RAČUNALNIŠTVO I
1998/99

Kolokvij

9. rožnika 1999

Kolokvij sestoji iz petih nalog, katerih skupno število točk je **100** in **20** za dodatno nalogo. Vrednosti posameznih nalog so naznačne v spodnji tabeli.

1.	naloga: zapisi študentov v polju	__ / 20
2.	naloga: urejanje – prvič	__ / 20
3.	naloga: zapisi študentov v seznamu	__ / 24
4.	naloga: urejanje – drugič	__ / 16
5.	naloga: zapisi študentov kot predmeti	__ / 20
	dodatno vprašanje	__ / +20
SKUPNO		__ / 100
	dodatno vprašanje	__ / +20

OPOMBA: Nekatere naloge so posebej označene. Pri teh nalogah lahko „kupite“ odgovor. Cena odgovora so vse možne točke pri tistem vprašanju, oziroma podvprašanju. Zato se za nakup odločajte preudarno in le po potrebi!
Čas pisanja je 2 uri (120 minut).
Na kolokviju je dovoljena uporaba zapiskov a nikakor „dobrososedske“ pomoči.
Predno pričnete pisati, spodaj vpišite svoje ime in priimek ter se podpišite.

Ime in Priimek Podpis

(tiskano) (študent)

Veliko uspeha!

1. NALOGA: ..... _ / 20

Rdeča nit tega kolokvija so zapisi s podatki študentov in obdelovanje le-teh. Podatki študentov, ki nas zanimajo so:

- ime študenta: do 10 znakov dolg niz črk, med katerimi katerimi *ni šumnikov*.
- priimek študenta: do 20 znakov dolg niz črk, med katerimi *ni šumnikov*.
- letnik, v katerem je študent: celo število.

VPRAŠANJE: ..... _ / 7

Napišite definicijo tipa `tStudent`, ki popisuje enega študenta!

KUPLJIVO: _____

```
typedef struct {
  char ime[10];
  char priimek[20];
  int  letnik;
} tStudent;
```

VPRAŠANJE: ..... _ / 6

Napišite definicijo spremenljivke `letnik`, ki popisuje 100 študentov v letniku. Spremenljivka naj bo vektor ustreznega tipa in ustrezne velikosti.

```
#define maxLetnik 100

tStudent letnik[maxLetnik];
```

VPRAŠANJE: ..... _ / 7

Kakšna je razlika med *tipom* in *spremenljivko*?

Spremenljivka je določen predstavnik danega tipa, tip pa je množica vseh možnih spremenljivk – opisuje lastnosti vseh spremenljivk. S tipom ne moremo ničesar početi, spremenljivki pa lahko priredimo vrednost ali jo preberemo.

2. NALOGA: ..... _ / 20

Eno od opravil v uporabnikovem programu je izpis študentov v letniku po abecednem redu; no ja, skoraj v abecednem redu, saj morajo biti najprej izpisani vsi tisti, katerih priimek se prična na „A“, nato oni na „B“, sledijo na „C“, na „D“ itd. Pozor, ker ni šumnikov, ni nikogar na „Č“!

Napišite podprogram, ki bo *uredil* (*ne* izpisal) študente v letniku. Za opis posameznega študenta in letnika uporabite rešitev prejšnje naloge.

VPRAŠANJE: ..... _ / 4

Kateri so vhodni in kateri izhodni parametri?

Vhodni parameter je letnik. Izhodni parameter je tudi letnik.

VPRAŠANJE: ..... _ / 4

Kakšen je podpis funkcije?

KUPLJIVO: _____

(skupaj s prejšnjim vprašanjem)

```
void Uredi (tStudent * letnik);
                                /* ali */
void Uredi (tStudent letnik[maxLetnik]);
```

VPRAŠANJE: ..... _ / 12

Napišite samo funkcijo!

```
void Uredi (tStudent letnik[maxLetnik])
{ int i, j;                     /* stevca */
  tStudent zac;                 /* zacasna sprem. */

  for (i= 0; i < maxLetnik; i++)
    for (j= 0; j < i; j++)
      if (letnik[i].priimek[0] < letnik[j].priimek[0]) {
        zac= letnik[i];         /* zamenjamo */
        letnik[i]= letnik[j]; letnik[j]= zac;
      }
} /* Uredi */
```

3. NALOGA: ..... _ / 24

V prejšnjih nalogah smo predstavili letnik kot vektor študentov. V tej nalogi pa ga bomo predstavili kot *seznam študentov*.

VPRAŠANJE: ..... _ / 6

Napišite definicijo glave (elementa) seznama in samega seznama študentov kot rekurzivne podatkovne strukture!

KUPLJIVO: _____

```
typedef struct pod {
  tStudent    glava;
  struct pod* podSeznam;
} t_ElementSeznama;

typedef t_ElementSeznama* tSeznam;
```

Na vajah smo se srečali s seznamom celih števil. Od tam uporabite naslednje funkcije ter jih ustrezno predelajte, da bodo uporabne za delo s seznamom študentov:

VPRAŠANJE: ..... _ / 6

`vrednost_glave`:

KUPLJIVO: _____

```
tStudent vrednost_glave(tSeznam letnik)
{ return seznam->glava; }
```

VPRAŠANJE: ..... _ / 6

naredi_seznam:

KUPLJIVO: _____

```
tSeznam naredi_seznam(tStudent student)
{ tSeznam seznam;

  seznam = (tSeznam)malloc(sizeof(t_ElementSeznama));
  seznam->glava = student;
  seznam->podSeznam = NULL;
  return seznam;
} /* naredi_seznam */
```

VPRAŠANJE: ..... _ / 6

Dodaj_zacetek:

KUPLJIVO: _____

```
tSeznam Dodaj_zacetek(tStudent student, tSeznam seznam)
{ tSeznam seznam1;

  seznam1 = naredi_seznam(student);
  seznam1->podSeznam = seznam;
  return seznam1;
} /* Dodaj_zacetek */
```

4. NALOGA: ..... _ / 16

Sedaj, ko imamo letnik predstavljen kot seznam študentov, bomo še za takšno predstavitev napisali funkcijo, ki bo ustrezno preuredila seznam študentov, da bodo na začetku študenti, katerih priimek se prična na „A“, nato na „B“ itd.

VPRAŠANJE: ..... _ / 4

Kakšen je podpis funkcije?

```
tSeznam Uredi (tSeznam letnik);
```

VPRAŠANJE: ..... _ / 12

Napišite samo funkcijo!

NAMIG: Poglejte si, katere funkcije smo napisali na vajah za delo s seznamom celih števil in morda kakšno od njih ustrezno priredite, da bo počela, kar zahteva vprašanje. Pri pisanju uporabite tudi funkcije, ki ste jih napisali pri prejšnjem vprašanju.

```
tSeznam izloci_glavo(tStudent * student, tSeznam seznam)
{ tSeznam novi_seznam;

  *student = seznam->glava;
  novi_seznam = seznam->podSeznam;
  (void) free(seznam);
  return novi_seznam;
} /* izloci_glavo */

tSeznam vrini(tSeznam letnik, tStudent student)
{ tStudent glava;               /* prvi v seznamu */

  if (seznam == NULL)           /* ce je seznam prazen */
    return naredi_seznam(student);

  glava= vrednost_glave(letnik);
  if(glava.priimek[0] >= student.priimek[0]){
    letnik = Dodaj_zacetek(student, letnik);
    return letnik;
  } else {
    letnik->podSeznam = vrini(letnik->podSeznam, student);
    return letnik;
  }
} /* vrini */

tSeznam Uredi (tSeznam letnik)
{ tSeznam urLetnik= NULL;       /* urejeni letnik je prazen */
  tStudent student;             /* obdelovani student */

  while (letnik != NULL) {
    letnik= izloci_glavo(&student, letnik);
    urLetnik= vrini(urLetnik, student);
  }
  return urLetnik;
} /* Uredi */
```

5. NALOGA: ..... _ / 20

V zadnji nalogi si bomo ogledali še kako rokovati s študenti in letnikom, ki je predstavljen v predmetni obliki. V tej nalogi bomo definirali razred študetov in nato, za dodatno nalogo, še razred letnikov.

VPRAŠANJE: ..... _ / 9

Napišite definicijo razreda študentov, od katerih ima vsak vse lastnosti kot prej (ime, priimek in letnik) ter dodatno lastnost (funkcijo), da se zna izpisati v obliki:

Priimek, Ime: letnik

```
class cStudent {
public:
  tStudent podatki;

  void Izpisi()
  { printf("%s, %s: %d\n",
           podatki.priimek, podatki.ime, podatki.letnik);
  } // Izpisi
}; // cStudent
```

VPRAŠANJE: ..... _ / 4

Za kateri lastnosti (funkciji) smo rekli na vajah, da ju ima vsak razred predmetov? Napišite ju še za razred študentov!

To sta funkciji, ki ustvarita in uničita predmet iz razreda – *kreator*, *destruktor*. Primera za razred `cStudent` sta:

```
cStudent()
{ strcpy(podatki.priimek, "");
  strcpy(podatki.ime, "");
  podatki.letnik= 0;
} // cStudent

~cStudent() { }
```

VPRAŠANJE: ..... _ / 7

Kakšna je razlika med predmetom in razredom?

Razred popisuje množico predmetov, oziroma definira lastnosti, ki jih imajo predmeti. Predmet pa je udejanjen predstavnik razreda, ki „živi“, hrani resnične podatke in zna nekaj narediti.

VPRAŠANJE: ..... _ / +20

DODATNA: Napišite definicijo razreda letnik. Vsak letnik sestoji iz do 100 študentov (predmetov iz razreda študentov). Poleg tega:

- si zna dodati novega študenta v letnik s funkcijo (lastnostjo) `DodajSi`;
- se zna urediti po že prej opisanem abecednem vrstnem redu s funkcijo (lastnostjo) `UrediSe`.
- se zna izpisati s funkcijo (lastnostjo) `IzpisiSe`.

Napišite vsaj glave funkcij v razredu!

```
class cLetnik {
public:
  cStudent studenti[maxLetnik];
  int vLetniku;                  // studentov v letniku

  cLetnik() { vLetniku= 0 }

  void DodajSi(tStudent student)
  { if (vLetniku < maxLetnik) {
      studenti[vLetniku].podatki= student;
      vLetniku++;
    }
  } // DodajSi

  void UrediSe()
  { int i, j;                    // stevca
    cStudent zac;                // zacasna sprem.

    for (i= 0; i < vLetniku; i++)
      for (j= 0; j < i; j++)
        if (studenti[i].podatki.priimek[0] <
            studenti[j].podatki.priimek[0]) {
          zac= studenti[i];      // zamenjamo
          studenti[i]= studenti[j]; studenti[j]= zac;
        }
  } // UrediSe

  void IzpisiSe()
  { int i;                       // stevec
    for (i= 0; i < vLetniku; i++)
      studenti[i].Izpisi();
  } // IzpisiSe

} // cLetnik
```