PROGRAMIRANJE ZA ELEKTROTEHNIKE II

Dva primera nalog iz snovi za 2. test z rešitvami

1. Implementirajte delujoč program (funkcije main ni treba, ker je napisana že spodaj), ki bo omogočal enostavno delo z datotekami. V programu definirajte strukturo z imenom `Statistika`, ki ima tri elemente:

- število samoglasnikov,
- število vseh vrstic,
- število kopiranih vrstic.

V strukturi implementirajte funkcijo za inicializacijo in izpis podatkov na zaslon. Dodatno v programu implementirajte funkcijo `kopiraj_brez_komentarja`. Funkcija naj omogoča kopiranje vsebine vhodne tekstovne datoteke v izhodno. Med kopiranjem naj funkcija preveri, ali vrstica predstavlja komentar (tj. ali se vrstica začne z znakoma `//`). Če vrstica predstavlja komentar, naj se vrstica ne prekopira v izhodno datoteko. Med kopiranjem naj funkcija dodatno naredi analizo o:

- številu samoglasnikov v vhodni datoteki,
- številu vrstic vhodne datoteke,
- številu vrstic, kopiranih v izhodno datoteko.

Če uspe uspešno odpreti obe datoteki, naj vrne vrednost `true`, sicer pa `false`. Funkcija naj ima tri argumente. Prvi predstavlja ime vhodne datoteke, drugi ime izhodne datoteke, tretji argument pa strukturo, v katero se shranjujejo podatki o analizi.

Primer uporabe strukture `Statistika` in funkcije `kopiraj_brez_komentarja`:

```
int main() {
   Statistika s;
   s.init(); // inicializacija podatkov
   if (kopiraj_brez_komentarja(″vhod.txt″, ″izhod.txt″, s)
      s.izpis(); // izpis statistike;
   else
      cout << ″Tezava pri odpiranju vsaj ene datoteke.″
           << endl;
   return 0;
}
```

Primer rešitve (npr. da bo vse v eni datoteki):

```
#include <iostream>
#include <string>
#include <fstream>

using namespace std;

struct Statistika {
   int stSamogl;
```

```
   int stVrstic;
   int stKopVrstic;

   void init() {
      stSamogl = 0;
      stVrstic = 0;
      stKopVrstic = 0;
   }
   void izpis() {
      cout << ″Stevilo samoglasnikov: ″ << stSamogl << endl;
      cout << ″Stevilo vrstic: ″ << stVrstic << endl;
      cout << ″Stevilo kopiranih vrstic: ″ << stKopVrstic
           << endl;
   }
};


bool kopiraj_brez_komentarja(char vhdat[], char izhdat[],
Statistika& stat)
{
   string vrstica;

   ofstream izhodna_datoteka(izhdat);
   if (!izhodna_datoteka.good())
      return false;
   ifstream vhodna_datoteka(vhdat);
   if (!vhodna_datoteka.good())
      return false;
   while (!vhodna_datoteka.eof()) {
      getline(vhodna_datoteka, vrstica, '\n');
      stat.stVrstic++;
      for (int i = 0; i < vrstica.size(); i++) {
         if (toupper(vrstica[i]) == 'A' ||
             toupper(vrstica[i]) == 'E' ||
             toupper(vrstica[i]) == 'I' ||
             toupper(vrstica[i]) == 'O' ||
             toupper(vrstica[i]) == 'U') {
            stat.stSamogl++;
         }
         if (vrstica[0] != '/' && vrtica[1] != '/ ') {
            stat.stKopVrstic++;
            izhodna_datoteka << vrstica << endl;
         }
      }
   }
   izhodna_datoteka.close();
   vhodna_datoteka.close();
   return true;
}
```

2. Implementirajte razred `Zaposlen`, ki ima atribut `ime`. Implementirajte konstruktor z enim argumentom (tj. za nastavitev imena) in metodo za izpis.

Iz razreda `Zaposlen` izpeljite razred `RednoZaposlen` (uporabite javno izpeljavo). Izpeljani razred naj ima shranjene podatke o ceni ure in o številu mesečno opravljenih ur. Privzeta vrednost cene ene ure je 12 EUR, število mesečno opravljenih ur pa je 176. Implementirajte samo potrebne konstruktorje ter funkciji za izračun izplačila in izpis na zaslon.

S pomočjo funkcije main prikažite uporabo razreda `RednoZaposlen`.

Primer rešitve: Ker v besedilu ni zahtevano drugače, spet vzemimo, da naj bo vse v eni datoteki. Recimo, da je potreben konstruktor za podrazred tisti, ki tvori redno zaposlenega s privzeto vrednostjo cene ene ure in števila opravljenih ur, ter tak, ki tvori redno zaposlenega s podano vrednostjo cene in števila ur.

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

class Zaposlen {
private:
   char ime[40];
public:
   Zaposlen(char* imezap) {
      strcpy(ime, imezap);
   }
   void izpis() {
      cout << ″Zaposleni: ″ ime << endl;
   }
};

class RednoZaposlen: public Zaposlen {
private:
   float cenaUre;
   float opravUre;
public:
   RednoZaposlen(char* ime): Zaposlen(ime), cenaUre(12),
                             opravUre(176) {}
   RednoZaposlen(char* ime, float cenaUre, float opravUre):
      Zaposlen(ime) {
      this->cenaUre = cenaUre;
      this->opravUre = opravUre;
   }
   float izplacilo() {
      return cenaUre * opravUre;
   }
   void izpis() {
      Zaposlen::izpis();
      cout << ″Cena ure (EUR): ″ << cenaUre << endl;
      cout << ″Stevilo ur: ″ << opravUre << endl;
```

```
      cout << ″Izplacilo (EUR): ″ << izplacilo() << endl;
   }
};

int main() {
   RednoZaposlen zaposlena(″Alenka Kranjc″);
   zaposlena.izpis();
   RednoZaposlen zaposleni(″Trajko Pecov″, 12.5, 221.55);
   zaposleni.izpis();
   return 0;
}
```