Міністерство освіти і науки України
Львівський національний університет імені Івана Франка
Факультет прикладної математики та інформатики

Кафедра дискретного аналізу та інтелектуальних систем

Лабораторна робота №7

ЧЕРГА ТА ЧЕРГА З ПРІОРИТЕТОМ

з курсу “Алгоритми та структури даних”

Виконав:
студент групи ПМІ-13
Лук’янчук Денис
Євгенійович

Львів – 2024

Черга з пріоритетами — це структура даних, що призначена для обслуговування множини елементів, кожний з яких додатково має "пріоритет", пов'язаний з ним. У пріоритетній черзі першим обслуговується елемент, який має найвищий пріоритет, відповідно елемент, що має найнижчий пріоритет буде обслугований останнім. У деяких реалізаціях, якщо два елементи мають однаковий пріоритет, вони подаються відповідно до порядку**.**

Алгоритм виконання:

1. Створення класу Queue:
   - Оголошення приватного вектора `elements`, що містить пари (значення, пріоритет).
2. Реалізація методів класу Queue:
   - enqueue(int val, int priority): Додає елемент до черги з вказаним значенням та пріоритетом, після чого сортує елементи за пріоритетом.
   - dequeue(): Видаляє перший елемент з черги, якщо вона не пуста.
   - front(): Повертає значення першого елемента черги без видалення, якщо черга не пуста.
   - isEmpty(): Перевіряє, чи є черга порожньою.
   - print(): Виводить на екран елементи черги.
   - size(): Повертає кількість елементів у черзі.
3. Основна функція main():
   - Створення об'єкту priority типу Queue.
   - Додавання елементів з різними пріоритетами до черги priority.
   - Виведення елементів черги та їх пріоритетів на кожній ітерації циклу.
   - Виведення значення першого елемента черги, видалення його та повторення до тих пір, поки черга не стане порожньою.

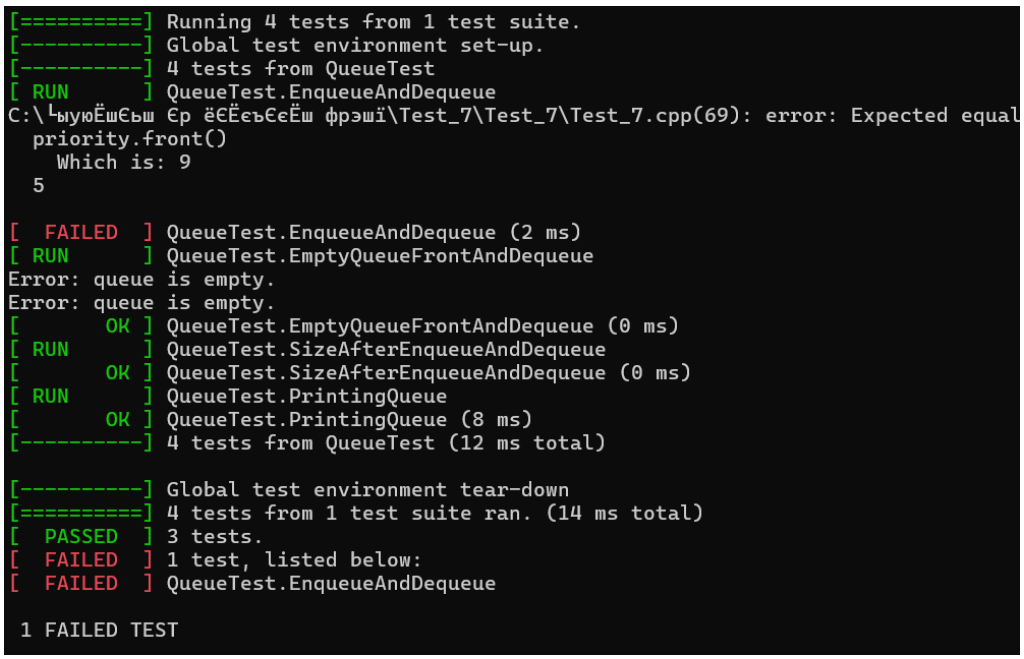
Приклад
Дано: (7, 4), (9, 1), (11, 2), (1, 5), (4, 3), (3, 1) — де перше число це елемент черги, а друге це його пріоритет.

```
Iteration: 0
Queue elements: (9, 1) (3, 1) (11, 2) (4, 3) (7, 4) (1, 5)
Front element: 9

Iteration: 1
Queue elements: (3, 1) (11, 2) (4, 3) (7, 4) (1, 5)
Front element: 3

Iteration: 2
Queue elements: (11, 2) (4, 3) (7, 4) (1, 5)
Front element: 11

Iteration: 3
Queue elements: (4, 3) (7, 4) (1, 5)
Front element: 4

Iteration: 4
Queue elements: (7, 4) (1, 5)
Front element: 7

Iteration: 5
Queue elements: (1, 5)
Front element: 1
Queue is empty
```

Приклад Unit-тесту(*без помилок*)

```
[==========] Running 4 tests from 1 test suite.
[----------] Global test environment set-up.
[----------] 4 tests from QueueTest
[ RUN      ] QueueTest.EnqueueAndDequeue
[       OK ] QueueTest.EnqueueAndDequeue (0 ms)
[ RUN      ] QueueTest.EmptyQueueFrontAndDequeue
Error: queue is empty.
Error: queue is empty.
[       OK ] QueueTest.EmptyQueueFrontAndDequeue (0 ms)
[ RUN      ] QueueTest.SizeAfterEnqueueAndDequeue
[       OK ] QueueTest.SizeAfterEnqueueAndDequeue (0 ms)
[ RUN      ] QueueTest.PrintingQueue
[       OK ] QueueTest.PrintingQueue (14 ms)
[----------] 4 tests from QueueTest (17 ms total)

[----------] Global test environment tear-down
[==========] 4 tests from 1 test suite ran. (20 ms total)
[  PASSED  ] 4 tests.
```

Приклад Unit-тесту(*з помилкою*)



Висновок: У даному коді реалізовано просту пріоритетну чергу на основі вектора з використанням стандартних бібліотек C++. Кожен елемент черги має своє значення та пріоритет, і вони сортуються за зростанням пріоритету при кожному додаванні нового елемента. Основні методи, які реалізовані в класі черги, включають додавання елементів з вказаним значенням та пріоритетом, видалення першого елемента, отримання значення першого елемента без видалення, перевірку на порожність та виведення елементів черги. Цей код демонструє основний механізм роботи пріоритетної черги та дозволяє вивести на екран послідовність додавання та видалення елементів з урахуванням їх пріоритетів. Він може бути використаний як основа для подальших розширень та використань, де потрібно керувати елементами в черзі з урахуванням їх пріоритетів.