PiPI µMASS 2

Zadatak #1

Napisati funkciju koja prima matricu i parametre koji određuju veličinu matrice. Funkcija treba za svaki redak ispisati sumu svih njegovih elemenata. Također, funkcija mora ispisati indeks retka s najvećom sumom.

Primljena matrica:
1 2 3 4 5
0 2 4 6 8
1 3 5 7 9
3 3 3 3 3
9 1 0 2 1

Izlaz:
15
20
25
15
13
2

Rješenje

```
#include <stdio.h>

void sumaRedaka(int *matrica, int redaka, int stupaca, int maxStupaca) {
	int i, j, suma, najveci, najvecaSuma;

	for (i = 0; i < redaka; ++i) {
		suma = 0;
		for (j = 0; j < stupaca; ++j)
			suma += matrica[i * maxStupaca + j];
		/*	suma += *(matrica + i * maxStupaca + j); */

		if (i == 0 || suma > najvecaSuma) {
			najveci = i;
			najvecaSuma = suma;
		}
		printf("%d\n", suma);
	}
	printf("%d\n", najveci);
}
```

Zadatak #2

Korisnik će unijeti tri broja: prvi koji ima tri znamenke, drugi koji ima pet i treći koji ima sedam, ali ih neće odvojiti prazninama (unijet će ih zajedno kao jedan veliki broj). Ispravno pročitati i ispisati:

- prvi broj tako da zauzima pet mjesta (kako nema dovoljno znamenki višak popuniti nulama),
- treći broj treba podijeliti s 1337.13 (realno dijeljenje – `double`) i ispisati rezultat na 3 decimale,
- dobiveni rezultat treba dignuti na 11. potenciju i ispisati u znanstvenoj notaciji.

Ulaz:
123123451234567
Izlaz:
00123
923.296
4.156725e+032

Rješenje

```c
#include <stdio.h>
#include <math.h>

int main() {
	int a, b, c;
	double rezultat;

	scanf("%3d%5d%7d", &a, &b, &c);

	printf("%05d\n", a);

	rezultat = c / 1337.13;
	printf("%.3lf\n", rezultat);

	rezultat = pow(rezultat, 11);
	printf("%le\n", rezultat);

	return 0;
}
```

Zadatak #3

Napisati program koji će učitati niz znakova naredbom `gets`. Pretpostaviti da neće biti učitano više od 50 znakova. Učitani niz znakova treba prepisati u novi niz tako da se svako malo slovo prepiše kao uskličnik pa veliko slovo. Na primjer niz „a“ bi postao „!A“, a niz „IvaN“ bi postao „I!V!AN“. Ispisati novi niz.

Rješenje

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

int main() {
	char prvi[51];
	char drugi[101];
	int pP, pD;

	gets(prvi);
	for (pP = pD = 0; prvi[pP] != '\0'; ++pP) {
		if (islower(prvi[pP])) {
			drugi[pD] = '!';
			++pD;
			drugi[pD] = toupper(prvi[pP]);
			++pD;
		} else {
			drugi[pD] = prvi[pP];
			++pD;
		}
	}
	drugi[pD] = '\0';
	puts(drugi);
	return 0;
}
```

Zadatak #4

Napisati program koji će učitati niz znakova naredbom `gets`. Pretpostaviti da neće biti učitano više od 50 znakova. Program treba napraviti transformaciju obrnutu od one iz prethodnog zadatka. Svaki uskličnik nakon kojega se nalazi veliko slovo treba u novi niz prepisa kao malo slovo. Na primjer niz „!Iva“ bi postao „iva“, a niz „A!NA“, bi postao „AnA“.

Rješenje

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

int main() {
	char prvi[51];
	char drugi[101];
	int pP, pD;

	gets(prvi);
	for (pP = pD = 0; prvi[pP] != '\0'; ++pP) {
		if (prvi[pP] == '!' && isupper(prvi[pP + 1])) {
			++pP;
			drugi[pD] = tolower(prvi[pP]);
			++pD;
		} else {
			drugi[pD] = prvi[pP];
			++pD;
		}
	}
	drugi[pD] = '\0';
	puts(drugi);
	return 0;
}
```

Zadatak #5

Napisati program koji učitava dva niza znakova (nizovi će biti odvojeni razmacima i neće sadržavati više od 50 znakova). Program treba iz prvog niza izbaciti sva pojavljivanja drugog niza. Ispisati izmijenjeni niz.

Ulaz:
petar-petri-plete-petlju pe
Izlaz:
tar-tri-plete-tlju

Rješenje

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

int jePodniz(char *prvi, int pP, char *drugi);

int main() {
	char prvi[51], drugi[51];
	char novi[51];
	int pP, pN;

	scanf("%s %s", prvi, drugi);
	for (pP = pN = 0; prvi[pP] != '\0';) {
		if (jePodniz(prvi, pP, drugi))
			pP += strlen(drugi);
		else {
			novi[pN] = prvi[pP];
			++pN;
			++pP;
		}
	}
	novi[pN] = '\0';
	puts(novi);
	return 0;
}

int jePodniz(char *prvi, int pP, char *drugi) {
	int i;
	for (i = 0; drugi[i] != '\0'; ++i, ++pP) {
		if (prvi[pP] != drugi[i])
			return 0;
	}
	return 1;
}
```

Alternativno rješenje

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

int main() {
	char niz[51];
	char podniz[51];
	char *p;

	scanf("%s %s", niz, podniz);

	while((p = strstr(niz, podniz))!=0){
	/* strstr će u pokazivač p staviti prvo pojavljivanje podniza unutar
	 * niza. Ako je niz "petar" ([p][e][t][a][r][\0]), a podniz je
	 * "ta" ([t][a][\0]), tada će pokazivač p pokazivati na &niz[2].
	 */
	strcpy(p, p + strlen(podniz));
	/* funkcija strcpy(n1, n2) kopira niz n2 u niz n1. U našem slučaju
	 * pokazivač p pokazuje na &niz[2], a p + strlen(podniz) odgovara
	 * elementu niza &niz[4]. Dakle, strcpy kopira sve znakove počevši
	 * od niz[4] do kraja niza uključujući i '\0'. Prvi znak (niz[4]) se
	 * kopira u niz[2], drugi (niz[5]) u niz[3] itd.
	 */
	}
	puts(niz);
	return 0;
}
```

Zadatak #6

Napisati program koji učitava tri niza znakova (nizovi mogu sadržavati razmake). Program treba u prvom nizu svako pojavljivanje drugog niza zamijeniti s trećim. Ispisati izmijenjeni niz.

Ulaz:
petar petri plete petlju
pe
iva
Izlaz:
ivatar ivatri plete ivatlju

Rješenje

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

int jePodniz(char *prvi, int pP, char *drugi);

int main() {
	char prvi[51], drugi[51], treci[51];
	char novi[2501];
	int pP, pN;

	gets(prvi);
	gets(drugi);
	gets(treci);
	for (pP = pN = 0; prvi[pP] != '\0';) {
		if (jePodniz(prvi, pP, drugi)) {
			strcpy(novi + pN, treci);
			pP += strlen(drugi);
			pN += strlen(treci);
		} else {
			novi[pN] = prvi[pP];
			++pN;
			++pP;
		}
	}
	novi[pN] = '\0';
	puts(novi);
	return 0;
}

int jePodniz(char *prvi, int pP, char *drugi) {
	int i;
	for (i = 0; drugi[i] != '\0'; ++i, ++pP) {
		if (prvi[pP] != drugi[i])
			return 0;
	}
	return 1;
}
```

Zadatak #7

Napisati funkciju koja prima niz znakova i vraća strukturu koja sadrži:

- broj samoglasnika – cijeli broj
- omjer između ukupnog broja slova i broja suglasnika – realan broj jednostruke preciznosti
- pokazivač na drugu prazninu ili NULL ako je nema – pokazivač na znak

Napisati glavni program koji učitava niz znakova (pretpostaviti da neće biti učitano više od 50 znakova), poziva ovu funkciju ispisuje rezultate.

Ulaz:
petar petri plete petlju
Izlaz:
Samoglasnika: 8
Omjer: 1.615385
Adresa praznine: 0022FF3B

Rješenje

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

typedef struct Podaci {
	int samoglasnika;
	float omjer;
	char *drugaPraznina;
} Podaci;

int jeSuglasnik(char znak);

Podaci analiziraj(char *niz) {
	int i;
	Podaci podaci;
	podaci.samoglasnika = 0;
	podaci.drugaPraznina = NULL;

	int suglasnika = 0, slova = 0;
	int praznina = 0;

	for (i = 0; niz[i] != '\0'; ++i) {
		if (isalpha(niz[i])) {
			++slova;
			if (jeSuglasnik(niz[i]))
				++podaci.samoglasnika;
			else
				++suglasnika;
		} else if (isspace(niz[i])) {
			++praznina;
			if (praznina == 2)
				podaci.drugaPraznina = &niz[i];
		}
	}
	podaci.omjer = (float)slova / suglasnika;
	return podaci;
}
```

```c
int jeSuglasnik(char znak) {
	znak = tolower(znak);
	return znak == 'a' || znak == 'e' || znak == 'i' ||
		znak == 'o' || znak == 'u';
}


int main() {
	char niz[51];
	Podaci podaci;

	gets(niz);
	podaci = analiziraj(niz);

	printf("Samoglasnika: %d\n", podaci.samoglasnika);
	printf("Omjer: %f\n", podaci.omjer);
	printf("Adresa praznine: %p\n", podaci.drugaPraznina);
	return 0;
}
```