Kolokwium Testowe 1, gr. A, Języki Programowania, 5.12.2013

Treść zadania

Należy stworzyć klasę Owoc zawierającą trzy pola składowe:

- nazwa(char *) //zakładamy, że nazwa nie przekroczy 25 znaków, ale pamięć powinna być alokowana dynamicznie
- ilosc (liczba całkowita)
- cena (liczba zmiennoprzecinkowa)

Oraz zestaw metod: metody zwracające i ustawiające składniki klasy (metody typu “get” i “set”), konstruktor domyślny, konstruktor z 3 parametrami oraz konstruktor kopiujący i destruktor. Należy zewnętrzną funkcję Wypisz() która przyjmuje obiekt Owoc i wypisuje go na ekran . Ponadto należy stworzyć metodę SetAll ustawiającą wszystkie trzy parametry jednocześnie (jeśli podamy dwa parametry, wtedy cena=1.0, a jeśli tylko jeden parametr, wtedy cena=1.0, a ilosc = 0), oraz przeciążyć operator przekierowania: (“<<”).

Klasy powinny być skonstruowane w taki sposób, by działały z podanym niżej programem
(program można ściągnąć ze strony: http://www.if.pw.edu.pl/~majanik/data/JP/2013/kolokwium/mainowoc.cpp)

```
#include <iostream>
#include <cstring>
#include "owoc.h"

using namespace std;

int main()
{
  char* gruszka = new char[8];   strcpy(gruszka,"gruszka");
  char* jablko= new char[7];     strcpy(jablko,"jablko");
  char* banan= new char[6];      strcpy(banan,"banan");

  Owoc o1;
  o1.SetNazwa(gruszka);
  cout<<"Nazwa: "<<o1.GetNazwa()<<" Ilosc: "<<o1.GetIlosc()<<" Cena: "<<o1.GetCena()<<endl;
  delete gruszka;
  Wypisz(o1);

  Owoc *o2 = new Owoc(jablko,10,1.1);
  cout<<"Nazwa: "<<o2->GetNazwa()<<" Ilosc: "<<o2->GetIlosc()<<" Cena: "<<o2->GetCena()<<endl;
  cout<<o1<<", "<<*o2<<endl;
  delete o2;

  Owoc o3(o1); Wypisz(o3);
  o3.SetAll(banan);
  cout<<o3<<endl;
  o3.SetAll(banan,2);
  cout<<o3<<endl;
  o3.SetAll(banan,2,4.5);
  return 0;
}
```

Co po uruchomieniu skutkowało by pojawieniem się na ekranie:

```
$ ./program
Nazwa: gruszka Ilosc: 0 Cena: 0
Owoc: (gruszka, 0, 0)
Nazwa: jablko Ilosc: 10 Cena: 1.1
Owoc: (gruszka, 0, 0), Owoc: (jablko, 10, 1.1)
Jestem w destruktorze...
Owoc: (gruszka, 0, 0)
Owoc: (banan, 0, 1)
Owoc: (banan, 2, 1)
Jestem w destruktorze...
Jestem w destruktorze...
```

Plik nagłówkowy powinien zostać otoczony strukturą „`ifndef`”. Program powinien być napisany w 3 plikach (owoc.cpp, owoc.h oraz program.cpp z funkcją main()) oraz powinien się kompilować bez żadnych ostrzeżeń (flaga -Wall).

Program powinien być kompilowany przy pomocy komendy „`make`” (należy stworzyć odpowiedni Makefile!).

Program, oraz wszystkie pliki należy wysłać przed końcem zajęć na adres majanik@if.pw.edu.pl.

Kolokwium Testowe 1, gr. B, Języki Programowania, 5.12.2013

Treść zadania
Należy stworzyć klasę vector, reprezentującą wektor trójwymiarowy i zawierającą pola składowe:

- x, y, z (double)
- notatka (std::string)
- pole statyczne ilosc (int)

Oraz zestaw metod: metodę zwracającą i ustawiającą notatkę (metodę typu "get" i "set"), konstruktor domyślny, konstruktor z 3 parametrami (x,y,z), konstruktor kopiujący i destruktor. Konstruktory prócz przypisywania wartości odpowiednim składnikom powinny zwiększać wartość pola ilość o 1, destruktor powinien zmniejszać wartość tego pola.
Należy również stworzyć metodę statyczną zwracającą wartość pola ilosc, oraz przeciążyć operator + (zwracający vector, będący sumą wektorową, a notatka powinna zostać ustawiona na "wynik sumy"). Oraz metodę Dlugosc() zwracającą długość wektora. Ponadto należy stworzyć metodę SetXYZ ustawiającą trzy parametry (x, y, z) jednocześnie oraz stworzyć metodę WypiszWektor() która wypisuje na ekran informację o składnikach danego obiektu. Do tego napisać funkcję WypiszWektory przyjmującą tablicę wektorów oraz ich liczbę, służącą do wypisywania owej podanej tablicy na ekran.

Klasy powinny być skonstruowane w taki sposób, by działały z podanym niżej programem :
(program można ściągnąć ze strony: http://www.if.pw.edu.pl/~majanik/data/JP/2013/kolokwium/mainvector.cpp)

```
#include <iostream>
#include "vector.h"
using namespace std;

int main()
{
  vector v;
  v.WypiszWektor(); cout<<endl;
  vector *v2 = new vector(2,3,4);
  v2->WypiszWektor(); cout<<endl;
  v.SetXYZ(1,2,3);
  v.SetNotatka("wektor pierwszy");
  v2->SetNotatka("wektor drugi");
  v.WypiszWektor(); cout<<endl;
  v2->WypiszWektor(); cout<<endl;

  vector v3 = v;
  v3 = v3 + *v2;
  v3.WypiszWektor(); cout<<", dlugosc wektora v3: "<<v3.Dlugosc()<<endl;
  cout<<"Liczba wektorow: "<<vector::GetIlosc()<<endl;
  delete v2;
  cout<<"Liczba wektorow: "<<vector::GetIlosc()<<endl;

  vector tabV[2]; tabV[0].SetXYZ(6,6,6); tabV[1].SetXYZ(2,2,2);
  WypiszWektory(tabV,2);
  return 0;
}
```

Co po uruchomieniu skutkowało by pojawieniem się na ekranie:

```
$ ./program
(0,0,0)
(2,3,4)
(1,2,3) wektor pierwszy
(2,3,4) wektor drugi
(3,5,7) wynik sumy, dlugosc wektora v3: 9.11043
Liczba wektorow: 3
Liczba wektorow: 2
(6,6,6)
(2,2,2)
```

Plik nagłówkowy powinien zostać otoczony strukturą „`ifndef`". Program powinien być napisany w 3 plikach (vector.cpp, vector.h oraz program.cpp z funkcją main()) oraz powinien się kompilować bez żadnych ostrzeżeń (flaga -Wall).
Program powinien być kompilowany przy pomocy komendy „`make`" (należy stworzyć odpowiedni Makefile!).

Program należy wysłać przed końcem zajęć na adres majanik@if.pw.edu.pl