PROGRAMIRANJE ZA ELEKTROTEHNIKE II

Rešitev 1. kolokvija, 17. 4. 2008

Tu so samo rešitve oziroma napotki, kje jih najdete. Za vsebino vprašanj glejte, prosim, kolokvij.

1.
a) Glejte na primer prosojnice.

b) Deklaracija spremenljivke samo pove (prevajalniku), da nekje obstaja definicija spremenljivke, da se lahko uporablja v programu. Napiše se z uporabo ključne besede `extern`. Primer deklaracije spremenljivke `x1`:

```
extern int x1;
```

Definicija pa povzroči, da se rezervira za spremenljivko toliko prostora, kolikor je potrebnega glede na njen tip, lahko pa spremenljivki tudi že dodeli začetno vrednost (tj. jo inicializira). Primer definicije spremenljivk `x1` in `x2`:

```
int x1, x2 = 5;
```

c) Globalna definicija spremenljivke se nahaja v programu izven vseh funkcij (tudi main()), lokalna pa znotraj funkcije. To tudi pomeni, da se spremenljivka, definirana s prvo (tj. globalna spremenljivka), lahko uporablja povsod za definicijo, lokalno definirana (tj. lokalna) spremenljivka pa se lahko uporablja samo v funkciji oz. natančneje, v bloku, kjer je definirana.

d) Vzemimo, da imamo izsek programa:

```
int stevilo1 = 1, stevilo2 = 2;

void zamenjaj(int x, int y) {
    int pomoz = x;
    x = y;
    y = pomoz;
}

int main() {
    zamenjaj(stevilo1, stevilo2);
…
}
```

Ker argumenta funkcije nimata pri tipu znaka &, pomeni, da je zamenjaj(stevilo1, stevilo2) klic funkcije s prenosom obeh argumentov po vrednosti. Vrednost stevila1 se

uporabi kot vrednost formalnega argumenta x, vrednost stevila2 pa kot vrednost formalnega argumenta y pri izvajanju funkcije. Funkcija povzroči, da x postane enak 2, y pa 1, stevilo1 in stevilo2 pa ostaneta nespremenjena.

Če damo pri tipu formalnih argumentov x in y znak &:

```
void zamenjaj(int& x, int& y) {
    int pomoz = x;
    x = y;
    y = pomoz;
}
```

pa pomeni, da bomo imeli ob klicu funkcije prenašanje obeh argumentov po referenci. To pomeni, da ob klicu zamenjaj(stevilo1, stevilo2) x postane drugo ime za stevilo1, y pa drugo ime za stevilo2. Ker zamenjamo vrednosti x in y, se zamenjata tudi vrednosti stevila1 in stevila2. Funkcija ob klicu ne le uporabi vrednosti stevila1 in stevila2, ampak ju lahko tudi spremeni, v nasem primeru zamenja njuni vrednosti.

Lahko bi rekli, da je pri prenosu po vrednosti dejanski argument funkcije samo vhodni argument, pri prenosu po referenci pa vhodno-izhodni argument, ker mu funkcija lahko spremeni vrednost, tj. v njem vrne novo vrednost (glejte tudi prvo funkcijo v nalogi 3).

e) Glejte na primer prosojnice: odločili bi se lahko za opis treh izmed osnovnih tipov celoštevilčni, realni, naštevni, logični, znakovni. Strukturirana sta na primer polje in struktura.

V letošnjem kolokviju bodo bolj konkretna vprašanja in ne »opišite z enim ali dvema stavkoma«.

2.
a) V polje z imenom `polje` lahko shranimo 10 elementov.
b) Vrednost spremenljivke `rezultat` pri prvem izpisu na zaslon je 18.
c) Vrednost spremenljivke `rezultat` pri drugem izpisu na zaslon je 11.

3.
Iz zadnjega stavka naloge izgleda, da mora prva funkcija vsoto vrniti in ne kar izpisati na zaslon. Ker je funkcija tipa `void`, jo lahko vrača v argumentu ali pa jo priredi globalni spremenljivki. Napišimo funkcijo `sestej` z enim formalnim argumentom s prenosom po referenci. Funkcijo kličimo z dejanskim argumentom `n`. Ob klicu naj ima vrednost `n` iz besedila, funkcija pa naj v njem vrne vsoto. Vzemimo, da prvih `n` lihih celih števil pomeni liha števila med 1 in `n`.

Pri drugi funkciji vzemimo, da ne smemo uporabiti niti `<cstring>` niti `<string>`, sicer bi dolžino niza lahko dobili s funkcijo `strlen()` v prvem primeru ali s `size()`

ali `length()`v drugem. Vzemimo, da se niz vpiše v glavnem programu in da z njim kličemo funkcijo, vrne pa njegovo dolžino tipa `unsigned short`.

Program bi lahko bil na primer tak:

```
#include <iostream>
using namespace std;

void sestej(unsigned int& n) {
   unsigned int vsota = 0;
   for (unsigned int i = 1; i <= n; i++) {
      if (i % 2 == 1) {
         vsota += i * i * i;
      }
   }
   n = vsota;
}

unsigned short vrniDolzino(char polje[]) {
   unsigned short int i = 0;
   while (polje[i] != '\0') {
      i++;
   }
   return i;
}

int main() {
   unsigned int n;
   cout << ˝Vpisi n: ˝;
   cin >> n;
   sestej(n);
   cout << n << endl;
   cout << ˝Vpisi niz: ˝;
   char niz[30];
   cin >> niz;
   cout << vrniDolzino(niz) << endl;
   return 0;
}
```

Če bi hoteli na primer prebrati niz, ki bi vseboval presledke, bi lahko napisali naslednjo funkcijo main():

```
int main() {
   unsigned int n;
   char niz[30];
   cout << ˝Vpisi n: ˝;
```

```
    cin >> n;
    sestej(n);
    cout << n << endl;
    cin.ignore(30, '\n');
    cout << ˝Vpisi niz: ˝;
    cin.getline(niz, 30, '\n');
    cout << vrniDolzino(niz) << endl;
    return 0;
}
```

Sami lahko poskusite napisati tudi program z globalno spremenljivko `vsota`.