Основи програмирања 1
Лекција 7

др Зоран Бањац

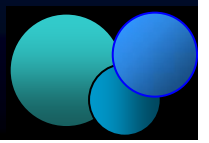
zoran.banjac@viser.edu.rs

Висока школа електротехнике и рачунарства струковних студија Београд

Садржај

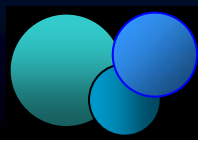
Знакови (*character*)

- Кодирање знакова
- Константе
- Матрице знакова
- Улазно излазна конверзија
- Библиотечке функције за рад са знаковима



Знакови (*character*)

- ***C*** има само нумеричке типове података.
- Потреба за рад са текстом...
- Постоји тип **`char`**
  - целобројан тип
  - предвиђен за представљање појединачних карактера
- За представљање низа знакова (текста) променљиве дужине, који се називају ***знаковни низови*** користи се низ типа **`char`**.



Кодирање знакова (*ASCII* кôд)

Ctrl	Dec	Hex	Char	Code
^@	0	00		NUL
^A	1	01	☺	SOH
^B	2	02	☻	STX
^C	3	03	♥	ETX
^D	4	04	♦	EOT
^E	5	05	♣	ENQ
^F	6	06	♠	ACK
^G	7	07	•	BEL
^H	8	08	◘	BS
^I	9	09	○	HT
^J	10	0A	◙	LF
^K	11	0B	♂	VT
^L	12	0C	♀	FF
^M	13	0D	♪	CR
^N	14	0E	♫	SO
^O	15	0F	☼	SI
^P	16	10	►	DLE
^Q	17	11	◄	DC1
^R	18	12	↕	DC2
^S	19	13	‼	DC3
^T	20	14	¶	DC4
^U	21	15	§	NAK
^V	22	16	▬	SYN
^W	23	17	↨	ETB
^X	24	18	↑	CAN
^Y	25	19	↓	EM
^Z	26	1A	→	SUB
^[	27	1B	←	ESC
^\	28	1C	∟	FS
^]	29	1D	↔	GS
^^	30	1E	▲	RS
^_	31	1F	▼	US

Dec	Hex	Char
32	20	sp
33	21	!
34	22	"
35	23	#
36	24	$
37	25	%
38	26	&
39	27	'
40	28	(
41	29	)
42	2A	*
43	2B	+
44	2C	,
45	2D	-
46	2E	.
47	2F	/
48	30	0
49	31	1
50	32	2
51	33	3
52	34	4
53	35	5
54	36	6
55	37	7
56	38	8
57	39	9
58	3A	:
59	3B	;
60	3C	<
61	3D	=
62	3E	>
63	3F	?

Dec	Hex	Char
64	40	@
65	41	A
66	42	B
67	43	C
68	44	D
69	45	E
70	46	F
71	47	G
72	48	H
73	49	I
74	4A	J
75	4B	K
76	4C	L
77	4D	M
78	4E	N
79	4F	O
80	50	P
81	51	Q
82	52	R
83	53	S
84	54	T
85	55	U
86	56	V
87	57	W
88	58	X
89	59	Y
90	5A	Z
91	5B	[
92	5C	\
93	5D	]
94	5E	^
95	5F	_

Dec	Hex	Char
96	60	`
97	61	a
98	62	b
99	63	c
100	64	d
101	65	e
102	66	f
103	67	g
104	68	h
105	69	i
106	6A	j
107	6B	k
108	6C	l
109	6D	m
110	6E	n
111	6F	o
112	70	p
113	71	q
114	72	r
115	73	s
116	74	t
117	75	u
118	76	v
119	77	w
120	78	x
121	79	y
122	7A	z
123	7B	{
124	7C	|
125	7D	}
126	7E	~
127	7F	⌂†

† ASCII code 127 has the code DEL. Under MS-DOS, this code has the same effect as ASCII 8 (BS). The DEL code can be generated by the CTRL+BKSP key.

Знаковне константе

- Знаковна константа је цео број чија је вредност једнака кôду жељеног знака
- Податку типа **char** може да се додели ASCII кôд жељеног знака или сам знак написан под ЈЕДНОСТРУКИМ знаковима навода.

```
#include<stdio.h>
int main(){
 char c1, c2, c3;
 c1='A';
 c2=65;
 c3='#';
 printf("Vrednost znakova: c1=%c, c2=%c, c3=%c\n",
                                    c1,c2,c3);
}
```

```
Vrednost znakova: c1=A, c2=A, c3=#
```

Знаковне константе

- Само се децимални бројеви пишу без знакова навода
- Октални бројеви се пишу иза знака косе црте уназад (\ ) нпр. '\101'. (декадно 65)
- Хексадецимални број се пише иза знакова \x или \X. Нпр. '\x41'. (декадно 65)

```
main()
{
 char c1, c2, c3;
 c1='\101';
 c2=65;
 c3='\x41';
 printf("Vrednost znakova: c1=%c, c2=%c,
  c3=%c\n",c1,c2,c3);
}
```

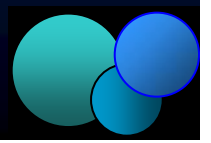
```
Vrednost znakova: c1=A, c2=A, c3=A
```

Знаковни низ

- Најчешћи термин који се користи за знаковне низове је **стринг**
- Стринг је секвенца од нула или више знакова који се пишу у овиру ДВОСТРУКИХ знакова навода.

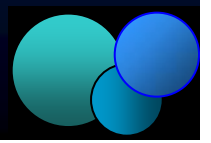
```
"John L. Johanson"
"Zdravo svima!"
```

- Сваки стринг се завршава са тзв. нул знаком `'\0' (null character)`.
- Знаковни низ у општем случају не мора (али може) да садржи нул знак ('\0')



Стринг – низ знакова

- 'A' представља један знак: **A**,
- "A" представља стринг тј. два знака: **'A'** и **'\0'**
- Да би се у меморију уписало 'A' потребна је једна меморијска јединица типа **char**
- За **"A"** су потребне две меморијске јединице: једна за **'A'** и једна за **'\0'**.
- стринг **"Zdravo"** представља 7 знакова: **'Z','d','r','a','v','o'** и **'\0'**

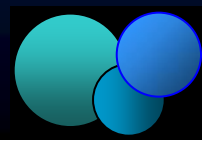


Стринг - низ знакова

- char ime[16]={'J', 'o', 'h', 'n', '\0'}; је исто као и
  char ime[16]="John";

- char imeA[4]={'J', 'o', 'h', 'n'};
  char imeB[5]="John";
  низови imeA и imeB нису исти.

- Дужина низа се не мора навести уколико се низу додељује почетна вредност
  char ime[ ]={'J', 'o', 'h', 'n', '\0'};
  char ime[ ]="John";



Стринг - низ знакова

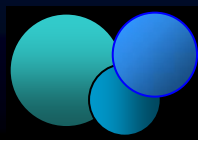
- Стринг је једнодимензионални низ знакова
- То је низ знакова који се завршава *null* знаком (знак чији је код 0) ’\0’

- **Декларација стринга:**

```
char ime[duzina] = "string" ;
```

или

```
char ime[duzina] = { znak, znak …,'\0'};
```

Стринг - низ знакова

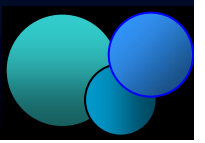
- Пример:

```
char grad1[] = "Beograd" ;
char grad2[] = { 'N', 'I', 'S',
'\0' };
```

- Означавање знаковних и стринг константи:

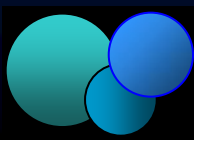
'D' – ovo je znak D

"D" – ovo je string : 'D','\0'



Симболичке константе

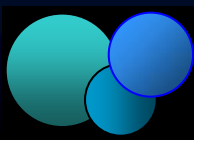
- `#define ZVONO '\a'`
- `#define NOVI_RED '\n'`
- `#define VALUTA "US dollar"`



Матрице знакова

```
char daniA[7][11]={"ponedeljak", "utorak",
                                  ..., "nedelja"};
```

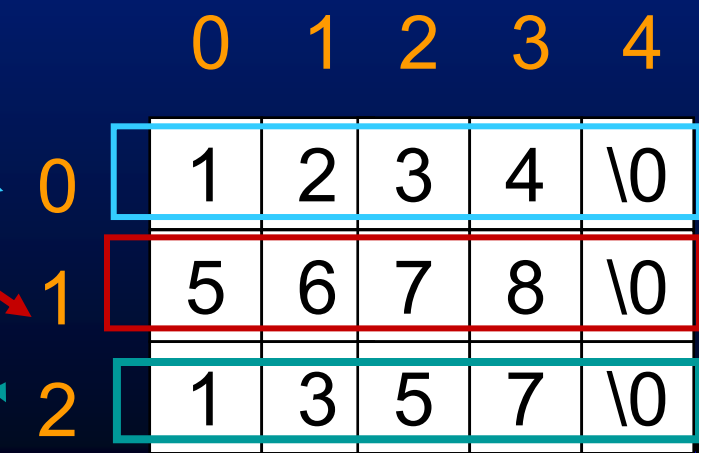
p	o	n	e	d	e	l	j	a	k	\0
u	t	o	r	a	k	\0				
s	r	e	d	a	\0					
c	e	t	v	r	t	a	k	\0		
p	e	t	a	k	\0					
s	u	b	o	t	a	\0				
n	e	d	e	l	j	a	\0			



Дводимензионални низови

```
#include<stdio.h>
main()
{
  char niz[3][5]={"1234","5678","1357"};

  printf("prvi %s\ndrugi %s\ntreci
  %s\n",niz[0],niz[1],niz[2]);
}
```

```
prvi 1234
drugi 5678
treci 1357
```



Улазно излазна конверзија

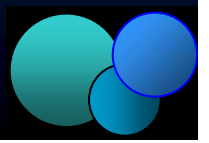
```
#include <stdio.h>
int main()
{
 char a[6] = {'A','l','e','k','s','a'};
 char b[5] = "Pera";
 printf("\nIspis karaktera: %c", a[3]);
 printf("\nIspis stringa: %s\n", b);
 return 0;
}
```

Излаз:

Ispis karaktera: k

Ispis stringa: Pera

Шта би био резултат да је овде %d?

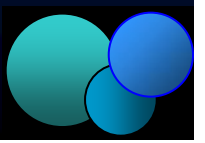


Читање знакова без конверзије

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

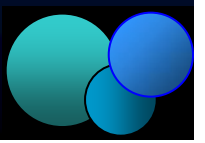
- Чита следећи знак, укључујући и беле знакове, са стандардног улаза (тастатуре).

```
#include <stdio.h>
int main()
{
   int znak;
   printf("Unesite karakter\n");
   znak = getchar();
   printf(" Unet je znak %c\n",znak);
   return 0;
}
```

Читање знака без конверзије

```
#include <stdio.h>
main()
{
 char znak;
 printf("Unesite tekst. Zavrsite sa  tackom:");
  do
  {
    znak = getchar();
    printf("%c",znak);
  } while (znak != '.');
}
```

Пример

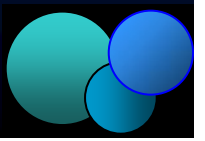
```c
#include <stdio.h>
void main( void )
{
  char bafer[81];
  int i, ch;
  printf( "Unesite tekst: " );

  for( i = 0; i < 80; i++ )
  {
    ch = getchar();
    if(ch == '\n')
       break;

    bafer[i] = (char)ch;
   }
   printf( "%s\n", bafer );
}
```

```
Unesite tekst: Pera
Pera╠╠╠╠╠╠╠╠╠╠╠╠╠╠╠╠╠╠╠╠╠╠╠╠╠╠╠╠╠╠╠╠╠╠╠╠╠╠╠╠╠╠╠╠╠╠╠
╠╠╠╠╠╠╠╠╠╠╠╠╠╠╠╠╠╠╠╠╠╠╠╠╠╠╠╠╠╠╠╠╠└ ↕
```

Пример

```
#include <stdio.h>
void main( void )
{
  char bafer[81];
  int i, ch;
  printf( "Unesite tekst: " );

  for( i = 0; i < 80; i++ )
  {
    ch = getchar();
    if(ch == '\n')
       break;

    bafer[i] = (char)ch;
   }
  bafer[i] = '\0';
  printf( "%s\n", bafer );
}
```

```
Unesite tekst: Pera
Pera
```

Писање знака без конверзије

```
int putchar (int character );
```

- Исписује знак на главни излаз (монитор).
- Повратна вредност исписани карактер или EOF

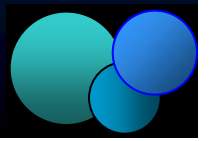
```
#include <stdio.h>
int main()
{
  int znak;  /* moze  char znak */
  for (znak = 'A' ; znak <= 'Z' ; znak++)
  {
    putchar(znak);
  }
}
```

```
ABCDEFGHIJKLMNOPQRSTUVWXYZ
```

Читање знакова без конверзије

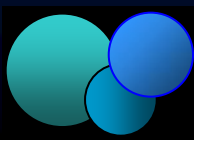
```
char * gets(char *imeNiza );
```

- Прихвата стринг са тастауре у иписује га у низ у низ **imeNiza**
- аргумент: **char *imeNiza** је име низа у који се уписује стринг
- повратна вредност: је показивач на прихваћени низ или NULL ако је дошло до грешке
- Уписује у низ унете знакове са тастатуре (до знака за прелазак у нови ред тј. \n).
- Уместо знака \n уписује знак \0.



Пример gets

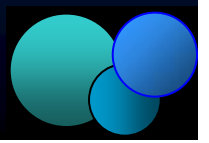
```
#include<stdio.h>
int main()
{
  char str[256];

  printf ("Upisite vasu adresu: ");
  gets(str);
  printf ("Vasa adresa je: %s\n", str);
  return 0;
}
```

Писање знакова без конверзије

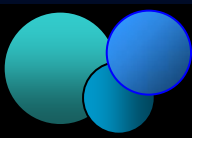
```
int  puts( char *niz );
```

- аргумент: **`char *niz`** је име низа
- повратна вредност:
  не негативна вредност или
  EOF ако је дошло до грешке
- Исписује садржај знаковног низа **`niz`** до завршног знака `\0`
  Додаје знак за прелазак у нови ред `\n` иза последњег знака.



Пример puts

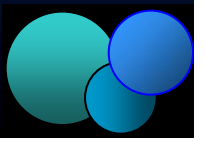
```
#include<stdio.h>
int main()
{
   char string [ ] = "Hello world!";
   puts(string);
   puts("Neki tekst");

   return 0;
}
```

Пример

```
#include<stdio.h>
int main()
{
   char  niz[3][5]={"1234","5678","1357"};

  /* ispis sva 3 stringa */
  puts(niz[0]);
  puts(niz[1]);
  puts(niz[2]);

  printf("Unesite rec do 4 znaka: ");
  gets(niz[0]);

  /* ispis sva 3 stringa */
  puts(niz[0]);
  puts(niz[1]);
  puts(niz[2]);
}
```

```
1234
5678
1357
Unesite rec do 4 znaka: car
car
5678
1357
```

Стрингови - рекапитулација

Исписивање стринга

помоћу функције

```
printf(”%s”, string);
```

```
#include<stdio.h>
main()
{
   char ime[]= "Ana";
   printf("%s\n",ime);
}
```

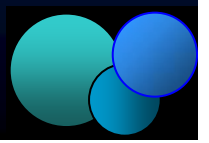
```
Ana
```

помоћу функције

```
puts(string);
```

```
#include<stdio.h>
main()
{
 char ime[]= "Ana";
 puts(ime);
}
```

```
Ana
```

Стрингови - рекапитулација

Учитавање стрингова

помоћу функције

```
scanf(”%s”, &string);
```

```
#include <stdio.h>
main()
{
 char tekst[50];
printf("Unesi recenicu:");
 scanf("%s", tekst);
 printf("%s", tekst);
}
```

```
Unesi recenicu: Novi Sad
Novi
```

помоћу функције

```
gets(string);
```

```
#include <stdio.h>
main()
{
 char tekst[50];
printf("Unesi recenicu:")
 gets(tekst);
 printf("%s", tekst);
}
```

```
Unesi recenicu: Novi Sad
Novi Sad
```

Стрингови - Пример

Учитати стринг, и одредити и исписати његову дужину.

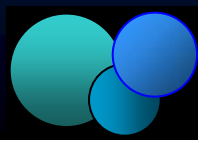
```
#include <stdio.h>
main()
{
 char s ;
 char grad[] = "Pancevo";
 int i = 0;

 do
 {
  s = grad[i];
  i++
 }while(s != '\0');

 printf("Duzina Stringa je: %d\n", i );
}
```

```
Duzina Stringa je: 8
```

Испитивање знакова

```
#include <ctype.h>
int isalpha(int ch);
```

- Функција **isalpha**, проверава да ли њен аргумент (**int ch**) из скупа малих и великих слова (a-z, A-Z).
- Повратна вредност
  - различита од нуле ако је унета вредност из скупа (a-z, A-Z)
  - 0 ако није из тог скупа



Пример isalpha

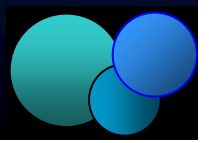
```
#include <stdio.h> /* zbog printf*/
#include <ctype.h> /* zbog isalpha */
int main()
{
 int i=0;
 char str[]="C++";
 while (str[i]) /* str[3] je 0 */
 {
  if (isalpha(str[i]))
    printf("karak. %c je slovo\n",str[i]);
  else
    printf("karak. %c nije slovo\n",str[i]);
  i++;
 }
}
```

```
karak.  C je slovo
karak. + nije slovo
karak. + nije slovo
```

Испитивање знакова

```
#include <ctype.h>
int isdigit(int ch);
```

- Функција **`isdigit`**, проверава да ли је њен аргумент (**`int ch`**) из скупа декадних цифара (0-9).
- Повратна вредност
  - различита од нуле ако је унета вредност из скупа (0-9)
  - 0 ако није из тог скупа



Пример isdigit

```
#include <stdio.h>
#include <ctype.h>
int main() /* ispis cifara iz stringa */
{
  char str[80];  int i=0;

  printf("Unesite niz karaktera\n");
  gets(str);

 while(1)
 {
  if(str[i] == '\0')
       break;
    if(isdigit(str[i]))
      printf(" %c", str[i]);
    i++;
  }
}
```

Испитивање знакова

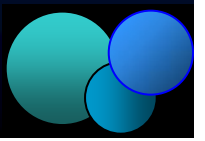
```
#include <ctype.h>
int isalnum(int ch);
```

- Функција **isalnum** има за повратну вредност цео број (**int**) различит од нуле ако њен улазни аргумент (**int ch**) има вредност која одговара ASCII коду знакова који спадају у групу бројева (0 до 9) или слова (**a** до **z** и **A** до **Z**). У супротном повратна вредност функције је 0.

```
char ch;
scanf( "%c", &ch );
if( isalnum(ch) )
   printf("Uneli ste alfanumericki znak
                                    %c", ch );
```

Пример isalnum

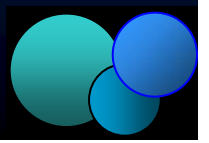
```
/* da li je ch iz skupa slova i cifara? */

#include <ctype.h>
#include <stdio.h>
main()
{
   short int i;

    for (i = 0; i < 256; i++)
    {
      if(isalnum(i))
         printf("isalnum (%c) Da\n", i);
      else
         printf("isalnum (%c) Ne\n", i);

    }
}
```

Испитивање знакова

```
#include <ctype.h>
int islower(int ch);
```

- Функција **`islower`**, проверава да ли је њен аргумент (**`int ch`**) из скупа малих слова (a-z).
- Повратна вредност
  - различита од нуле ако је унета вредност из скупа (a-z)
  - 0 ако није из тог скупа



Пример islower

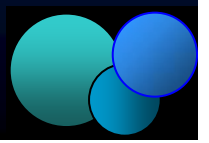
```
#include <stdio.h>
#include <ctype.h>
int main ()
{
  int i=0;
  char str[]="Test String.\n";
  char c;
  while (str[i])
  {
    c=str[i];
    if (islower(c))
    putchar (c);
    i++;
  }
  return 0;
}
```

esttring

Испитивање знакова

```
#include <ctype.h>
int isupper(int ch);
```

- Функција **`isupper`**, проверава да ли је њен аргумент (**`int ch`**) из скупа великих слова (A-Z).
- Повратна вредност
  - различита од нуле ако је унета вредност из скупа (A-Z)
  - 0 ако није из тог скупа



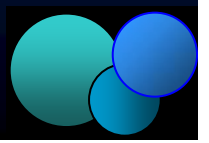
Пример isupper

```
#include <stdio.h>
#include <ctype.h>
int main ()
{
  int i=0;
  char str[]="Test String.\n";
  char c;
  while (str[i])
  {
    c=str[i];
    if (isupper(c))
    putchar (c);
    i++;
  }
  return 0;
}
```

TS

Испитивање знакова

- int isalpha (int c); c је слово?
- int isdigit (int c); c је број?
- int islower (int c); c је мало слово?
- int isupper (int c); c је велико слово?
- int isdigit (int c); c је цифра 0-9 ?
- int isxdigit (int c); c је хекса цифра 0-9,a-f, A-F ?

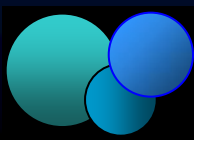


Конверзија знакова

```
#include <ctype.h>
int tolower(int ch);
```

Функција, конвертује мала слова у велика.

- Повратна вредност:
  - мало слово уколико постоји еквивалент аргумента ch
  - ch уколико не постоји одговарајући еквивалент.



Пример tolower

```
#include <stdio.h>
#include <ctype.h>
int main()
{
  int i=0;
  char str[]="Test String.\n";
  char c;
  while (str[i]) {
    c=str[i];
    putchar (tolower(c));
    i++;
  }
}
```

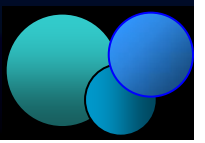
```
test string.
```

Конверзија знакова

```
#include <ctype.h>
int toupper(int ch);
```

Функција, конвертује мала слова у велика.

- Повратна вредност:
  - мало слово уколико постоји еквивалент аргумента ch
  - ch уколико не постоји одговарајући еквивалент.



Пример tolower

```
#include <stdio.h>
#include <ctype.h>
int main()
{
  int i=0;
  char str[]="Test String.\n";
  char c;
  while (str[i]) {
    c=str[i];
    putchar (toupper(c));
    i++;
  }
}
```

```
TEST STRING.
```

Рад са знаковним низовима

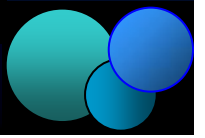
```
#include<string.h>
int strlen(char *niz)
```

- Функција, враћа дужину стринга чија адреса `(char *niz)` је наведена као њен аргумент
- име низа представља адресу првог елемента низа
- дужина стринга не обухвата NULL

Пример strlen

```
#include<string.h>
#include<stdio.h>
main ()
{
  char niz[]="Koliko imam karaktera";
  int duzina;

  duzina=strlen(niz);
  printf("U stringu: -%s- ima %d
  karakter\n",niz, duzina);
}
```

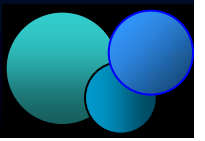
```
U stringu: -Koliko imam karaktera- ima 21 karakter
```

Рад са знаковним низовима

```
#include<string.h>
char * strcpy(char *dest, char
  *src );
```

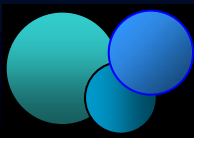
- Функција, копира стринг чију адресу садржи показивач **`src`** у стринг чију адресу садржи показивач **`dest`**.
- Копира се и знак **`'\0'`**
- Треба да се води рачуна о димензијама одредишног стринга



Пример strcpy

```
#include <stdio.h>
#include <string.h>
main ()
{
 char str1[ ]="Sadrzaj stringa 1";
 char str2[40];
 char str3[40];
 strcpy (str2,str1);
 strcpy (str3,"Kopiranje uspesno");
 printf ("str1: %s\nstr2: %s\nstr3:%s\n",
                        str1,str2,str3);
}
```

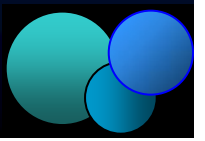
```
str1: Sadrzaj stringa 1
str2: Sadrzaj stringa 1
str3: Kopiranje uspesno
```

Пример strcpy

```
#include <stdio.h>
#include <string.h>
main ()
{
 char str1[ ]="abcdefghijklm";
 char str2[40]="ABCDEFG";

 strcpy (&str2[4],&str1[3]);

 printf ("str1: %s\nstr2: %s\n",str1,str2);
}
```

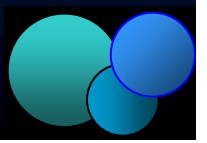
```
str1: abcdefghijklm
str2: ABCDdefghijklm
```

Рад са знаковним низовима

```
#include<string.h>
char * strncpy(char *dest, char *src,
  n );
```

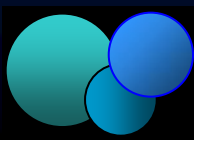
- Функција, копира првих n карактера из стринга чију адресу садржи показивач **src** у стринг чију адресу садржи показивач **dest.**
- Треба да се води рачуна о димензијама одредишног стринга



Пример strncpy

```
#include <stdio.h>
#include <string.h>
main ()
{
   char str1[] = "To be or not to be";
   char str2[6];
   strncpy (str2, str1, 5);
   str2[5]='\0'; /* !! */
   puts (str2);
}
```

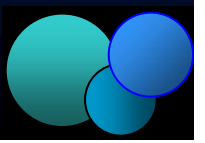
```
To be
```

Пример strncpy

```
#include <stdio.h>
#include <string.h>
int main()
{
   char str1[] = "To be or not to be";
   char str2[16]="123";
   strncpy (&str2[3], &str1[5], 5);
   puts(str2);
   return 0;
}
```

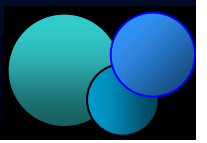
123 or n



Рад са знаковним низовима

```
#include<string.h>
char *strcat(char *dest, char
  *src);
```

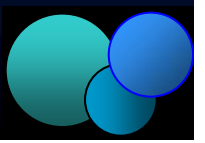
- Функција, дописује стринг чију адресу садржи показивач **`src`** на стринг чију адресу садржи показивач **`dest`**.
- Треба да се води рачуна о димензијама одредишног стринга



Пример strcat

```
#include <string.h>
#include<stdio.h>
int main( )
{
  char str[80];
  strcpy(str,"Stringovi  ");
  strcat(str,"su ");
  strcat(str,"spojeni.");
  puts(str);
  return 0;
}
```

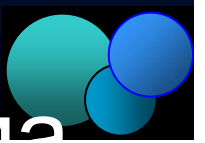
```
Stringovi  su spojeni.
```

Рад са знаковним низовима

```
#include<string.h>
char * strncat (char *dest, char
  *src, n);
```

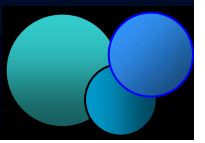
- Функција, дописује првих n карактера из стринга чију адресу садржи показивач **src** на стринг чију адресу садржи показивач **dest.**
- ако полазни стринг има мање од n карактера дописује се његов цео садржај.
- Треба да се води рачуна о димензијама



Пример strncat

```
#include<string.h>
#include<stdio.h>
int main()
{
  char str1[20];
  char str2[20];
  strcpy (str1,"To be ");
  strcpy (str2,"or not to be");
  strncat (str1, str2, 6);
  puts (str1);
  return 0;
}
```

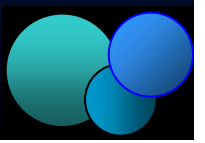
```
To be or not
```

Рад са знаковним низовима

```
#include<string.h>
int strcmp(char *str1, char *str2);
```

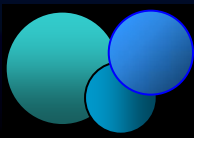
- Функција, пореди два стринга
- Почиње са поређењем првих карактере оба стринга. Уколико су они једнаки наставља са поређењем следећег пара све док не наиђе на различите или до не наиђе на крај стринга.
- Враћа цео број као резултат поређења:
  - 0 стрингови су једнаки
  - >0 први различит карактер у str1 има већу вредност од одговарајућег карактера у str2
  - <0 у супротном случају



Рад са знаковним низовима

```
#include<string.h>
#include<stdio.h>
int main()
{
 char Resenje[] = "kruska";
 char Odgovor[80];
 do{
    printf("Moje omiljeno voce? ");
    gets (Odgovor);
 } while(strcmp (Resenje,Odgovor) != 0);
 printf ("Odgovor tacan!\n");
 return 0;
}
```

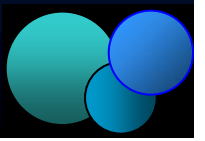
```
Moje omiljeno voce? jabuka
Moje omiljeno voce? kruska
Odgovor tacan!
```

Конверзија у нумеричке типове података

```
#include<math.h>
#include<stdlib.h>
double  atof(char *niz);
```

- Конвертује стринг у реалан број
- Прво одбаци све “празне” знаке. Почев од тог знака узима у обзир све наредне знаке од којих може да формира реалан број.
- Од првог знака који не задовољава овај критеријум престаје даља претрага



Пример atof

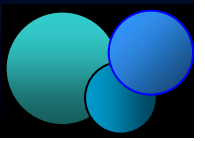
```
Unesite vrednost u stepenima: 90
sin(90.000000) je 1.000000
```

```
#include<math.h>
#include<stdlib.h>

int main ()
{
  double n,m;
  double pi=3.1415926535;
  char Ulaz[256];
  printf( "Unesite vrednost u stepenima: " );
  gets( Ulaz );
  n = atof( Ulaz );
  m = sin(n*pi/180);
  printf ( "sin(%f) je %f\n" , n, m );
  return 0;
}
```

Конверзија у нумеричке типове података

```
int  atoi (char *niz);  /* конверзија у тип int */
```

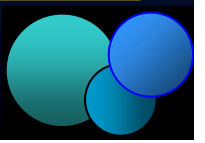
```
#include<math.h>
#include<stdlib.h>
int main()
{
  int i;
  char Ulaz[256];
  printf ("Unesite broj: ");
  gets( Ulaz );
  i = atoi(Ulaz);
  printf("Uneta vrednost je %d, a njena
     dvostruka vrednost je %d\n",i,i*2);
  return 0;
}
```

Конверзија у нумеричке типове података

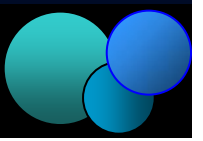
`long atol (string);` /*конверзија у тип *long* */

```
#include<math.h>
#include<stdlib.h>
int main()
{
  int i;
  char Ulaz[256];
  printf ("Unesite broj: ");
  gets ( Ulaz );
  i = atol (Ulaz);
  printf ("Uneta vrednost je %d, a njena
  dvostruka vrednost je %d\n",i,i*2);
  return 0;
}
```

Пример 1

```
#include<string.h>
#include<stdio.h>
main ()
{
  char niz[80];
  do
  {
   puts("Unesite jedan red teksta");
   gets(niz);
   if(strlen(niz)>=10)
      puts(niz);
  }while(strcmp(niz,"Kraj"));

}
```

Пример 2

```
#include<string.h>
#include<stdio.h>
main ()
{
  char niz[80];
  int duzina,i;
  do
  {
   puts("Unesite jedan red teksta");
   gets(niz);

   duzina = strlen(niz);
   for(i = 0; i < duzina; i++)
   {
     if(niz[i] == ' ')
      niz[i]='_';
   }
   puts(niz);
  }while(duzina);
}
```

Пример 3 1/4

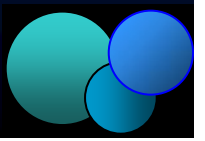
```
#include<stdlib.h>
#include<string.h>
#include<stdio.h>
int main()
{
  int zahtev, i, kraj = 0, postoji;
  char priv[80];
  char niz[ 100 ][80]; /* ime i prezime  */

  /* prikazi zahtevane opcije */
  printf( "Unesite zahtev\n"
       " 1 - Unos imena i prezimena\n"
       " 2 - Izmene postojecih podataka\n"
       " 3 - Prikaz postojecih podataka\n"
       " 4 - Kraj\n? " );

  scanf( "%d", &zahtev );  fflush(stdin);
```

Пример 3 2/4

```
while(zahtev !=4) {
    switch(zahtev){
      case 1:
        puts("Unesite ime novog clana:" );
        gets(priv);
        postoji = 0;
        for(i = 0; i < kraj; i++)
           if(!strcmp(priv, niz[i])) {
             postoji = 1;
             break;
        }
        if(!postoji){
            strcpy(niz[kraj],priv);
            kraj++;
        }

        break;
```

Пример 3 3/4

```
case 2:
   puts("Unesite indeks imena za izmenu");
   gets(priv);
   i=atoi(priv);
   puts("Ime koje zelite da promenite je");
   puts(niz[i]);
   puts("Unesite novo ime ili Enter da
                                    odustanete");
   gets(priv);
   if(strlen(priv))
      strcpy(niz[i],priv);

   break;
```

Пример 3 4/4

```
        case 3:
          puts( "Prikaz clanova liste" );
        for(i=0;i<kraj;i++)
            puts(niz[i]);
          break;
      } /* kraj switch */

      puts( "Unesite zahtev? " );
      scanf( "%d", &zahtev );  fflush(stdin);
    } /* kraj while(zahtev !=4 ) */

    puts("Kraj rada");

  return 0; /* oznacava uspesan zavrsetak */
}
```

Хвала на пажњи

Питања?