Univerzitet u Sarajevu
Elektrotehnički fakultet Sarajevo
Osnove računarstva

Laboratorijska vježba 9

Cilj ove vježbe je utvrđivanje znanja iz oblasti rad sa stringovima. Takođe, kroz rad sa stringovima utvrđuje se znanje pokazivačke aritmetike. Dato je više zadataka iz oblasti stringova kako bi studenti imali priliku dodatno vježbati kod kuće.

I UVODNI DIO VJEŽBE

1. Unesite sljedeći program sa predavanja:

```
#include <stdio.h>

void unesi(char niz[], int velicina) {
	char znak = getchar();
	if (znak == '\n') znak=getchar();
	int i = 0;
	while (i < velicina-1 && znak != '\n') {
		niz[i] = znak;
		i++;
		znak = getchar();
	}
	niz[i]='\0';
}

int prebroji(char* s, char znak) {
	int broj = 0;
	while (*s != '\0') {
		if (*s == znak)
			broj++;
		s++;
	}
	return broj;
}

int main() {
	char a[80],c;
	printf ("\nUnesite jednu rijec do 80 znakova (ENTER za kraj): ");
	unesi(a, 80);
	printf ("\nKoji znak treba prebrojati: ");
	scanf ("%c", &c);
	printf ("\nBroj znakova %c je: %d\n",c,prebroji(a,c));
	return 0;
}
```

a) Analizirajte program.
b) Kompajlirajte program.
c) Testirajte program tako što ćete tri puta unositi različite riječi i znakove koji se prebrojavaju.

II ZADATAK ZA PROVJERU RAZUMIJEVANJA UVODNOG ZADATKA

2. U prethodnom programu dodajte funkciju `samoglasnici` koja će prebrojati broj samoglasnika u zadatoj riječi. Prototip funkcije `samoglasnici` je:

```
int samoglasnici(char *s);
```

Ova funkcija treba uzimati u obzir i velika i mala slova. Zatim pri kraju main() funkcije umetnite poziv funkcije `samoglasnici` tako da program na standardnom izlazu pored dužine riječi i broja pojavljivanja nekog znaka ispiše i broj samoglasnika u unesenoj riječi.

Primjer ulaza i izlaza sada treba biti:

```
Unesite jednu rijec do 80 znakova (ENTER za kraj):
Otorinolaringologija
Koji znak treba prebrojati: o
Broj znakova o je: 4
Broj samoglasnika je: 10
```

U ovom primjeru kod broja znakova **o** nije ubrojano početno veliko slovo O jer funkcija `prebroji` koju smo dali iznad razlikuje velika i mala slova.

III ZADACI ZA SAMOSTALNU VJEŽBU

3. Napišite funkciju sa prototipom:

```
char* kapitaliziraj(char *s);
```

koja u primljenom stringu svako početno slovo riječi koje je malo slova pretvara u veliko. Pretpostavljamo da su riječi međusobno omeđene razmacima, početkom ili krajem stringa. Funkcija treba da vraća pokazivač na početak istog stringa koji je proslijeđen funkciji kao parametar. Ovo omogućuje da se funkcija poziva lančano, npr. možete pisati:

```
char s[] = "Ovo je primjer.";
printf("%s", kapitaliziraj(s));
```

Napravite kraći testni program u kojem demonstrirate rad ove funkcije. Primjer ulaza i izlaza:

```
Unesite neki tekst: Ovo je primjer.
Ovo Je Primjer.
```

4. Napišite funkciju **max_slovo** koja prima string a vraća (veliko) slovo koje se najviše puta pojavljuje u stringu. Funkcija ne treba razlikovati velika i mala slova. Ukoliko se više slova ponavlja isti broj puta, treba vratiti najmanje takvo slovo. Znakovi koji nisu slova nas ne

zanimaju.

Primjer: Ako string glasi:
"Ovo je probni primjer."
Funkcija treba vratiti slovo O jer se ono pojavljuje tri puta u stringu a manje je od slova R koje se također pojavljuje tri puta.

5. Napišite funkciju sa prototipom:

```
int prva_rijec(char *s);
```

koja string zamjenjuje prvom riječi u stringu. Npr. ako je dat string:
"Danas je lijep dan."
funkcija ga treba pretvoriti u:
"Danas"
Funkcija vraća broj riječi u polaznom stringu (prije odsijecanja ostalih riječi).

Radi jednostavnosti zadatka, pretpostavite da je riječ bilo koji niz znakova razdvojen znakom razmak. Za vježbu možete napraviti složenija pravila za riječi uzimajući u obzir znakove interpunkcije.

6. Implementirajte funkciju *strcmp*() iz biblioteke **string.h**. Funkcija ima sljedeći prototip:

```
int uporedi(const char *s1, const char *s2);
```

Funkcija vrši abecedno poređenje dva data stringa i vraća cjelobrojnu vrijednost:

- -1 ako je prvi string abecedno ispred drugog,
- 0 ako su stringovi jednaki,
- 1 ako je prvi string abecedno iza drugog.

Ako su dva stringa različite dužine, a identični su do kraja kraćeg, smatra se da je kraći string abecedno ispred dužeg.

"Abecedni poredak" stringova je poredak koji se koristi npr. u telefonskom imeniku. Primjeri:

s1	s2	uporedi(s1,s2)	Komentar
aaaa	bbbb	-1	Slovo *a* je abecedno prije slova *b*
iijkl	iiiii	1	Slovo *j* je abecedno nakon slova *i*
aaa	aaa	0	Stringovi su jednaki
aaa	aaaa	-1	Stringovi su jednaki do dužine kraćeg stringa, ali pošto je s1 kraći smatra se da je abecedno ispred s2

Napravite kraći program pomoću kojeg možete testirati ovu funkciju.
7. Modificirajte funkciju iz prethodnog zadatka tako da prima još jedan parametar:

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

Parametar **velikamala** je logička vrijednost koja određuje da li će se pri poređenju uzimati u obzir razlika između velikih i malih slova (velika slova su abecedno ispred malih) ili ne.

Primjeri:

s1	s2	strcmpi(s1,s2,0)	strcmpi(s1,s2,1)
Sarajevo	Sarajevo	0	0
Sarajevo	sarajevo	0	-1
aaa	Aaaa	-1	1

Izmjene:

-