ELEKTROTEHNIČKI FAKULTET UNIVERZITETA U BEOGRADU BEOGRAD, 23.04.2018.

Drugi kolokvijum iz Programiranja 2

Kolokvijum traje 90 minuta

Napomene:

a) Pažljivo proučite Uputstvo pre popunjavanja Obrasca za odgovore.
b) Vrednost odgovora: tačan = **15**; netačan = **-3.75**; nevažeći (nula ili više zacrnjenih kružića) = **0.**
c) Na pitanjima se može osvojiti najviše **45** poena.
d) Zadatak nosi **55** poena.

I ZADACI

1)Napisati program na programskom jeziku C koji pomaže pri formiranju matrice za igru ukrštenih reči. Program najpre učitava dimenzije matrice i alocira dinamičku matricu navedenih dimenzija. Matrica je na početku prazna, a prazna polja se označavaju znakom **#**. Zatim korisnik učitava reči u matricu redom, sleva na desno i odozgo na dole sve dok se ne unesu nekorektne koordinate reči. Reči se unose tako što se unese sama reč, a zatim i koordinate ćelije matrice počevši od koje zadata reč treba da se upiše. Poznato je da reč nema više od 10 karaktera. Reč se u matricu upisuje samo ukoliko u zadatoj vrsti može cela da stane sleva na desno od zadatog broja kolone i ne postoji druga reč već upisana na tom mestu, u suprotnom se ignoriše. Na kraju dodavanja svih reči iz matrice ukloniti sve nepopunjene vrste i kolone. Za formiranje matrice potrebno je implementirati funkciju sa sledećim prototipom (potpisom):

```
char** ukrsteneReci(int *m, int *n);
```

Funkcija **ukrsteneReci** treba da vrati formiranu matricu (**char****). Glavni program treba da pozove funkciju za formiranje matrice, a zatim da sadržaj iste ispiše prvo po vrstama, a zatim po kolonama na standardnom izlazu. Voditi računa o ispravnom korišćenju dinamičke memorije.

Primer ulaza	Primer izlaza	
4 5 PROC 0 0 IR 0 2 ETF 1 0 SI 2 0 K 2 3 I -2 7	Po vrstama: PROC ETF# SI#K	Po kolonama: PES RTI OF# C#K

II PITANJA

1)Šta ispisuje sledeći program na programskom jeziku C?

```
#include <stdio.h>
#include <string.h>
int f2(char *s){
    int i=0;
    do i++; while(*s++);
    return i;
}
int f1(char **s, int i){
    int duz = strlen(s[i]);
    return f2(s[i])-duz;
}
```

```
int main() {
    char *str[3] = { "algoritam", "maj", "v"}, s[10];
    int i;
    for (i=2; i>0; i--){
        strcpy(s, str[i]);
        s[f1(str, 2-i)] = '\0';
        printf("%s", s);
    }
}
```

A) vma **(B)** vm C) vmaj

2)Šta ispisuje sledeći program na programskom jeziku C?

```
#include <stdio.h>
#include <stdlib.h>
int radi(int *p, int *q) {
  int *t, *r, s;
  if(q - p + 1 <= 0) return -1;
  t = calloc(q - p + 1, sizeof(*t));
  r = t; s = *p; *r = s;
  while(++p <= q) {
    r[1] = *r + *p > *p ? *r + *p : *p;
    if(r[1] > s) s = r[1];
    r++;
  }
  free(t);
  return s;
}

int main(void) {
  int niz[] = {4, -5, 7, -3, 5, -4, 8};
  int n = sizeof(niz)/sizeof(*niz);
  printf("%d", radi(niz, niz + n/2));
  printf("%d", radi(niz + n/2, niz + n - 1));
  printf("%d", radi(niz, niz + n - 1));
  return 0;
}
```

(A) 7913 B) 4713 C) 7128

3)Šta ispisuje sledeći program na programskom jeziku C?

```
#include <stdio.h>
#include <stdlib.h>
#define N 4
#define M 2
void main() {
  int *a[N], i, j, **b;
  for (i = 0; i < N; i++) {
    a[i] = malloc((i%M + 1) * sizeof(int));
    for (j = i%M; j >= 0; j--)
        (*(a + i))[j] = (M * (i + 1)) / (N - j + 1);
  }
```

```
for(b = a + N/2, i=0; i<N; i++)
    if (a[i][0]==0)
        *b[i*(i%M?-1:1)] += N;
    else
        *b[i/2*(i%M?-1:1)] <<= M;
  for (i = N - 1; i >= 0; i--) {
    for (j = 0; j < i%M + 1; j++)
        printf("%d ", a[i][j]);
    free(a[i]);
  }
}
```

A) 4 2 6 16 1 0 **(B)** 4 2 5 16 1 0 C) 4 2 5 16 1 1

4)Koja od sledećih tvrđenja su tačna za sledeći program na programskom jeziku C? Smatrati da se u program unose stringovi od najviše 10 znakova.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#define MAX 10
int main () {
   char **a, str[MAX+1]; int i, m;
   printf("m? "); scanf("%d", &m);
   a=(char**)malloc(m*sizeof(char*));
```

```
   for (i=0; i<m; i++) {
      scanf("%s", str);
      a[i]=(char*)malloc (strlen(str)+1);
      strcpy(a[i], str);
   }
   for (i=0; i<m; i++)
       printf("%s\n", a[i]);
   free(a);
   return 0;
}
```

A) Program korektno oslobađa svu zauzetu dinamičku memoriju.
(B) Program se neće završiti korektno ukoliko alokacija memorije ne uspe.
C) Naredba **strcpy(a[i], str);** se može zameniti dodelom vrednosti **a[i] = str;** uz zadržavanje istog efekta izvršavanja programa.

13S111P2, Programiranje 2, drugi kolokvijum 2017/2018.
Rešenje zadatka

Zadatak

Napisati program na programskom jeziku C koji pomaže pri formiranju matrice za igru ukrštenih reči. Program najpre učitava dimenzije matrice i alocira dinamičku matricu navedenih dimenzija. Matrica je na početku prazna, a prazna polja se označavaju znakom **#**. Zatim korisnik učitava reči u matricu redom, sleva na desno i odozgo na dole sve dok se ne unesu nekorektne koordinate reči. Reči se unose tako što se unese sama reč, a zatim i koordinate ćelije matrice počevši od koje zadata reč treba da se upiše. Poznato je da reč nema više od 10 karaktera. Reč se u matricu upisuje samo ukoliko u zadatoj vrsti može cela da stane sleva na desno od zadatog broja kolone i ne postoji druga reč već upisana na tom mestu, u suprotnom se ignoriše. Na kraju dodavanja svih reči iz matrice ukloniti sve nepopunjene vrste i kolone. Za formiranje matrice potrebno je implementirati funkciju sa sledećim prototipom (potpisom):

```
char** ukrsteneReci(int *m, int *n);
```

Funkcija **ukrsteneReci** treba da vrati formiranu matricu (**char****). Glavni program treba da pozove funkciju za formiranje matrice, a zatim da sadržaj iste ispiše prvo po vrstama, a zatim po kolonama na standardnom izlazu. Voditi računa o ispravnom korišćenju dinamičke memorije.

```c
// prepostavka je da se calloc, malloc i realloc uspesno izvrsavaju
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#define MAX_LEN 11

char** ukrsteneReci(int*, int*);
int main(){
	int m, n, i, j;
	printf("Unesite inicijalne dimenzije matrice,\
 a zatim reci sa koordinatama prvog slova.\n");
	scanf("%d%d", &m, &n);
	char** mat = ukrsteneReci(&m, &n);
	// --- ispis po vrstama ---//
	printf("Po vrstama:\n")
	for(i = 0; i < m; i++){
		for(j = 0; j < n; j++) printf("%c ", mat[i][j]);
		putchar('\n');
	}
	// --- ispis po kolonama --- //
	printf("Po kolonama:\n");
	for(i = 0; i < n; i++){
		for(j = 0; j < m; j++) printf("%c ", mat[j][i]);
		putchar('\n');
	}
	// --- oslobadjanje memorije --- //
	for(i = 0; i < m; i++) free(mat[i]); free(mat);
	return 0;
}
```

```c
char** ukrsteneReci(int *_m, int *_n){
	char **mat, rec[MAX_LEN];
	int m = *_m, n = *_n, i, j, k, t, pop;
	// --- alokacija --- //
	mat = calloc(m, sizeof(char*));
	for(i = 0; i < m; i++){
		mat[i] = malloc((n + 1) * sizeof(char));
		for(j = 0; j < n; j++) mat[i][j] = '#';
	}
	// --- popunjavanje --- //
	while(1){
		scanf("%s %d %d", rec, &i, &j);
		if(i < 0 || i >= m || j < 0 || j >= n) break;
		if(mat[i][j] != '#' || j + strlen(rec) > n) continue;
		else {
			strcpy(mat[i] + j, rec);
			mat[i][j + strlen(rec)] = '#';
		}
	}
	// --- realokacija --- //
	// * po vrstama
	for(i = 0; i < m; ){
		pop = 0;
		for(j = 0; j < n; j++) pop += (mat[i][j] == '#') ? 1 : 0;
		if(pop == n){
			free(mat[i]); m--;
			for(k = i; k < m; k++) mat[k] = mat[k + 1];
			mat = realloc(mat, m * sizeof(char*));
		}
		else i++;
	}
	// * po kolonama
	for(i = 0; i < n; ){
		pop = 0;
		for(j = 0; j < m; j++) pop += (mat[j][i] == '#') ? 1 : 0;
		if(pop == m){
			n--;
			for(t = 0; t < m; t++){
				for(k = i; k < n; k++)
					mat[t][k] = mat[t][k + 1];
				mat[t] = realloc(mat[t], n * sizeof(char));
			}
		}
		else i++;
	}
	*_n = n; *_m = m;
	return mat;
}
```