МІНІСТЕРСТВО ОСВІТИ І НАУКИ, МОЛОДІ ТА СПОРТУ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»
ФІЗИКО-ТЕХНІЧНИЙ ІНСТИТУТ
Кафедра Інформаційної Безпеки

Практикум з Алгоритмів та структур даних

Лабораторна робота №4
Структури даних: стеки, черги

Мета роботи:

отримати навики роботи зі стеком та чергою,
реалізованими у вигляді:
одновимірного масиву та зв'язного лінійного списку.

Виконав:
студент II курсу
групи ФБ-01
Сахній Н.Р.

Київ 2022

Виконання лабораторної роботи

На максимальний бал реалізувати чергу двома способами:

/*
Як можна буде помітити у демонстрації виконання програми та у прикладених нижче кодах, було реалізовано чергу за її основним принципом: "У **черзі** (queue) завжди видаляється елемент, який міститься в множині довше за інших: в черзі реалізується стратегія **‹першим зайшов - першим вийшов›** (first-in, first-out — FIFO)".

До того ж при виводі черги на екран, ми будемо бачити лише саму чергу з її поточними елементими, тобто без таких, що можуть ще знаходитися в масиві або у зв'язному списку, проте вони уже вважатимуться виведені поза чергу.

Як відомо із теорії, черга вважається заповненою навіть якщо ще один елемент масиву лишається не заповненим.
*/

а) Масивом (використовувати кільцеву чергу)

```
"D:\KPI\АСД\ASD_Sakhnii Nazar FB-01\venv\Scripts\python.exe" "D:/KPI/АСД/ASD_Sakhnii Nazar FB-01/using_Queue.py"


◘ Задайте розмірність черги для її реалізації масивом та двозв'язним списком відповідно:
    ▼ Довжина черги повинна бути довільним натуральним числом!
        >>> Розмірність черги заданої масивом —→ some_number 24!
        >>> Довжина черги заданої двозв'язним списком —→ 4
Йой, введений тип даних не відповідає заданому формату!
Спробуйте використовувати лише натуральні числа


◘ Задайте розмірність черги для її реалізації масивом та двозв'язним списком відповідно:
    ▼ Довжина черги повинна бути довільним натуральним числом!
        >>> Розмірність черги заданої масивом —→ -4
        >>> Довжина черги заданої двозв'язним списком —→ 3.2
Йой, введений тип даних не відповідає заданому формату!
Спробуйте використовувати лише натуральні числа


◘ Задайте розмірність черги для її реалізації масивом та двозв'язним списком відповідно:
```

```
    ▼ Довжина черги повинна бути довільним натуральним числом!
       >>> Розмірність черги заданої масивом —→ 4
       >>> Довжина черги заданої двозв'язним списком —→ 4
Черги було успішно створено!


* Щоб продовжити, натисність |Enter| *

                                                  ◀◎▶ Меню можливих операцій ◀◎▶
       ------------------------------------------------------------------------------------------
       | 1. Помістити в чергу довільний елемент (Циклічна черга, реалізована масивом)
       | 2. Вивести із черги елемент, який зайшов першим до черги (Циклічна черга, реалізована масивом)
       |
       | 3. Помістити в чергу довільний елемент (Черга реалізована двозв'язним списком)
       | 4. Вивести із черги елемент, який зайшов першим до черги (Черга реалізована двозв'язним списком)
       |
       └ 0. Закінчити виконання програми

* Щоб виконати необхідну операцію меню, введіть її номер:
    >>> 1
◘ Новий елемент може містити будь-які символи:
       ▼ Який запис необхідно добавити до черги?
    >>> John
☼ Переглад черги, реалізованої за допомогою масиву, у яку було додано 'John' ↓
['John']

* Щоб продовжити, натисність |Enter| *

                                                  ◀◎▶ Меню можливих операцій ◀◎▶
       ------------------------------------------------------------------------------------------
       | 1. Помістити в чергу довільний елемент (Циклічна черга, реалізована масивом)
       | 2. Вивести із черги елемент, який зайшов першим до черги (Циклічна черга, реалізована масивом)
       |
       | 3. Помістити в чергу довільний елемент (Черга реалізована двозв'язним списком)
       | 4. Вивести із черги елемент, який зайшов першим до черги (Черга реалізована двозв'язним списком)
       |
       └ 0. Закінчити виконання програми

* Щоб виконати необхідну операцію меню, введіть її номер:
    >>> 1
◘ Новий елемент може містити будь-які символи:
       ▼ Який запис необхідно добавити до черги?
    >>> Steve
☼ Переглад черги, реалізованої за допомогою масиву, у яку було додано 'Steve' ↓
['John', 'Steve']
```

```
* Щоб продовжити, натисність |Enter| *

                                                ◀◎▶ Меню можливих операцій ◀◎▶
        ------------------------------------------------------------------------------------------------------
        | 1. Помістити в чергу довільний елемент (Циклічна черга, реалізована масивом)
        | 2. Вивести із черги елемент, який зайшов першим до черги (Циклічна черга, реалізована масивом)
        |
        | 3. Помістити в чергу довільний елемент (Черга реалізована двозв'язним списком)
        | 4. Вивести із черги елемент, який зайшов першим до черги (Черга реалізована двозв'язним списком)
        |
        └ 0. Закінчити виконання програми

* Щоб виконати необхідну операцію меню, введіть її номер:
    >>> 3000
Неа, такого номера опцерації в меню не існує

* Щоб продовжити, натисність |Enter| *

                                                ◀◎▶ Меню можливих операцій ◀◎▶
        ------------------------------------------------------------------------------------------------------
        | 1. Помістити в чергу довільний елемент (Циклічна черга, реалізована масивом)
        | 2. Вивести із черги елемент, який зайшов першим до черги (Циклічна черга, реалізована масивом)
        |
        | 3. Помістити в чергу довільний елемент (Черга реалізована двозв'язним списком)
        | 4. Вивести із черги елемент, який зайшов першим до черги (Черга реалізована двозв'язним списком)
        |
        └ 0. Закінчити виконання програми

* Щоб виконати необхідну операцію меню, введіть її номер:
    >>> 1
◘ Новий елемент може містити будь-які символи:
       ▼ Який запис необхідно добавити до черги?
    >>> Tony
☼ Переглад черги, реалізованої за допомогою масиву, у яку було додано 'Tony' ↓
['John', 'Steve', 'Tony']

* Щоб продовжити, натисність |Enter| *

                                                ◀◎▶ Меню можливих операцій ◀◎▶
        ------------------------------------------------------------------------------------------------------
        | 1. Помістити в чергу довільний елемент (Циклічна черга, реалізована масивом)
        | 2. Вивести із черги елемент, який зайшов першим до черги (Циклічна черга, реалізована масивом)
        |
        | 3. Помістити в чергу довільний елемент (Черга реалізована двозв'язним списком)
        | 4. Вивести із черги елемент, який зайшов першим до черги (Черга реалізована двозв'язним списком)
        |
```

```
        └ 0. Закінчити виконання програми

* Щоб виконати необхідну операцію меню, введіть її номер:
    >>> 1
◘ Новий елемент може містити будь-які символи:
        ▼ Який запис необхідно добавити до черги?
    >>> Ben
Черга заповнена, і спроба додати до неї елемент призводить до її переповнення
☼ Переглад заповної черги ↓
['John', 'Steve', 'Tony']

* Щоб продовжити, натисність |Enter| *

                                                ◄☼► Меню можливих операцій ◄☼►
        ------------------------------------------------------------------------------------------
        | 1. Помістити в чергу довільний елемент (Циклічна черга, реалізована масивом)
        | 2. Вивести із черги елемент, який зайшов першим до черги (Циклічна черга, реалізована масивом)
        |
        | 3. Помістити в чергу довільний елемент (Черга реалізована двозв'язним списком)
        | 4. Вивести із черги елемент, який зайшов першим до черги (Черга реалізована двозв'язним списком)
        |
        └ 0. Закінчити виконання програми


* Щоб виконати необхідну операцію меню, введіть її номер:
    >>> 2
☼ Переглад черги, реалізованої за допомогою масиву, із якої було виведено елемент 'John' ↓
['John', 'Steve']

* Щоб продовжити, натисність |Enter| *

                                                ◄☼► Меню можливих операцій ◄☼►
        ------------------------------------------------------------------------------------------
        | 1. Помістити в чергу довільний елемент (Циклічна черга, реалізована масивом)
        | 2. Вивести із черги елемент, який зайшов першим до черги (Циклічна черга, реалізована масивом)
        |
        | 3. Помістити в чергу довільний елемент (Черга реалізована двозв'язним списком)
        | 4. Вивести із черги елемент, який зайшов першим до черги (Черга реалізована двозв'язним списком)
        |
        └ 0. Закінчити виконання програми

* Щоб виконати необхідну операцію меню, введіть її номер:
    >>> 2
☼ Переглад черги, реалізованої за допомогою масиву, із якої було виведено елемент 'Steve' ↓
['Tony']
```

```
* Щоб продовжити, натисність |Enter| *

                                                  ◀◎▶ Меню можливих операцій ◀◎▶
        ------------------------------------------------------------------------------------------------
        | 1. Помістити в чергу довільний елемент (Циклічна черга, реалізована масивом)
        | 2. Вивести із черги елемент, який зайшов першим до черги (Циклічна черга, реалізована масивом)
        |
        | 3. Помістити в чергу довільний елемент (Черга реалізована двозв'язним списком)
        | 4. Вивести із черги елемент, який зайшов першим до черги (Черга реалізована двозв'язним списком)
        |
        └ 0. Закінчити виконання програми

* Щоб виконати необхідну операцію меню, введіть її номер:
    >>> 2
☼ Переглад порожньої черги ↓
[]

* Щоб продовжити, натисність |Enter| *

                                                  ◀◎▶ Меню можливих операцій ◀◎▶
        ------------------------------------------------------------------------------------------------
        | 1. Помістити в чергу довільний елемент (Циклічна черга, реалізована масивом)
        | 2. Вивести із черги елемент, який зайшов першим до черги (Циклічна черга, реалізована масивом)
        |
        | 3. Помістити в чергу довільний елемент (Черга реалізована двозв'язним списком)
        | 4. Вивести із черги елемент, який зайшов першим до черги (Черга реалізована двозв'язним списком)
        |
        └ 0. Закінчити виконання програми

* Щоб виконати необхідну операцію меню, введіть її номер:
    >>> 2
Черга порожня, і при спробі видалити з неї елемент відбувається помилка спустошення
☼ Переглад порожньої черги ↓
[]

* Щоб продовжити, натисність |Enter| *

                                                  ◀◎▶ Меню можливих операцій ◀◎▶
        ------------------------------------------------------------------------------------------------
        | 1. Помістити в чергу довільний елемент (Циклічна черга, реалізована масивом)
        | 2. Вивести із черги елемент, який зайшов першим до черги (Циклічна черга, реалізована масивом)
        |
        | 3. Помістити в чергу довільний елемент (Черга реалізована двозв'язним списком)
        | 4. Вивести із черги елемент, який зайшов першим до черги (Черга реалізована двозв'язним списком)
        |
        └ 0. Закінчити виконання програми
```

```
* Щоб виконати необхідну операцію меню, введіть її номер:
    >>> 0

Process finished with exit code 0
```

Зробивши деякі зміни у виводі черги та масиву елементів, продемоструємо ключові моменти як реалізується циклічна черга:

```
"D:\KPI\АСД\ASD_Sakhnii Nazar FB-01\venv\Scripts\python.exe" "D:/KPI/АСД/ASD_Sakhnii Nazar FB-01/using_Queue.py"

◘ Задайте розмірність черги для її реалізації масивом та двозв'язним списком відповідно:
    ▼ Довжина черги повинна бути довільним натуральним числом!
        >>> Розмірність черги заданої масивом —→ 4
        >>> Довжина черги заданої двозв'язним списком —→ 4
Черги було успішно створено!


* Щоб продовжити, натисність |Enter| *

                                          ◀◎▶ Меню можливих операцій ◀◎▶
       ------------------------------------------------------------------------------------------
       | 1. Помістити в чергу довільний елемент (Циклічна черга, реалізована масивом)
       | 2. Вивести із черги елемент, який зайшов першим до черги (Циклічна черга, реалізована масивом)
       |
       | 3. Помістити в чергу довільний елемент (Черга реалізована двозв'язним списком)
       | 4. Вивести із черги елемент, який зайшов першим до черги (Черга реалізована двозв'язним списком)
       |
       └ 0. Закінчити виконання програми

* Щоб виконати необхідну операцію меню, введіть її номер:
    >>> 1
◘ Новий елемент може містити будь-які символи:
        ▼ Який запис необхідно добавити до черги?
    >>> Джеймс
☼ Переглад черги, реалізованої за допомогою масиву, у яку було додано 'Джеймс' ↓
['Джеймс']
▼ Масив, який містить поточні елементи черги, а також ще можуть бути присутні елементи, які вже були виведені з масиву, проте на їхнє місце ще не було нічого записано:
['Джеймс', None, None, None]

* Щоб продовжити, натисність |Enter| *
```

```
                                                ◀ю▶ Меню можливих операцій ◀ю▶
        ------------------------------------------------------------------------------------------
        | 1. Помістити в чергу довільний елемент (Циклічна черга, реалізована масивом)
        | 2. Вивести із черги елемент, який зайшов першим до черги (Циклічна черга, реалізована масивом)
        |
        | 3. Помістити в чергу довільний елемент (Черга реалізована двозв'язним списком)
        | 4. Вивести із черги елемент, який зайшов першим до черги (Черга реалізована двозв'язним списком)
        |
        └ 0. Закінчити виконання програми

* Щоб виконати необхідну операцію меню, введіть її номер:
    >>> 1
◘ Новий елемент може містити будь-які символи:
        ▼ Який запис необхідно добавити до черги?
    >>> Роберт
☼ Переглад черги, реалізованої за допомогою масиву, у яку було додано 'Роберт' ↓
['Джеймс', 'Роберт']
▼ Масив, який містить поточні елементи черги, а також ще можуть бути присутні елементи, які вже були виведені з масиву, проте на
їхнє місце ще не було нічого записано:
['Джеймс', 'Роберт', None, None]

* Щоб продовжити, натисність |Enter| *

                                                ◀ю▶ Меню можливих операцій ◀ю▶
        ------------------------------------------------------------------------------------------
        | 1. Помістити в чергу довільний елемент (Циклічна черга, реалізована масивом)
        | 2. Вивести із черги елемент, який зайшов першим до черги (Циклічна черга, реалізована масивом)
        |
        | 3. Помістити в чергу довільний елемент (Черга реалізована двозв'язним списком)
        | 4. Вивести із черги елемент, який зайшов першим до черги (Черга реалізована двозв'язним списком)
        |
        └ 0. Закінчити виконання програми

* Щоб виконати необхідну операцію меню, введіть її номер:
    >>> 1
◘ Новий елемент може містити будь-які символи:
        ▼ Який запис необхідно добавити до черги?
    >>> Томас
☼ Переглад черги, реалізованої за допомогою масиву, у яку було додано 'Томас' ↓
['Джеймс', 'Роберт', 'Томас']
▼ Масив, який містить поточні елементи черги, а також ще можуть бути присутні елементи, які вже були виведені з масиву, проте на
їхнє місце ще не було нічого записано:
['Джеймс', 'Роберт', 'Томас', None]

* Щоб продовжити, натисність |Enter| *
```

```
                                        ◀◎▶ Меню можливих операцій ◀◎▶
       ------------------------------------------------------------------------------------------
       | 1. Помістити в чергу довільний елемент (Циклічна черга, реалізована масивом)
       | 2. Вивести із черги елемент, який зайшов першим до черги (Циклічна черга, реалізована масивом)
       |
       | 3. Помістити в чергу довільний елемент (Черга реалізована двозв'язним списком)
       | 4. Вивести із черги елемент, який зайшов першим до черги (Черга реалізована двозв'язним списком)
       |
       └ 0. Закінчити виконання програми

* Щоб виконати необхідну операцію меню, введіть її номер:
    >>> 1
◘ Новий елемент може містити будь-які символи:
       ▼ Який запис необхідно добавити до черги?
    >>> Джек
Черга заповнена, і спроба додати до неї елемент призводить до її переповнення
☼ Перегляд заповної черги ↓
['Джеймс', 'Роберт', 'Томас']
▼ Масив, який містить поточні елементи черги, а також ще можуть бути присутні елементи, які вже були виведені з масиву, проте на
їхнє місце ще не було нічого записано:
['Джеймс', 'Роберт', 'Томас', None]

* Щоб продовжити, натисність |Enter| *
```

Як можна помітити черга була заповнена, і тепер, щоб добавити новий елемент необхідно виконати операцію виведення

```
                                        ◀◎▶ Меню можливих операцій ◀◎▶
       ------------------------------------------------------------------------------------------
       | 1. Помістити в чергу довільний елемент (Циклічна черга, реалізована масивом)
       | 2. Вивести із черги елемент, який зайшов першим до черги (Циклічна черга, реалізована масивом)
       |
       | 3. Помістити в чергу довільний елемент (Черга реалізована двозв'язним списком)
       | 4. Вивести із черги елемент, який зайшов першим до черги (Черга реалізована двозв'язним списком)
       |
       └ 0. Закінчити виконання програми

* Щоб виконати необхідну операцію меню, введіть її номер:
    >>> 2
☼ Перегляд черги, реалізованої за допомогою масиву, із якої було виведено елемент 'Джеймс' ↓
['Роберт', 'Томас']
▼ Масив, який містить поточні елементи черги, а також ще можуть бути присутні елементи, які вже були виведені з масиву, проте на
їхнє місце ще не було нічого записано:
['Джеймс', 'Роберт', 'Томас', None]

* Щоб продовжити, натисність |Enter| *

                                        ◀◎▶ Меню можливих операцій ◀◎▶
       ------------------------------------------------------------------------------------------
```

```
        | 1. Помістити в чергу довільний елемент (Циклічна черга, реалізована масивом)
        | 2. Вивести із черги елемент, який зайшов першим до черги (Циклічна черга, реалізована масивом)
        |
        | 3. Помістити в чергу довільний елемент (Черга реалізована двозв'язним списком)
        | 4. Вивести із черги елемент, який зайшов першим до черги (Черга реалізована двозв'язним списком)
        |
        └ 0. Закінчити виконання програми

* Щоб виконати необхідну операцію меню, введіть її номер:
    >>> 2
☼ Переглад черги, реалізованої за допомогою масиву, із якої було виведено елемент 'Роберт' ↓
['Томас']
▼ Масив, який містить поточні елементи черги, а також ще можуть бути присутні елементи, які вже були виведені з масиву, проте на
їхнє місце ще не було нічого записано:
['Джеймс', 'Роберт', 'Томас', None]

* Щоб продовжити, натисність |Enter| *
```

Було виконано операцію виведення 2 рази, тобто 2 елементи вийшли з черги, однак у масиві вони до сих пір залишаються, поки на

```
                                        ◀○▶ Меню можливих операцій ◀○▶
        ------------------------------------------------------------------------------------------------------
        | 1. Помістити в чергу довільний елемент (Циклічна черга, реалізована масивом)
        | 2. Вивести із черги елемент, який зайшов першим до черги (Циклічна черга, реалізована масивом)
        |
        | 3. Помістити в чергу довільний елемент (Черга реалізована двозв'язним списком)
        | 4. Вивести із черги елемент, який зайшов першим до черги (Черга реалізована двозв'язним списком)
        |
        └ 0. Закінчити виконання програми

* Щоб виконати необхідну операцію меню, введіть її номер:
    >>> 1
◘ Новий елемент може містити будь-які символи:
        ▼ Який запис необхідно добавити до черги?
    >>> Джек
☼ Переглад черги, реалізованої за допомогою масиву, у яку було додано 'Джек' ↓
['Томас', 'Джек']
▼ Масив, який містить поточні елементи черги, а також ще можуть бути присутні елементи, які вже були виведені з масиву, проте на
їхнє місце ще не було нічого записано:
['Джеймс', 'Роберт', 'Томас', 'Джек']

* Щоб продовжити, натисність |Enter| *

                                        ◀○▶ Меню можливих операцій ◀○▶
        ------------------------------------------------------------------------------------------------------
        | 1. Помістити в чергу довільний елемент (Циклічна черга, реалізована масивом)
        | 2. Вивести із черги елемент, який зайшов першим до черги (Циклічна черга, реалізована масивом)
        |
```

```
        | 3. Помістити в чергу довільний елемент (Черга реалізована двозв'язним списком)
        | 4. Вивести із черги елемент, який зайшов першим до черги (Черга реалізована двозв'язним списком)
        |
        L 0. Закінчити виконання програми

* Щоб виконати необхідну операцію меню, введіть її номер:
    >>> 1
◘ Новий елемент може містити будь-які символи:
        ▼ Який запис необхідно добавити до черги?
    >>> Авраам
☼ Перегляд черги, реалізованої за допомогою масиву, у яку було додано 'Авраам' ↓
['Томас', 'Джек', 'Авраам']
▼ Масив, який містить поточні елементи черги, а також ще можуть бути присутні елементи, які вже були виведені з масиву, проте на
їхнє місце ще не було нічого записано:
['Авраам', 'Роберт', 'Томас', 'Джек']

* Щоб продовжити, натисність |Enter| *
```

Після того як ми добавили 2 елемента до черги, то можна помітити, що на порожнє місце (None), яке було до цього, було додано елемент ‘Джек’, а на місце ‘Джеймс’ записали ‘Авраам’

```
                ◄☼► Меню можливих операцій ◄☼►
        ------------------------------------------------------------------------------------------------
        | 1. Помістити в чергу довільний елемент (Циклічна черга, реалізована масивом)
        | 2. Вивести із черги елемент, який зайшов першим до черги (Циклічна черга, реалізована масивом)
        |
        | 3. Помістити в чергу довільний елемент (Черга реалізована двозв'язним списком)
        | 4. Вивести із черги елемент, який зайшов першим до черги (Черга реалізована двозв'язним списком)
        |
        L 0. Закінчити виконання програми

* Щоб виконати необхідну операцію меню, введіть її номер:
    >>> 2
☼ Перегляд черги, реалізованої за допомогою масиву, із якої було виведено елемент 'Томас' ↓
['Джек', 'Авраам']
▼ Масив, який містить поточні елементи черги, а також ще можуть бути присутні елементи, які вже були виведені з масиву, проте на
їхнє місце ще не було нічого записано:
['Авраам', 'Роберт', 'Томас', 'Джек']

* Щоб продовжити, натисність |Enter| *
```

Знову виконуємо операцію виведення елемента з черги, і помічаємо, що ‘Томас’ із черги, проте він ще залишається в масиві елементів, як і ‘Роберт’, який давніше уже вийшов з черги.

```
                ◄☼► Меню можливих операцій ◄☼►
        ------------------------------------------------------------------------------------------------
        | 1. Помістити в чергу довільний елемент (Циклічна черга, реалізована масивом)
        | 2. Вивести із черги елемент, який зайшов першим до черги (Циклічна черга, реалізована масивом)
        |
        | 3. Помістити в чергу довільний елемент (Черга реалізована двозв'язним списком)
        | 4. Вивести із черги елемент, який зайшов першим до черги (Черга реалізована двозв'язним списком)
        |
```

```
       └ 0. Закінчити виконання програми

* Щоб виконати необхідну операцію меню, введіть її номер:
    >>> 1
◘ Новий елемент може містити будь-які символи:
       ▼ Який запис необхідно добавити до черги?
    >>> Стів
☼ Перегляд черги, реалізованої за допомогою масиву, у яку було додано 'Стів' ↓
['Джек', 'Авраам', 'Стів']
▼ Масив, який містить поточні елементи черги, а також ще можуть бути присутні елементи, які вже були виведені з масиву, проте на їхнє місце ще не було нічого записано:
['Авраам', 'Стів', 'Томас', 'Джек']

* Щоб продовжити, натисність |Enter| *
```

Так як у нас є вільне місце, то на це раз додаємо елемент ‘Стів’ у кінець черги, і внаслідок чого вона стає знову заповненою. Щодо масиву елементів, то можна помітити, що він заповнений повністю, проте ‘Томас’ уже не знаходиться в черзі, тому його можна перезаписати наступним. І як бачимо знову, черга дійсно є циклічною, так як ‘Стів’ додається в середину черги, яка колись до цього містила зовсім інший елемент

b) Зв'язним списком

```
"D:\KPI\АСД\ASD_Sakhnii Nazar FB-01\venv\Scripts\python.exe" "D:/KPI/АСД/ASD_Sakhnii Nazar FB-01/using_Queue.py"

◘ Задайте розмірність черги для її реалізації масивом та двозв'язним списком відповідно:
    ▼ Довжина черги повинна бути довільним натуральним числом!
       >>> Розмірність черги заданої масивом —→ Яке%сь чи!!сл0
       >>> Довжина черги заданої двозв'язним списком —→ 9
Йой, введений тип даних не відповідає заданому формату!
Спробуйте використовувати лише натуральні числа


◘ Задайте розмірність черги для її реалізації масивом та двозв'язним списком відповідно:
    ▼ Довжина черги повинна бути довільним натуральним числом!
       >>> Розмірність черги заданої масивом —→ -500
       >>> Довжина черги заданої двозв'язним списком —→ 3.6
Йой, введений тип даних не відповідає заданому формату!
Спробуйте використовувати лише натуральні числа


◘ Задайте розмірність черги для її реалізації масивом та двозв'язним списком відповідно:
    ▼ Довжина черги повинна бути довільним натуральним числом!
       >>> Розмірність черги заданої масивом —→ 4
       >>> Довжина черги заданої двозв'язним списком —→ 4
Черги було успішно створено!
```

```
* Щоб продовжити, натисність |Enter| *

                                        ◀o▶ Меню можливих операцій ◀o▶
        ------------------------------------------------------------------------------------------
        | 1. Помістити в чергу довільний елемент (Циклічна черга, реалізована масивом)
        | 2. Вивести із черги елемент, який зайшов першим до черги (Циклічна черга, реалізована масивом)
        |
        | 3. Помістити в чергу довільний елемент (Черга реалізована двозв'язним списком)
        | 4. Вивести із черги елемент, який зайшов першим до черги (Черга реалізована двозв'язним списком)
        |
        └ 0. Закінчити виконання програми

* Щоб виконати необхідну операцію меню, введіть її номер:
    >>> 3
◘ Новий елемент може містити будь-які символи:
       ▼ Який запис необхідно добавити до черги?
    >>> Андрій
☼ Переглад черги, реалізованої за допомогою двозв'язного списку, у яку було додано 'Андрій' ↓
Андрій <->
* Щоб продовжити, натисність |Enter| *

                                        ◀o▶ Меню можливих операцій ◀o▶
        ------------------------------------------------------------------------------------------
        | 1. Помістити в чергу довільний елемент (Циклічна черга, реалізована масивом)
        | 2. Вивести із черги елемент, який зайшов першим до черги (Циклічна черга, реалізована масивом)
        |
        | 3. Помістити в чергу довільний елемент (Черга реалізована двозв'язним списком)
        | 4. Вивести із черги елемент, який зайшов першим до черги (Черга реалізована двозв'язним списком)
        |
        └ 0. Закінчити виконання програми

* Щоб виконати необхідну операцію меню, введіть її номер:
    >>> Юра 10
Неа, такого номера опцерації в меню не існує

* Щоб продовжити, натисність |Enter| *

                                        ◀o▶ Меню можливих операцій ◀o▶
        ------------------------------------------------------------------------------------------
        | 1. Помістити в чергу довільний елемент (Циклічна черга, реалізована масивом)
        | 2. Вивести із черги елемент, який зайшов першим до черги (Циклічна черга, реалізована масивом)
        |
        | 3. Помістити в чергу довільний елемент (Черга реалізована двозв'язним списком)
        | 4. Вивести із черги елемент, який зайшов першим до черги (Черга реалізована двозв'язним списком)
```

```
        |
        └ 0. Закінчити виконання програми

* Щоб виконати необхідну операцію меню, введіть її номер:
    >>> 3
◘ Новий елемент може містити будь-які символи:
        ▼ Який запис необхідно добавити до черги?
    >>> Олег
☼ Переглад черги, реалізованої за допомогою двозв'язного списку, у яку було додано 'Олег' ↓
Андрій <-> Олег <->

* Щоб продовжити, натисність |Enter| *

                                                ◄o► Меню можливих операцій ◄o►
        ------------------------------------------------------------------------------------------
        | 1. Помістити в чергу довільний елемент (Циклічна черга, реалізована масивом)
        | 2. Вивести із черги елемент, який зайшов першим до черги (Циклічна черга, реалізована масивом)
        |
        | 3. Помістити в чергу довільний елемент (Черга реалізована двозв'язним списком)
        | 4. Вивести із черги елемент, який зайшов першим до черги (Черга реалізована двозв'язним списком)
        |
        └ 0. Закінчити виконання програми

* Щоб виконати необхідну операцію меню, введіть її номер:
    >>> 3
◘ Новий елемент може містити будь-які символи:
        ▼ Який запис необхідно добавити до черги?
    >>> Назар
☼ Переглад черги, реалізованої за допомогою двозв'язного списку, у яку було додано 'Назар' ↓
Андрій <-> Олег <-> Назар <->

* Щоб продовжити, натисність |Enter| *

                                                ◄o► Меню можливих операцій ◄o►
        ------------------------------------------------------------------------------------------
        | 1. Помістити в чергу довільний елемент (Циклічна черга, реалізована масивом)
        | 2. Вивести із черги елемент, який зайшов першим до черги (Циклічна черга, реалізована масивом)
        |
        | 3. Помістити в чергу довільний елемент (Черга реалізована двозв'язним списком)
        | 4. Вивести із черги елемент, який зайшов першим до черги (Черга реалізована двозв'язним списком)
        |
        └ 0. Закінчити виконання програми

* Щоб виконати необхідну операцію меню, введіть її номер:
    >>> 3
◘ Новий елемент може містити будь-які символи:
```

```
        ▼ Який запис необхідно добавити до черги?
    >>> Дмитро
Черга заповнена, і спроба додати до неї елемент призводить до її переповнення
☼ Перегляд заповної черги ↓
Андрій <-> Олег <-> Назар <->

* Щоб продовжити, натисність |Enter| *

                                          ◀o▶ Меню можливих операцій ◀o▶
       ------------------------------------------------------------------------------------------
       | 1. Помістити в чергу довільний елемент (Циклічна черга, реалізована масивом)
       | 2. Вивести із черги елемент, який зайшов першим до черги (Циклічна черга, реалізована масивом)
       |
       | 3. Помістити в чергу довільний елемент (Черга реалізована двозв'язним списком)
       | 4. Вивести із черги елемент, який зайшов першим до черги (Черга реалізована двозв'язним списком)
       |
       └ 0. Закінчити виконання програми

* Щоб виконати необхідну операцію меню, введіть її номер:
    >>> 4
☼ Перегляд черги, реалізованої за допомогою двозв'язного списку, із якої було виведено елемент 'Андрій' ↓
Олег <-> Назар <->

* Щоб продовжити, натисність |Enter| *

                                          ◀o▶ Меню можливих операцій ◀o▶
       ------------------------------------------------------------------------------------------
       | 1. Помістити в чергу довільний елемент (Циклічна черга, реалізована масивом)
       | 2. Вивести із черги елемент, який зайшов першим до черги (Циклічна черга, реалізована масивом)
       |
       | 3. Помістити в чергу довільний елемент (Черга реалізована двозв'язним списком)
       | 4. Вивести із черги елемент, який зайшов першим до черги (Черга реалізована двозв'язним списком)
       |
       └ 0. Закінчити виконання програми

* Щоб виконати необхідну операцію меню, введіть її номер:
    >>> 4
☼ Перегляд черги, реалізованої за допомогою двозв'язного списку, із якої було виведено елемент 'Олег' ↓
Назар <->

* Щоб продовжити, натисність |Enter| *

                                          ◀o▶ Меню можливих операцій ◀o▶
       ------------------------------------------------------------------------------------------
       | 1. Помістити в чергу довільний елемент (Циклічна черга, реалізована масивом)
       | 2. Вивести із черги елемент, який зайшов першим до черги (Циклічна черга, реалізована масивом)
```

```
        |
        | 3. Помістити в чергу довільний елемент (Черга реалізована двозв'язним списком)
        | 4. Вивести із черги елемент, який зайшов першим до черги (Черга реалізована двозв'язним списком)
        |
        └ 0. Закінчити виконання програми

* Щоб виконати необхідну операцію меню, введіть її номер:
    >>> 4
Єдиний елемент двозв'язного списку, що залишився, видалено!
☼ Переглад черги, реалізованої за допомогою двозв'язного списку, із якої було виведено елемент 'Назар' ↓
Не можливо виконати, так як двозв'язний список не містить жодного елемента

* Щоб продовжити, натисність |Enter| *

                                          ◀o▶ Меню можливих операцій ◀o▶
        ------------------------------------------------------------------------------------------
        | 1. Помістити в чергу довільний елемент (Циклічна черга, реалізована масивом)
        | 2. Вивести із черги елемент, який зайшов першим до черги (Циклічна черга, реалізована масивом)
        |
        | 3. Помістити в чергу довільний елемент (Черга реалізована двозв'язним списком)
        | 4. Вивести із черги елемент, який зайшов першим до черги (Черга реалізована двозв'язним списком)
        |
        └ 0. Закінчити виконання програми

* Щоб виконати необхідну операцію меню, введіть її номер:
    >>> 0

Process finished with exit code 0
```

Програмні коди:

- Файл `Queue_using_Array.py` містить реалізацію циклічної черги за допомогою масива:

```python
class Array_CyclicQueue:
    # Кільцева черга, реалізована через масив

    def __init__(self, size):
        """ Конструктор черги, який створює порожній масив заданої довжини """
        self.head = 0
        self.tail = 0
```

```
        self.queue = [None for _ in range(size)]

    def Queue_is_Empty(self) -> bool:
        """ Перевірка на те, чи черга порожня """
        if self.head == self.tail:
            return True
        else:
            return False

    def Queue_is_Full(self) -> bool:
        """ Перевірка на те, чи черга заповнена """
        if self.head == self.tail + 1 or (self.head == 0 and self.tail == len(self.queue) - 1):
            return True
        else:
            return False

    def Enqueue(self, new_item):
        if self.Queue_is_Full():
            print("Черга заповнена, і спроба додати до неї елемент призводить до її переповнення")
            return True
        else:  # Інакше додаєм елемент на відповідну позицію
            self.queue[self.tail] = new_item
            if self.tail == len(self.queue) - 1:
                self.tail = 0
            else:
                self.tail = self.tail + 1
            return None

    def Dequeue(self):
        if self.Queue_is_Empty():
            print("Черга порожня, і при спробі видалити з неї елемент відбувається помилка спустошення")
            return False
        else:  # Інакше видаляємо з неї елемент за принципом "First-In —→ First-Out"
            del_item = self.queue[self.head]
            if self.head == len(self.queue) - 1:
                self.head = 0
            else:
                self.head = self.head + 1
            return del_item

    def outputArray(self):
        print([item for item in self.queue[self.head:self.tail]])
```

- Файл `Queue_using_DL_list.py` містить реалізацію черги за допомогою двозв'язного списку:

(Тут було використано файл `DoubleLinkedList.py`, як уже реалізований двозв'язний список для Лабораторної роботи №3)

```
from DoubleLinkedList import *

class DoubleLinked_Queue:
    # Кільцева черга, реалізована через масив

    def __init__(self, size):
        """ Конструктор черги, який створює порожній масив заданої довжини """
        self.head = 0
        self.tail = 0
        self.length = size
        self.queue = DoublyLinkedList()

    def Queue_is_Empty(self) -> bool:
        """ Перевірка на те, чи черга порожня """
        if self.head == self.tail:
            return True
        else:
            return False

    def Queue_is_Full(self) -> bool:
        """ Перевірка на те, чи черга заповнена """
        if self.head == self.tail + 1 or (self.head == 0 and self.tail == self.length - 1):
            return True
        else:
            return False

    def Enqueue(self, new_item):
        if self.Queue_is_Full():
            print("Черга заповнена, і спроба додати до неї елемент призводить до її переповнення")
            return True
        else:  # Інакше додаєм елемент на відповідну позицію
            self.queue.insertAtEnd(new_item)
            if self.tail == self.length - 1:
                self.tail = 0
            else:
                self.tail = self.tail + 1
            return None

    def Dequeue(self):
        if self.Queue_is_Empty():
```

```
            print("Черга порожня, і при спробі видалити з неї елемент відбувається помилка спустошення")
            return False
        else:  # Інакше видаляємо з неї елемент за принципом "First-In —→ First-Out"
            del_item = self.queue.deleteAtBegin()
            if self.head == self.length - 1:
                self.head = 0
            else:
                self.head = self.head + 1
            return del_item
```

- Основний файл `using_Queue.py` містить реалізацію функціональних можливостей (`Enqueue & Dequeue`) черги, реалізованої масивом та зв'язним списком (У тому числі були враховані можливі помилки):

```
from Queue_using_Array import *
from Queue_using_DL_list import *


while True:
    print("\n▫ Задайте розмірність черги для її реалізації масивом та двозв'язним списком відповідно: \n\
    ▼ Довжина черги повинна бути довільним натуральним числом!")
    array_size = input("\t>>> Розмірність черги заданої масивом —→ ")
    dl_list_size = input("\t>>> Довжина черги заданої двозв'язним списком —→ ")
    try:
        array_queue = Array_CyclicQueue(int(array_size))
        dl_list_queue = DoubleLinked_Queue(int(dl_list_size))
        print("Черги було успішно створено! \n")
        break
    except TypeError and ValueError:
        print("Йой, введений тип даних не відповідає заданому формату! \n"
              "Спробуйте використовувати лише натуральні числа \n")


while True:
    input("\n* Щоб продовжити, натисність |Enter| *\n")
    print("\t\t\t\t\t\t    ◀◎▶ Меню можливих операцій ◀◎▶    \n"
          "\t-------------------------------------------------------------------------------------------\n"
          "\t| 1. Помістити в чергу довільний елемент (Циклічна черга, реалізована масивом)\n"
          "\t| 2. Вивести із черги елемент, який зайшов першим до черги (Циклічна черга, реалізована масивом)\n\t|\n"
          "\t| 3. Помістити в чергу довільний елемент (Черга реалізована двозв'язним списком)\n"
          "\t| 4. Вивести із черги елемент, який зайшов першим до черги (Черга реалізована двозв'язним списком)\n\t|\n"
          "\t└ 0. Закінчити виконання програми\n")
```

```
    option = input("* Щоб виконати необхідну операцію меню, введіть її номер: \n    >>> ")
    if option == "1":
        new_item = input("◘ Новий елемент може містити будь-які символи: \n\
                         \r\t▼ Який запис необхідно добавити до черги? \n    >>> ")
        if array_queue.Enqueue(new_item):
            print(f"☼ Переглад заповної черги ↓ ")
            array_queue.outputArray()
        else:
            print(f"☼ Переглад черги, реалізованої за допомогою масиву, у яку було додано '{new_item}' ↓ ")
            array_queue.outputArray()
    elif option == "2":
        if not (del_item := array_queue.Dequeue()):
            print(f"☼ Переглад порожньої черги ↓ ")
            array_queue.outputArray()
        else:
            print(f"☼ Переглад черги, реалізованої за допомогою масиву, із якої було виведено елемент '{del_item}' ↓ ")
            array_queue.outputArray()
    elif option == "3":
        new_item = input("◘ Новий елемент може містити будь-які символи: \n\
                    \r\t▼ Який запис необхідно добавити до черги? \n    >>> ")
        if dl_list_queue.Enqueue(new_item):
            print(f"☼ Переглад заповної черги ↓ ")
            dl_list_queue.queue.outputList()
        else:
            print(f"☼ Переглад черги, реалізованої за допомогою двозв'язного списку, у яку було додано '{new_item}' ↓ ")
            dl_list_queue.queue.outputList()
    elif option == "4":
        if not (del_item := dl_list_queue.Dequeue()):
            print(f"☼ Переглад порожньої черги ↓ ")
            dl_list_queue.queue.outputList()
        else:
            print(f"☼ Переглад черги, реалізованої за допомогою двозв'язного списку, "
                  f"із якої було виведено елемент '{del_item.mItem}' ↓ ")
            dl_list_queue.queue.outputList()
    elif option == "0":
        break
    else:
        print("Неа, такого номера опцерації в меню не існує")
```