, **Tab**-om ili znakom `\n`.

Obično se koristi praznina. Paziti kod korištenja c-formata jer on prihvaća bilo koji unešeni znak.

Primjer: Ako se utipka `A B C`, a naredbe za učitavanje su:

```
char x, y, z;  
scanf("%c%c%c", &x, &y, &z);   ⇨   x=A, y= , z=B  
scanf("%c %c %c", &x, &y, &z); ⇨   x=A, y=B, z=C
```

44

Funkcija printf

```
#include <stdio.h>
...
int printf(const char *format, arg1, arg2,
           ..., arg n);
```

`printf` kao rezultat daje broj bajtova ispisanih na standardnoj izlaznoj jedinici (`stdout`). Argumenti mogu biti varijable, imena polja ili kompliciraniji izrazi.

45

Primjer korištenja funkcije printf

```
#include <stdio.h>
#include <math.h>
int main() {
    float i = 2.0, j = 3.0;
    printf("%f%f %f %f", i, j, i+j, sqrt(i+j));

    return 0;
}
```

Izgled ispisa:

```
2.000000 3.000000 5.000000 2.236068
```

46

Primjer korištenja funkcije printf

```
#include <stdio.h>
int main() {
    double x = 5000.0, y = 0.0025;
    printf("%f %f %f %f\n", x, y, x * y, x / y);
    printf("%e %e %e %e", x, y, x * y, x / y);

    return 0;
}
```

Izgled ispisa:

```
5000.000000 0.002500 12.500000 2000000.000000
5.000000e+003 2.500000e-003 1.250000e+001 2.000000e+006
```

47

Izgled formatske specifikacije kod funkcije printf

`[%znak][širina][.preciznost][modifikator] tip`

[znak]	
ništa	desno pozicioniranje
praznina	tiska - predznak, a umjesto + predznaka je praznina
-	lijevo pozicioniranje
+	rezultat uvijek počinje s + ili -
#	konverzija na alternativan način

48

Izgled formatske specifikacije kod funkcije `printf`

[širina]

n najmanje *n* mjesta
0n najmanje *n* mjesta s tim da se lijevo stavljaju nule

[.preciznost]

ništa preciznost po definiciji
.0 d, o, u, x preciznost po definiciji
e, f bez decimalne točke
.n najviše *n* znakova

49

Izgled formatske specifikacije kod funkcije `printf`

tip

c znak
d cijeli broj s predznakom
u cijeli broj bez predznaka
o oktalni broj bez predznaka bez vodećih nula
e broj s pomičnim zarezom prikazan u eksponencijalnom obliku
f broj s pomičnim zarezom
g broj s pomičnim zarezom (e ili f oblika ovisno o vrijednosti)
s znakovni niz
x heksadecimalni broj bez oznake 0x ispred rezultata

50

Primjeri korištenja funkcije `printf`

```
#include <stdio.h>
int main() {
    char tekst[]="studomat";
    printf("%5s %10s %10.5s %.5s",
           tekst,tekst,tekst,tekst);
    return 0;
}
```

Ispis:

```
studomat   studomat   studo studo
```

51

Primjeri korištenja funkcije `printf`

```
#include <stdio.h>
int main() {
    int i = 123; float x = 12.0, y = -3.3;
    printf(":%6d %7.0f %10.1e:\n", i, x, y);
    printf(":%-6d %-7.0f %-10.1e:\n", i, x, y);
    printf(":%+6d %+7.0f %+10.1e:\n", i, x, y);
    printf(":%++6d %++7.0f %++10.1e:\n", i, x, y);
    printf(":%7.0f %#7.0f %7g %#7g:\n", x,x, y,y);
    return 0;
}
```

Ispis:

```
:  123      12  -3.3e+000:
:123      12   -3.3e+000 :
:  +123     +12  -3.3e+000:
: +123    +12   -3.3e+000 :
:         12   12.  -3.3 -3.30000:
```

52

Primjeri korištenja funkcije printf

```
#include <stdio.h>
int main() {
    int i=1234, j=01777, k=0xa08c;
    printf("%8u %8o %8x:\n", i, j, k);
    printf(":%-8u %-8o %-8x:\n", i, j, k);
    printf(":%#8u %#8o %#8X:\n", i, j, k);
    printf(":%08u %08o %08X:\n", i, j, k);
    return 0;
}
```

Ispis:

```
:   1234    1777    a08c:
:1234    1777    a08c  :
:   1234    01777   0xA08C:
:00001234 00001777 0000A08C:
```

53

Funkcije gets i puts

```
#include <stdio.h>

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

Ove funkcije nude jednostavnu zamjenu za `scanf` i `printf` kada se radi o znakovnim nizovima. Kada se unos obavlja s pomoću `gets` unos se smatra gotovim kada korisnik stisne *Enter*.

54

Primjer korištenja funkcija gets i puts

```
#include <stdio.h>
int main() {
    char red[80];
    gets(red);
    puts(red);

    return 0;
}
```

Još jedan primjer:

📁 ElementarnaEnkripcija

55

Zadatak: Pročitati vrijednosti za $mr \leq 50$ i $ns \leq 10$. Pročitati vrijednosti članova dvodimenzionalnog realnog polja od mr redaka i ns stupaca. Ispisati pročitano polje, sume redaka i sume stupaca te ukupnu sumu u obliku:

```
Polje A:
      !   1   2   3   4 ... Sumr
-----
  1 ! xxx.x xxx.x xxx.x xxx.x ... xxx.x
-----
... !
      !
-----
Sums! xxx.x xxx.x xxx.x xxx.x ... xxx.x
```

Rješenje u pseudokodu

```
čitaaj mr i ns dok ne budu ispravni
učitaaj realno polje od mr redaka i ns
stupaca
ispisi naslov
ponavljaj za sve retke od 1 do mr
| izračunaj sumu retka
| ispiši redak i sumu
| ispiši crtu
izračunaj sume stupaca i ukupnu sumu
ispisi sume stupaca i ukupnu sumu
kraj
```

57

Rješenje u C-u

```
SumeRedakaISTupaca
#include <stdio.h>
#define MAXR 50
#define MAXS 10
int main() {
    int mr, ns, i, j;
    float a[MAXR][MAXS], sums[MAXS], sumr, ukupno;
    /* čitanje mr i ns dok ne budu ispravni */
    do {
        printf("Upisite vrijednosti za broj redaka i
        stupaca: ");
        scanf ("%d %d",&mr, &ns);
    } while (mr <= 0 || mr > MAXR ||
            ns <= 0 || ns > MAXS);
```

58

Rješenje u C-u, nastavak

```
/* čitanje polja od mr redaka i ns stupaca */
printf("Upisite polje po retcima\n");
for (i=0; i < mr; i++) {
    for (j=0; j < ns; j++) {
        scanf("%f", &a[i][j]);
    }
}
/* ispis naslova */
printf("          Polje A:\n\n");
printf("          !");
for (j=1; j <= ns; j++) printf("%6d",j);
printf("  Sumr\n");
for (j=1; j <= 6*(ns+1)+5; j++) printf("%c",'=');
printf("\n");
```

59

Rješenje u C-u, nastavak

```
/* ponavljaj za sve retke od 1 do mr */
for (i=0; i<mr; i++) {
    sumr = 0;
    for (j=0; j<ns; j++) {
        sumr += a[i][j];
    }
    printf("%3d !",i+1);
    for (j=0; j<ns; j++) printf(" %5.1f", a[i][j]);
    printf(" %5.1f\n", sumr);
    for (j=1; j <= 6 * (ns+1) + 5; j++) printf("-");
    printf("\n");
}
```

60

Rješenje u C-u, nastavak

```
/* izračunaj sume stupaca i ukupnu sumu */
ukupno = 0;
for (j=0; j<ns; j++) {
    sums[j] = 0;
    for (i=0; i<mr; i++) {
        sums[j] += a[i][j];
    }
    ukupno += sums[j];
}
/* ispisi sume stupaca i ukupnu sumu */
printf("Sums!");
for (j=0; j<ns; j++) printf(" %5.1f", sums[j]);
printf(" %5.1f\n", ukupno);

return 0;
}
```

61

Rezultat izvođenja

Upisite vrijednosti za broj redaka i stupaca: 2 4↓

Upisite polje po retcima

1 2 3 4↓

0.1 0.2 -0.1 -0.2↓

Polje A:

	!	1	2	3	4	Sumr
1 !		1.0	2.0	3.0	4.0	10.0
2 !		0.1	0.2	-0.1	-0.2	0.0
Sums!		1.1	2.2	2.9	3.8	10.0

62