ELEKTROTEHNIČKI FAKULTET UNIVERZITETA U BEOGRADU BEOGRAD, 18.06.2014.

Ispit iz Programiranja 2

Ispit traje 150 minuta

Napomene:
a) Pažljivo proučite Uputstvo pre popunjavanja Obrasca za odgovore.
b) Vrednost odgovora: tačan = **5**; netačan = **-1.25**; nevažeći (nula ili više zacrnjenih kružića) = **0.**
c) Na pitanjima se može osvojiti najviše **25** poena, a na svakom zadataku po **20** poena.

I ZADACI

1)Napisati potprogram na progamskom jeziku C koji vrši obradu nad kvadratnom matricom celih brojeva, tako što u svaki element matrice ispod sporedne dijagonale čija je vrednost prost broj upiše vrednost njemu simetričnog elementa u odnosu na sporednu dijagonalu. Potprogram kao argumente dobija pokazivač na dinamičku matricu i dimenziju matrice. Napisati glavni program kojem se dimenzija matrice prosleđuje kao argument komandne linije. Program na osnovu zadate dimenzije napravi dinamičku kvadratnu matricu i incijalizuje elemente vrednostima unesenim sa standardnog ulaza. Program potom pozove potprogram za obradu matrice i ispiše elemente matrice nakon obrade. Voditi računa o pravilnom rukovanju korišćenim resursima.

2)U datoteci *prodavnica.txt* nalazi se spisak namirnica koje se mogu kupiti u nekoj prodavnici. U svakom redu se nalazi naziv namirnice (reč do 30 znakova), količina (ceo broj) i jedinična cena namirnice (realan broj). U datoteci *potrepstine.txt* nalazi se spisak namirnica koje treba kupiti. U svakom redu datoteke nalazi se naziv namirnice (reč do 30 znakova) i potrebna količina (ceo broj). Napisati glavni program na programskom jeziku C koji na osnovu opisanih ulaznih datoteka formira izlaznu datoteku *faktura.txt* u kojoj se nalaze potrebne namirnice koje je moguće kupiti u prodavnici. U svakom redu izlazne datoteke nalazi se naziv namirnice, količina koju je moguće kupiti u prodavnici, jedinična cena namirnice, kao i ukupna cena namirnice. Na kraju datoteke treba da se nalazi i ukupna cena svih namirnica u fakturi. Ukoliko se neka namirnica iz spiska potrepština uopšte ne prodaje u prodavnici ili je nema u dovoljnoj količini, ona se ne ubacuje u fakturu. Voditi računa o pravilnom rukovanju korišćenim resursima. Sadržaj ulaznih datoteka je dozvoljeno pročitati samo jednom, a maksimalan broj redova u datotekama, kao ni raspored stavki po redovima, nije u napred poznat. Primer datoteka:

prodavnica.txt	**potrepstine.txt**	**faktura.txt**
kupus 100 30.5 ulje 15 122.8 pivo 50 62.5	pivo 20 jabuke 30 ulje 25	pivo 20 62.5 1250.00 1250.00

II PITANJA

1)Koje od ponuđenih tvrdnji su tačne u skladu sa standardom programskog jezika C?
A) `int (*p)[](int *a);` je potpuno ispravna definicija promenljive `p`.
B) Ako su date definicije: `struct flasa {char boja[10]; float zapremina;}a; struct boca {char boja[10]; float zapremina;}b;` tada je dozvoljena dodela a=b;.
(C) Upisom u jedno polje unije moguće je promeniti vrednost drugog polja te unije.

2)Šta treba da stoji na mestu `#####` da bi funkcija `ubaci` ispravno ubacivala broj u dvostruko ulančanu listu koja je sortirana neopadajuće? Lista treba da ostane sortirana i nakon ubacivanja.

```
typedef struct elem {
  int b; struct elem *sl, *pr; } Elem;
typedef struct { Elem *first, *last;} List;
void ubaci(List *lst, int broj) {
  Elem *novi= malloc(sizeof(Elem)), *tek;
  novi->b = broj;
  if (lst->first == NULL) {
    lst->first = lst->last = novi;
    novi->sl = novi->pr = NULL;
    return; }
  for (tek=lst->last; tek; tek=tek->pr)
    if (tek->b < broj) break;
  if (tek != NULL) {
    novi->pr = tek;
    novi->sl = tek->sl;
    if (tek->sl) tek->sl->pr = novi;
    else lst->last = novi;
    tek->sl = novi;
  } else { ##### }
}
```

(A)
```
novi->sl = lst->first;
novi->pr = NULL;
lst->first->pr = novi;
lst->first = novi;
```

B)
```
novi->pr = lst->last;
novi->sl = NULL;
lst->last = novi;
```

C)
```
novi->sl = lst->first;
lst->first->pr = novi;
novi->pr=lst->first->pr;
lst->first = novi;
```

3)Šta ispisuje sledeći program na programskom jeziku C, ako je sadržaj argumenata komandne linije: **`nenenenedadada 3 x x x x`**?

```
#include <stdio.h>
#include <stdlib.h>
char* rpc(int num, char* str, char ch,
          char tr){
   int i=0;
   for(; *str && i<num; str++, i++);
   for(; *str && *str!=ch; str++);
   if(*str) *str = tr;
   return (*str) ? str : NULL;
}
void main(int argc, char** argv){
   char* str=argv[1], ch = *argv[1];
   int num=atoi(argv[2]), rem=argc-2;
   while(--rem){
      char* res = rpc(num, str, ch,
                      argv[argc-rem][0]);
      if(res) putchar(*res);
   }
   printf(" %s\n", str);
}
```

A) **`xx nenexexedada`** B) **`xxxx xexexexedada`** C) **`xx xexenenedada`**

4)Šta ispisuje sledeći program na programskom jeziku C?

```
#include <stdio.h>
#define X(a) sizeof(a)/sizeof(int)
typedef int (*FuncPtr)(int*);

int f1(int *p) { return *p++; }
int f2(int *q) { return (*(q+1))++ + *q; }
int f3(int *r) { return ++r[-2]; }
void main() {
   FuncPtr f[] = {f1, f3, f2, f1};
   int a[] = {2, 3, 5, 7, 9}, i = X(a);
   while (--i > 0)
       a[i-1] = f[i-1](a+i);
   for (i=0; i<5; printf("%d", a[i++]));
}
```

A) 3210980 **(B)** 3318910 C) 3259710

5)Šta treba da stoji na mestu `#####` da bi se znakovi stringa unetog preko standardnog ulaza ispisali po sledećem redosledu: poslednji karakter stringa, prvi karakter stringa, pretposlednji karakter stringa, drugi karakter stringa itd., ali tako da se svaki znak stringa ispiše tačno jednom?

```
void main() {  char string[10], *p, *k; scanf("%s", string); p = k = string;
  while (*(k+1)) k++;  ##### }
```

A)
```
while (p < k) {
      printf("%c", *k--);
      if (k != p)
        printf("%c", *p++);}
```

B)
```
for (; p < k; p++, k--){
      printf("%c", *k);
      printf("%c", *p);}
```

(C)
```
while ( p<=k ) {
if (k != p)
  printf("%c", *k--);
printf("%c", *p++);}
```

6)Šta ispisuje sledeći program na programskom jeziku C?

```
#include <stdio.h>
#include <stdlib.h>
typedef struct {int a, b;} S1;
typedef struct { S1* ps1; int c;
      union {int* d; int* e;} u; } S2;
void main() {
  S1 s1; S2 s2, s22; int a = 6, b = 8;
  s1.a = 1; s1.b = 2;
  s2.ps1 = &s1; s2.c = 3; s2.u.d = &a;
  s2.u.e = &b; s22 = s2; s2.ps1->b = 5;
  *s2.u.e = 10;
  printf("%d %d %d %d %d", s22.ps1->a,
    s22.ps1->b, s22.c, *s22.u.d, *s22.u.e);
}
```

A) 1 2 3 6 8 **(B)** 1 5 3 10 10 C) 1 5 3 6 10