

Języki Programowania+, wtorek 25.10.2016

Zadanie 3

Zadanie polega na napisaniu klasy `samolot`, która ma:

1. pola prywatne:

- **char* nazwiskoPilota** – nazwisko pilota
- **float* loty** - tablica przechowuje długość wykonanych lotów w km
- **int nLot**- liczba elementów w tablicy loty.

2. metody publiczne:

- **void setNazwiskoPilota(char* nazwisko)** – nadaje polu `nazwiskoPilota` wartość parametru `nazwisko`
- **void info()** - wyświetla informacje samolocie, tzn. nazwisko pilota oraz wszystkie elementy w tablicy loty
- **void setLot(int nrLotu, float dlugoscLotu)** – metoda wpisuje wartość `dlugoscLotu` do tablicy `loty` na pozycji `nrLotu`
- **void setNLot(int ileLotow)** – ustawia wartość pola `nLot` na wartość `ileLotow`. Do klasy można dodawać inne pola i metody, jeśli uznają to Państwo za użyteczne.

Dodatkowo należy napisać funkcję globalną:

void znajdzMinMax(samolot* tab, int Nsamolot, float& min, float& max)

zaprzyjażnioną z klasą `samolot`. Funkcja wyszukuje najmniejszą i największą wartość długości lotów wykonanych przez samoloty podane w tablicy `tab`. Parametry funkcji:

- **Samolot* tab** – tablica obiektów klasy `samolot`, rozmiar tej tablicy to **Nsamolot**
- **float& min** – zmienna, w której przez referencję przekazywana jest najmniejsza długość lotu dla wszystkich samolotów w tablicy `tab`.
- **float& max** – zmienna, w której przez referencję przekazywana jest największa długość lotu dla wszystkich samolotów w tablicy `tab`.

Potrzebne funkcje:

`int strlen(char*)` - zwraca długość ciągu znaków (z biblioteki `<string.h>`)

`void strcpy(char* a, char* b)` – kopiuje ciąg znaków `b` do `a` (z biblioteki `<string.h>`)

`int *tab = new int[N];` – dynamiczna alokacja pamięci dla tablicy `int` o długości `N`.

Na drugiej stronie można znaleźć opis funkcji głównej `main()`.

Punktacja:

- implementacja klasy – 1 pkt
 - implementacja funkcji zaprzyjażnionej – 1 pkt
 - znalezienie długości najkrótszego i najdłuższego lotu wykonanego przez samoloty w tablicy `samoloty` i wyświetlenie tych wartości na ekranie – 1 pkt.
 - utworzenie tablicy `samoloty`, pętla `switch-case` – 2 pkt
- Program powinien zostać napisany z podziałem na 3 pliki: 2 pliki dla klasy `Samolot` oraz oddzielny plik dla funkcji `main`.

W programie głównym (funkcji main) należy:

1. Utworzyć tablicę obiektów (bądź wskaźników na obiekty) klasy `samolot` (np. o nazwie `samoloty`) o rozmiarze 5 (będzie to maksymalna ilość przechowywanych samolotów). Stworzyć również zmienną odpowiedzialną za zliczanie samolotów zapisanych w tablicy.
2. Następnie w pętli `switch-case` program powinien umożliwić:
 - [1] dodanie nowego samolotu (kolejno podawane z klawiatury powinny być: nazwisko pilota, ilość lotów, oraz loty jeden po drugim)
 - [2] wypisanie wszystkich dotychczas dodanych samolotów
 - [3] znalezienie najdłuższego i najkrótszego lotu w tablicy samoloty i wyświetlenie ich na ekranie (funkcja `znajdzMinMax`)
 - [0] wyjście z programu

Dodatkowe:

Będziemy chcieli stworzyć (zamiast tablicy) listę obiektów typu `samolot`. Do tego celu wykorzystamy listę podwójnie linkowaną (double-linked list) z tzw. Standardowej Biblioteki Szablonów języka C++ (STL).

Użycie listy z biblioteki standardowej pokazuje przykład poniżej:

```
#include <string>
#include <list>
#include <iostream>

using namespace std;

class A
{
public:
    string fCiagZnakow;
    int fLiczba;
};

int main()
{
    A a1;
    A a2;

    list<A> lista; //lista obiektow klasy A

    //wstawianie elementow na koniec listy
    lista.push_back(a1);
    lista.push_back(a2);

    //iterowanie po elementach listy
    list<A>::iterator i;
    for(i=lista.begin(); i!=lista.end(); i++)
    {
        cout<<(*i).fCiagZnakow<<endl;
        cout<<(*i).fLiczba<<endl;
        cout<<endl;
    }

    //usuwanie elementu z konca listy
    lista.pop_back();

    //zwrocenie elementu z konca listy
    A a3 = lista.back();

    return 0;
}
```

Należy zakomentować uprzednio napisaną część kodu opierającą się na tablicy i zaimplementować funkcjonalności [1], [2] i [0] używając listy z STL.