Metodyka i Techniki Programowania II

C++: Biblioteka STL (Standard Template Library)

Autorzy instrukcji: mgr inż. Artur Kos, dr inż. Robert Chodorek, dr inż. Zbigniew Hulicki

Biblioteka STL (Standard Template Library)

Biblioteka STL – materiały z wykładu, specyfikacja biblioteki.

Zadanie 1

1. Uruchom i przeanalizuj kod (zamieszczony poniżej):

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

int main() {
   vector<int> v(10);

   for (size_t i=0; i<10; ++i)
   {
      v[i] = i;
      cout << v[i] << endl;
   }
}
```

2. Dokonaj modyfikacji kodu z punktu 1 tak aby rozmiar wektora mógł być dynamiczny.
3. Wypisz ilość elementów w wektorze.
4. Usuń elementy od 3 do 7 i wyświetl wynik.
5. Usuń ostatni element wektora i wyświetl wynik.
6. Wstaw na początek wektora liczbę 102 i wyświetl wynik.
7. Wstaw na koniec wektora liczbę 110011 i wyświetl wynik.
8. Następnie kod umieść **W SWOJEJ GAŁĘZI** w repozytorium zdalnym

Zadanie 2

Prosta kolejka typu FIFO (ang. First In First Out) umożliwia wstawianie elementów na końcu i pobieranie elementów z początku kolejki. Wykorzystując dwie metody kolejki **queue** z biblioteki STL: **push** i **pop**, które służą do dodawania danych do kolejki i pobierania danych z kolejki (z jednoczesnym ich usuwaniem):

1. Napisz program wstawiający do kolejki liczby naturalne od 1 do 12. Wyświetl zawartość kolejki na ekran.
2. Podaj długość kolejki używając metody **size**.
3. Usuń z kolejki 5 elementów. Wyświetl zawartość kolejki na ekran i podaj jej długość.
4. Uzupełnić program o funkcje ograniczającą długość kolejki do zadanej wartości np. 6 i usuwającej elementy przekraczające ograniczenie. W przypadku gdy przekroczone jest ograniczenie funkcja ma wyświetlać informację, ze kolejka jest pełna a element jest usuwany.
5. Wykorzystując opracowaną funkcję dodać do kolejki 4 elementy.
6. Następnie kod umieść **W SWOJEJ GAŁĘZI** w repozytorium zdalnym

Zadanie 3

1. Uruchom i przeanalizuj działanie kodu:

```
#include <iostream>
#include <queue> // std::queue
using namespace std;

class petent {
public:
   int numer_oczekujacego = 0;
   int godz;
};

int main()
{
   std::queue<petent> eit;
   petent p; // tworzenie obiektu klasy petent, jego kopie będą dodawane do kolejki

   eit.push(p);
   petent pomocniczy = eit.back(); // tworzenie pomocniczego obiektu klasy petent, aby można
                                   //było do niego przypisać referencję zwracaną przez metodę back()
   cout << "numer ostatniego petenta = " << pomocniczy.numer_oczekujacego << " rozmiar kolejki =  " << eit.size() << endl;
   // alterantywa
   cout << "numer ostatniego petenta = " << eit.back().numer_oczekujacego << " rozmiar kolejki =  " << eit.size() << endl;

   cout << "Liczba oczekujacych:   " << eit.size() - 1 << endl;
}
```

Zmodyfikuj kod tak, aby jego działanie symulowało system obsługi kolejki studentów/petentów.

2. Dodaj funkcję która:

- dodaje nowy element do kolejki
- zwiększa unikalny numer oczekującego

3. Dodaj funkcję która:

- usuwa najstarszy element z kolejki
- powoduje wyświetlenie komunikatu o aktualnej liczbie oczekujących (liczba oczekujących = rozmiar kolejki -1)

4. Dodaj funkcję która:

- bada czy rozmiar kolejki nie przekroczył określonej liczby; jeśli tak, to ma ona uniemożliwiać dalsze dodawanie petentów
- powoduje wyświetlenie komunikatu o zawieszeniu przyjmowania petentów, jeśli rozmiar kolejki przekroczył określoną liczbę

5. Następnie kod umieść **W SWOJEJ GAŁĘZI** w repozytorium zdalnym