Текстуални низи

1. Да се напише функција која од низата знаци ќе ги отстрани бланко (whitespace) знаците што се наоѓаат на крајот од низата.

```
#include <stdio.h>
#include <ctype.h>
#define MAXELEM 50

void RemoveTrail(char *str);
int strlen(const char *s);

int main ()
{
	char s[MAXELEM], ch;
	int i = 0;
	while((i<MAXELEM-1) && ((ch=getchar())!='\n'))
		s[i++] = ch;
	s[i] = '\0';
	printf("Vnesenata tekstualna niza e \"%s\".\n", s);
	RemoveTrail(s);
	printf("Tekstualnata niza po izvrsenite promeni e \"%s\".\n", s);
	return 0;
}

int strlen(const char *s)
{
	int n;
	for(n=0; *s!='\0' ;s++)
		n++;
	return n;
}

void RemoveTrail(char *str)
{
	int i;
	i = strlen(str);
	for (i--; i>=0 && isspace(str[i]); i--)
	{
	}
	str[i+1] = '\0';
}
/*so vklucuvanje na bibliotekata <string.h> moze da ne se pisuva
funkcijata int strlen(const char *s);*/
```

2. Да се напише функција што ќе одреди колку пати знак се наоѓа во даден стринг (низа од знаци). Знакот за споредување и стрингот се внесуваат од тастатура.

```
#include <stdio.h>
#define MAXELEM 50
int broiZnak(char *str, char znak);

int main ()
{
	char s[MAXELEM], znak;
	printf("Vnesete string: ");
	fflush(stdout);
	gets(s);
	printf("Vnesete znak koj treba da se bara vo vneseniot sting: \n");
	znak = getchar();
	printf("Vo vneseniot string \"%s\", znakot ", s);
	printf("\"%c\" se pojavuva vkupno %d pati.\n", znak, broiZnak(s, znak));
	return 0;
}

int broiZnak(char *str, char znak)
{
	int brojac = 0;
	while (*str !='\0') //while (*str)
	{
		brojac += (*str == znak); //brojac += (*str++ == znak);
		str++;
	}
	return brojac;
}
```

3. Да се напише функција која од дадена низа знаци ќе ги исфрли знаците почнувајќи од n-тиот во должина од k знаци.

```
#include <stdio.h>
#include <string.h>
#define MAXELEM 50

void strDelete(char *str, int poz, int dolz);

int main (){
	char s[MAXELEM];
	int poz, dolz;
	printf("Vnesete string: ");
	fflush(stdout);
	gets(s);
	printf("Vnesete od koja pozicija i kolku znaka treba da se isfrlat:\n");
	fflush(stdout);
	scanf("%d %d", &poz, &dolz);
	strDelete(s, poz, dolz);
	printf("Novo dobieniot string e: ");
	puts(s);
	return 0;
}
// ############ VERZIJA 1 ###############
void strDelete(char *str, int poz, int dolz){
	char *s = str + poz + dolz - 1, *d = str + poz-1;
	int len = strlen(str);
	for (; *s && ((poz+dolz-1)<len); *d++ = *s++);
```

```
	*d = 0;
}
// ############ VERZIJA 2 ###############
void strDelete(char *str, int poz, int dolz){
	if ((poz+dolz-1)<strlen(str)) strcpy(str+poz-1, str+poz+dolz-1);
	else *(str+poz-1) = 0;
}
```

4. Да се напишат функција која ќе врати подниза од зададена текстуална низа определена со позицијата и должината што како параметри се вчитуваат од тастатура. Поднизата започнува од карактерот што се наоѓа на соодветната позиција во текстуалната низа броено од лево.

```
#include <stdio.h>
#include <string.h>
#define MAXELEM 50

int main (){
	char s[MAXELEM], dest[MAXELEM];
	int poz, dolz;
	printf("Vnesete string: ");
	fflush(stdout);
	gets(s);
	printf("Vnesete pozicija i broj na znaci za podnizata: \n");
	fflush(stdout);
	scanf("%d %d", &poz, &dolz);
	if (poz<=strlen(s))
	{
		strncpy(dest, s+poz-1, dolz);
		*(dest+dolz)=0;
		printf("Novo dobienata tekstualna niza e: ");
		puts(dest);
	}
	else {
		printf("Vnesena e nevalidna pozicija za podnizata, vnesenata ");
		printf("niza ima samo %d znaci.\n", strlen(s));
	}
	return 0;
}
```

5. Да се напише функција која во стринг што и се предава како влезен параметар ќе ги промени малите букви во големи и обратно и ќе ги отфрли сите цифри.

```
#include <stdio.h>
#include <ctype.h>
#define MAXELEM 50

void promeniString(char *str);

int main (){
	char s[MAXELEM];
	printf("Vnesete string: ");
	fflush(stdout);
	gets(s);
	promeniString(s);
	printf("Novo dobieniot string e: ");
	puts(s);
	return 0;
}

void promeniString(char *str){
	int i = 0, j = 0;
	while (str[i] != '\0')
	{
		if (!(isdigit(str[i])))
		{
			if (islower(str[i])) str[j] = toupper(str[i]);
			else if (isupper(str[i])) str[j] = tolower(str[i]);
			else str[j] = str[i];
			j++;
		}
		i++;
	}
	str[j] = '\0';
}
```

6. Да се напише програма која за дадена низа од знаци (внесена од тастатура) ќе провери дали е палиндром (исто се чита и од десно на лево и од лево на десно). Од внесениот збор, пред проверката дали е палиндром, да се исфрлат празните места и да не се прави разлика помеѓу мали и големи букви.

```
#include <stdio.h>
#include <ctype.h>
#include <string.h>
#define MAXELEM 50

void promeniString(char *str);
int Palindrom(char *str);

int main (){
	char s[MAXELEM];
	printf("Vnesete string: ");
	fflush(stdout);
	gets(s);
	printf("Vneseniot string e ");
	printf("\"%s\" i %s palindrom.\n", s, Palindrom(s)? "E": "NE E");
	return 0;
}
```

```
void promeniString(char *str)
{
	char *a = str, *b = str;
	while (*a)
	{
		if (isalpha(*a))
		{
			if (isupper(*a)) *b = tolower(*a);
			else *b = *a;
			b++;
		}
		a++;
	}
	*b = 0;
}
// ############ VERZIJA 1 ###############
int Palindrom(char *str)
{
	int palin = 1, i, len;
	promeniString(str);
	len = strlen(str);
	for (i=0; i<len/2; i++)
		if (*(str+i) != *(str+len-1-i)) palin = 0;
	return palin;
}
// ############ VERZIJA 2 ###############
int Palindrom(char *str)
{
	char *a , *b;
	promeniString(str);
	a = str;
	b = str + strlen(str) - 1;
	while (a<b && *a++==*b--);
	return (a>=b);
}
```

Основни функции за работа со стрингови

Сите функции за работа со стрингови се сместени во:
```
#include <string.h>
```

Најчесто користени функции:

`char *strcpy (const char *dest,const char *src)` – Копирање на вториот стринг во првиот

`int strcmp(const char *string1,const char *string2)` – Споредба на два стрингови. Враќа 0 ако се исти.

`char *strerror(int errnum)` – Враќа опис во вид на низа од знаци врз основна бројот на грешка што се појавил

`int strlen(const char *string)` – Враќа должина на даден стринг.

`char *strncat(const char *string1, char *string2, size_t n)` –Додава n знаци од string2 на string1.

`int strncmp(const char *string1, char *string2, size_t n)` – Прави споредба на првите n знаци од двата стрингови.

`char *strncpy(const char *string1,const char *string2, size_t n)` --Копирање на првите n знаци од string2 во string1.

`int strcasecmp(const char *s1, const char *s2)` – Споредба на два стрингови без притоа да се прави разлика помеѓу мали и големи букви.

Употреба на функциите:

```
char *str1 = "ZRAVO";
char *str2;
int dolzina;
dolzina = strlen("ZDRAVO"); /* dolzina = 6 */
(void) strcpy(str2,str1);
```

Треба да се земе во предвид дека и strcat() и strcpy() враќаат копија од првиот аргумент, кој впрочем е и резултатот од операцијата (дестинациски стринг). Исто така треба да се земе во предвид дека редоследот на аргументите е таков што прво доаѓа дестинацискиот (резултантниот) стринг а потоа изворниот стринг. Овој редослед често знае да се помеша.

Функцијата strcmp() прави лексичка (алфабетска) споредба на два влезни стрингови и враќа:

- Вредност помала од нула доколку првиот стринг е лексички помал од вториот стринг
- Вредност еднаква на нула доколку првиот стринг е еднаков со вториот стринг
- Вредност поголема од нула доколку првиот стринг е лексички поголем од вториот стринг

```
char *str1 = "ZDRAVO";
char *str2;
int dolzina = 3;
(void) strncpy(str2,str1, dolzina); /* str2 = "ZDR" */
```

str2 **НЕ ЗАВРШУВА СО NULL ТЕРМИНАТОР!**

Пребарување во стрингови:

`char *strchr(const char *string, int c)` – Враќа покажувач кон првото појавување на знакот c во стрингот.

`char *strrchr(const char *string, int c)` – Враќа покажувач кон последното појавување на знакот c во стрингот.

`char *strstr(const char *s1, const char *s2)` – Бара подстринг во даден стринг т.е. го пронаоѓа првото појавување на s2 во s1.

- Доколку не го најде, враќа null покажувач.
- Доколку го најде, враќа покажувач кон првиот знак од подстрингот во s1

`char *strpbrk(const char *s1, const char *s2)` – Враќа покажувач кон првото појавување на било кој знак од S2 во рамки на S1.

`size_t strspn(const char *s1, const char *s2)` – Враќа колку знаци се поклопуваат од почетокот на s1 во рамки на s2.

`size_t strcspn(const char *s1, const char *s2)` – Спротивно од strspn (колку знаци не се поклопуваат).

`char *strtok(char *s1, const char *s2)` – Го дели стрингот s1 во секвенца од стрингови и како разделувач (делимитер) се земаат еден или повеќе знаци стрингот s2.

Споредување на знаци:

`int isalnum(int c)` – Точно ако е алфа-нумерички знак.
`int isalpha(int c)` – Точно ако е буква.
`int isascii(int c)` – Проверува дали е ASCII вредност .
`int iscntrl(int c)` – Проверува дали е контролен знак (Enter, Backspace,NullChar,…)
`int isdigit(int c)` – Проверува дали е бројка
`int isgraph(int c)` – Проверува дали е графички знак.
`int islower(int c)` – Проверува дали е мала буква
`int isprint(int c)` – Проверува дали е знак кој може да се испечати
`int ispunct(int c)` – Проверува дали е некој интерпункциски знак.
`int isspace(int c)` – Проверува дали е празно место.
`int isupper(int c)` – Проверува дали е голема буква.
`int isxdigit(int c)` – Проверува дали е хексадецимален број

Конверзија (претворување) на знаци:

`int toascii(int c)` – Прави претворување на с во ASCII .
`int tolower(int c)` – Претворување на с во мала буква.
`int toupper(int c)` – Претворување на с во голема буква.