

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

Кафедра програмування

Лабораторна робота №7  
**ЧЕРГА ТА ЧЕРГА З ПРІОРИТЕТОМ**  
з курсу “Алгоритми та структури даних”

Виконав:  
Студент групи ПМІ-12  
Бенько Володимир Сергійович

Львів – 2023

## Черга

Черга – лінійна, динамічна структура даних, що працює за принципом «перший прийшов — перший пішов» (FIFO — first in, first out). У черги є голова та хвіст. Елемент, що додається до черги, опиняється в її хвості. Елемент, що видаляється з черги, знаходиться в її голові.

Реалізована списком черга містить приватну структуру Item, яка містить значення та вказівник на наступний елемент черги, вказівники на голову та хвіст, довжину черги та набір публічних методів:

- IsEmpty() - повертає істину, якщо черга порожня
- GetSize() - повертає довжину черги
- Head() - повертає значення з голови черги, кидає underflow\_exception якщо черга порожня
- Enqueue(T val) - додає елемент в кінець (хвіст) черги
- Dequeue() - видаляє елемент з початку (голови) черги, повертає результат в вигляді логічного значення
- Clear() - видаляє всі елементи з черги

### Складність роботи методів:

- IsEmpty:  $O(1)$
- GetSize:  $O(1)$
- Head:  $O(1)$
- Enqueue:  $O(1)$
- Dequeue:  $O(1)$
- Clear:  $O(n)$

**Просторова складність черги:**  $O(n)$

## Черга з пріоритетом

Черга з пріоритетами — це нелінійна, динамічна 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.

Реалізована деревом черга з пріоритетом містить приватну структуру Node, яка містить значення вершини, посилання на правого та лівого синів та методи для обходу дерева; також вона містить вказівник на корінь дерева, лічильник елементів в черзі та набір публічних методів:

- `IsEmpty()` - повертає істину, якщо черга порожня
- `GetCount()` - повертає кількість елементів в черзі
- `PeakMin()` - повертає значення з найменшим пріоритетом в черзі, кидає `underflow_exception` якщо черга порожня
- `PopMin()` - видаляє елемент з найменшим пріоритетом з черги, повертає результат в вигляді логічного значення
- `PeakMax()` - повертає значення з найбільшим пріоритетом в черзі, кидає `underflow_exception` якщо черга порожня
- `PopMax()` - видаляє елемент з найбільшим пріоритетом з черги, повертає результат в вигляді логічного значення
- `Enqueue(T val)` - додає елемент в чергу
- `Clear()` - видаляє всі елементи з черги

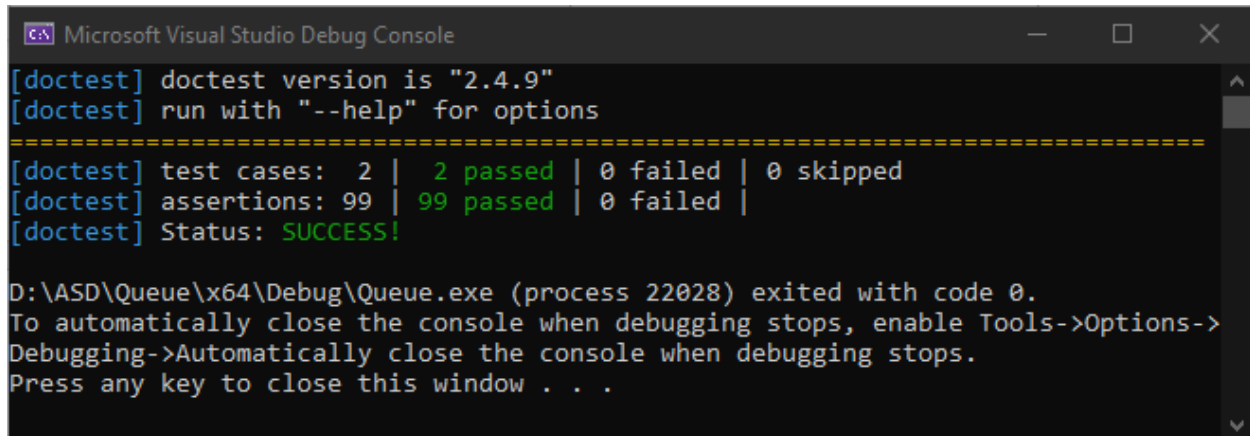
### Складність роботи методів:

- `IsEmpty`:  $O(1)$
- `GetSize`:  $O(1)$
- `PeakMin`:  $O(\log_2 n)$
- `PopMin`:  $O(\log_2 n)$
- `PeakMax`:  $O(\log_2 n)$
- `PopMax`:  $O(\log_2 n)$
- `Enqueue`:  $O(\log_2 n)$
- `Clear`:  $O(n)$

**Просторова складність черги з пріоритетом:**  $O(n)$

### Приклад:

Щоб переконатись, що всі функції працюють правильно, в програмі написані юніт-тести. Усі вони проходять успішно:

A screenshot of the Microsoft Visual Studio Debug Console window. The window title is "Microsoft Visual Studio Debug Console". The output shows doctest version "2.4.9", instructions to run with "--help", a separator line of equals signs, test results for 2 cases (2 passed, 0 failed, 0 skipped) and 99 assertions (99 passed, 0 failed), and a final status of "SUCCESS!". Below this, it states the process exited with code 0 and provides instructions on how to automatically close the console when debugging stops.

```
Microsoft Visual Studio Debug Console

[doctest] doctest version is "2.4.9"
[doctest] run with "--help" for options
=====
[doctest] test cases:  2 |  2 passed | 0 failed | 0 skipped
[doctest] assertions: 99 | 99 passed | 0 failed |
[doctest] Status: SUCCESS!

D:\ASD\Queue\x64\Debug\Queue.exe (process 22028) exited with code 0.
To automatically close the console when debugging stops, enable Tools->Options->
Debugging->Automatically close the console when debugging stops.
Press any key to close this window . . .
```

### Висновок:

Черга використовується, коли елементи не мають внутрішнього порядку або коли важливо зберігати послідовність їх додавання, а черга з пріоритетом використовується тоді, коли елементи мають відмінності у своїй важливості або терміновості. Знання про ці структури даних допоможе нам ефективно організувати наші проекти та забезпечити правильну обробку даних в порядку їх важливості.