Laboratorium Informatyki II
Ćwiczenie 9

Przyjaźń

Przyjaźń jest metodą nadania zewnętrznym elementom takim jak funkcje i klasy dostępu do elementów prywatnych klasy. Przyjaźń można nadać tylko w kodzie klasy umieszczając kod deklaracji funkcji lub klasy poprzedzony słowem kluczowym `friend`. Klasa może być zaprzyjaźniona z dowolną liczbą funkcji i klas. Deklarację przyjaźni w klasie można zdefiniować w dowolnym typie dostępu (`public` i `private`).

W poniższym przykładzie pokazano implementację przyjaźni klasy do funkcji i innej klasy. W kodzie zaimplementowano klasę `dane` przechowującą tablicę `n` elementową i jej nazwę. Klasa jest zaprzyjaźniona z funkcją `max()` zwracającą maksymalną wartość z tablicy klasy, oraz z klasą tablica. Klasa ta z wykorzystaniem klasy `dane` pozwala na przetwarzania tablicy 2D.

```
#include <iostream>
#include <random>
class tablica;
class dane
{
private:
    std::string nazwa;
    int size;
    int *tab;
    friend int max(dane);
    friend int max(tablica);
    friend class tablica;
public:
    void setDane(int P, int K)
    {
        std::random_device genrator;
        std::default_random_engine silnik(genrator());
        std::uniform_int_distribution<int> rozklad(P,K);
        for(int i=0;i<size;i++)
            tab[i] = rozklad(silnik);
    }
    void drukuj()
    {
        std::cout << nazwa << " --> [";
        for(int i=0;i<size-1;i++)
            std::cout << tab[i] << ",";
        std::cout << tab[size-1] << "]" << std::endl;
```

```
    }
    dane(){};
    dane(std::string nazwa, int size)
    {
        this->size = size;
        this->nazwa = nazwa;
        tab = new int[size];
    }
    ~dane()
    {
        delete [] tab;
    }
};
class tablica
{
private:
    int size;
    dane *nTab;
    friend int max(tablica);
public:
    tablica(int size)
    {
        this->size = size;
        nTab = new dane[size];
        for(int i=0;i<size;i++)
        {
            nTab[i].nazwa = std::to_string(i+1);
            nTab[i].tab = new int[size];
            nTab[i].size = this->size;
            nTab[i].setDane(0,100*size);
        }
    }
    ~tablica()
    {
        delete [] nTab;
    }
    void drukuj()
    {
        for(int i=0;i<size;i++)
            nTab[i].drukuj();
    }
};
int max(dane A)
{
    int M=A.tab[0];
    for(int i=1;i<A.size;i++)
        if(A.tab[i]>M) M=A.tab[i];
    return M;
}
int max(tablica A)
```

```
{
    int M=max(A.nTab[0]);
    for(int i=1;i<A.size;i++){
        int m = max(A.nTab[i]);
        if(m>M) M=m;
    }
    return M;
}
int main()
{
    dane A("PIERWSZY",5);
    A.setDane(10,20);
    A.drukuj();
    std::cout << "MAX:-->" << max(A) << std::endl;
    tablica B(8);
    B.drukuj();
    std::cout << "MAX:-->" << max(B) << std::endl;
}
```

Zadania

W oparciu o informacje wykładowe i treść niniejszej instrukcji zmodyfikować program kolekcjonujący informacje o trasie wycieczki i wyświetlający jej statystykę z wykorzystaniem przyjaźni z funkcją i klasą.