Metodyka i Techniki Programowania II

C++: Biblioteka STL (*Standard Template Library*) ciąg dalszy

---

Cele ćwiczenia – Poznanie i umiejętność użycia elementów biblioteki STL.
– Powtórzenie wiadomości.

Bibliografia: materiały z wykładu, specyfikacja biblioteki STL.
https://www.cplusplus.com/reference/stl/
https://www.cplusplus.com/reference/algorithm/

STL jest to biblioteka kontenerów, algorytmów i innych powszechnie używanych udogodnień programistycznych.
Algorytmy określają standardowe operacje wykonywane na zawartości np. kontenerów, np. *min(), max(), reverse (), sort(), count()*.
Kontenery służą do przechowywania obiektów np. *vector*, *list*.
Iteratory to uogólnienie zwykłych wskaźników, np *.begin()* zwraca iterator wskazujący na pierwszy element kontenera, *.end()* zwraca iterator reprezentujący pozycję za ostatnim elementem kontenera.

Zadania do wykonania w ramach laboratorium

Zadanie 1

1.1 Napisz funkcje, która z dwóch liczb *int* zwróci liczbę mniejszą.

Zadanie 2

2.1 Przeanalizuj działanie poniższego kodu.

```
#include <iostream>                                    //kod Zadanie 2
#include <algorithm>

using namespace std;

int main() {
    cout << "max(1,2)==" << max(1, 2) << '\n';
    cout << "max(2,1)==" << max(2, 1) << '\n';
    cout << "max('a','z')==" << max('a', 'z') << '\n';
    cout << "max(3.14,2.73)==" << max(3.14, 2.73) << '\n';

    cout << "min(1,2)==" << min(1, 2) << '\n';
    cout << "min(2,1)==" << min(2, 1) << '\n';
    cout << "min('a','z')==" << min('a', 'z') << '\n';
    cout << "min(3.14,2.72)==" << min(3.14, 2.72) << '\n';
    return 0;
}
```

2.2 Porównaj *min() z* napisaną przez siebie funkcją z zadania 1. Jaki jest wniosek?

Zadanie 3

3.1 Stwórz kontener *vector* o nazwie *vec* do przechowywania zmiennych typu *float*.
3.2 Umieść w *vec* 10 losowych liczb.
3.3 Posortuj elementy *vec* od najmniejszej do największej wartości użyj do tego odpowiedniego algorytmu. Wyświetl wynik.

Zadanie 4

4.1 Dokonaj analizy poniższego kodu.

```
#include <iostream>                                    //kod Zadanie 4
#include <list>
#include <algorithm>

using namespace std;

int main() {
    list<int> lst;
    lst.push_back(100);
    lst.push_back(40);
    lst.push_back(200);
    lst.push_back(50);
    int n = count(lst.begin(), lst.end(), 100);
    cout << "n wynosi " << n << endl;
    return 0;
}
```

4.2 Zlicz wystąpienia liczby 50 w *lst*. Nie rób tego ręcznie.

4.3 Odwróć kolejność elementów *lst* i wyświetl wynik.

4.4 Jaki jest maksymalny rozmiar *lst*? Wyświetl wynik.

Zadanie 5

5.1. Napisz implementacje klasy *Time*, której definicja podana jest poniżej.

```
class Time {                                           //kod Zadanie 5
private:
    int hours_;
    int minutes_;
public:
    Time();
    Time(int h, int m);
    void addMin(int m);
    void addHr(int h);
    void reset(int h, int m);
    void show();
};
```

5.2 Napisz program testujący działanie poszczególnych metod klasy *Time*.

5.3 Napisz implementację metody *sum*, której definicja podana jest poniżej. Dodaj odpowiedni kod do napisanego już programu oraz przetestuj działanie tej metody.

```
    Time sum(const Time &t) const;          //kod Zadanie 5 cd.
```

KONIEC 😊