МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ  
«ХАРКІВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

Кафедра «Системний аналіз та інформаційно-аналітичні технології»

Лабораторна робота № 2  
з дисципліни «Алгоритми та структури даних»

(Варіант III)

Виконав:

Олексій ДРАБЧАК

студент групи КН-320Г

Перевірила:

Євгенія МОШКО

Харків 2022

**Тема лабораторної роботи:** Стеки, черги і декі та робота з ними

**Мета:** ознайомитися із основними способами організації стеків, черг, деків та особливостями їх програмної реалізації. Набути практичних навичок роботи зі стеками, чергами та деками.

# Порядок виконання роботи:

1. Написати програму, яка демонструє основні дії по роботі із стеком пропонує користувачу вибір дії:
   * поміняти місцями перший і останній елементи стека;
   * розгорнути стек, тобто зробити "дно" стека вершиною, а вершину - "дном";
   * видалити кожен другий елемент стека;
   * вставити символ '\*' в середину стека, якщо число елементів парне, або після середнього елемента, якщо число елементів непарне;
   * знайти максимальний елемент і вставити після нього 0;
   * видалити мінімальний елемент;
   * видалити всі елементи, крім першого;
   * видалити всі елементи, крім останнього.
2. Написати програму, яка демонструє основні дії по роботі із чергою пропонує користувачу вибір дії:

* знайти та вивести кількість елементів черги;
* знайти та вивести середнє арифметичне збережених елементів;
* знайти та вивести мінімальний та максимальний елемент;
* знайти та вивести елемент, що йде перед мінімальним елементом.

1. Написати програму, яка демонструє основні дії по роботі із деком пропонує користувачу вибір дії:

* створити дек;
* перевірити на порожнечу;
* додати елемент в початок;
* додати елемент в кінець;
* прочитати шостий елемент;
* прочитати останній елемент;
* перевірити на порожнечу.

Для уявлення дека використовуються ті ж набори змінних, що і для черги. Варіант завдання вибирається за номером студента в списку групи.

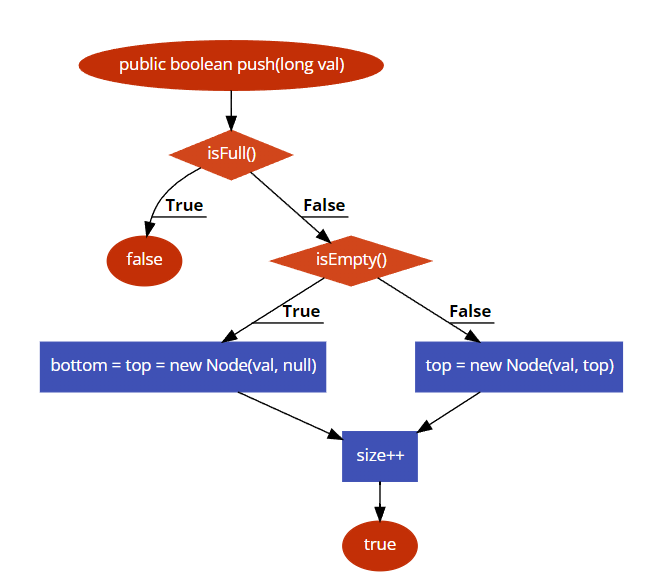


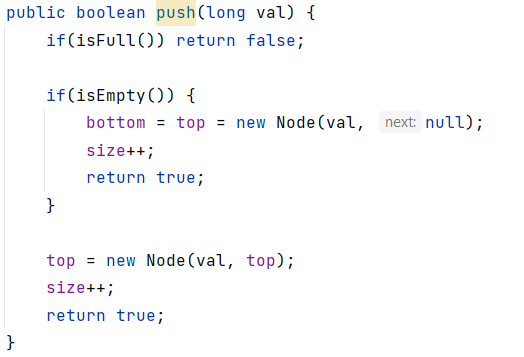
Варіанти завдань наведені в таблиці 1.

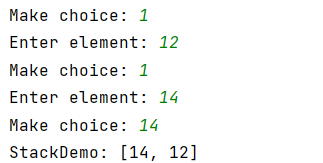
Зберегти файл з тестом програми для подальших робіт.

**Блок-схеми методів для роботи зі стеком:**

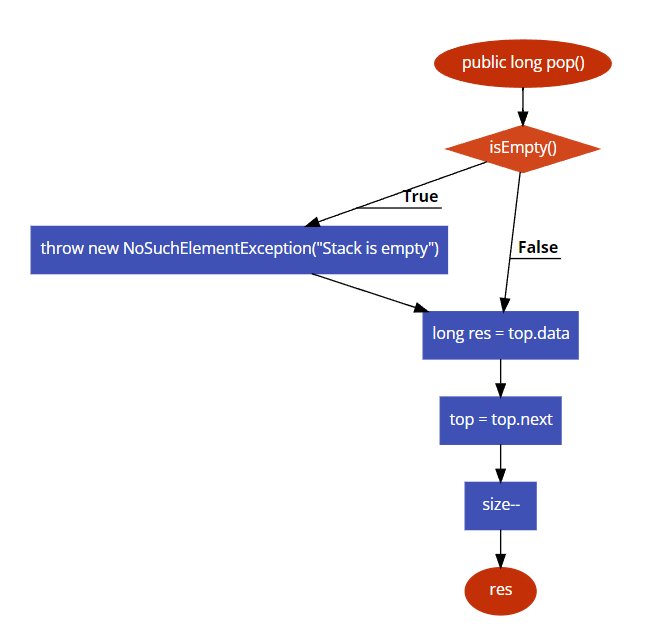
1. push() – додавання нового елемента в стек.

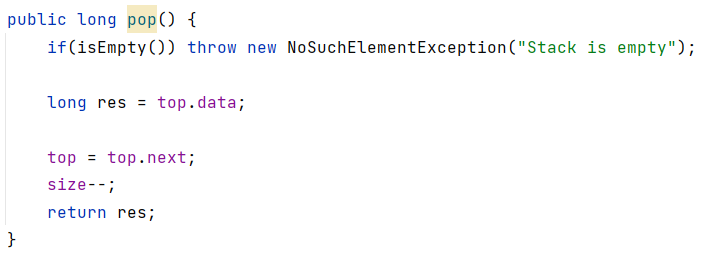


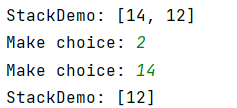




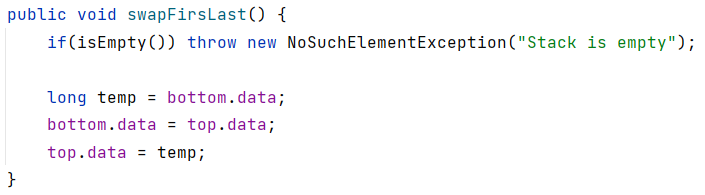
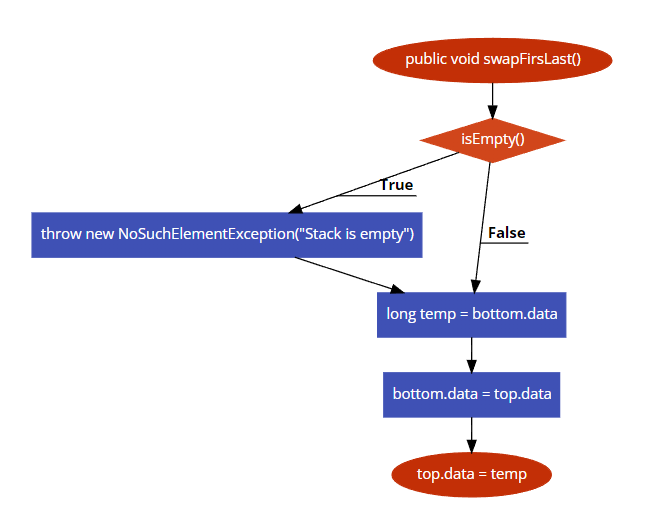
1. pop() – видалення елемента зі стека.

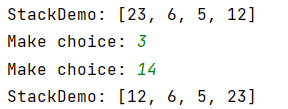




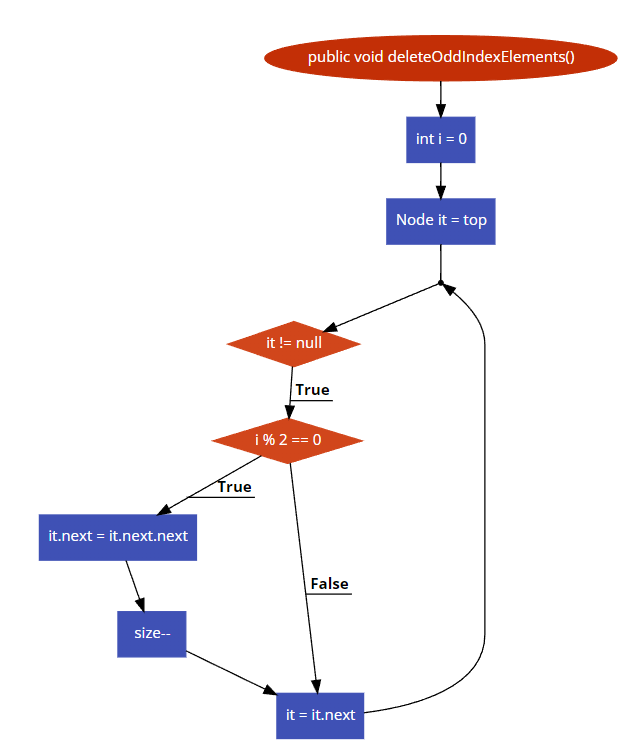


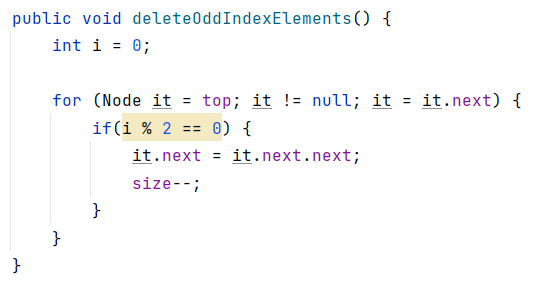
1. swapFirsLast() – заміна першого та останнього елемента стеку.

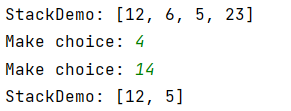




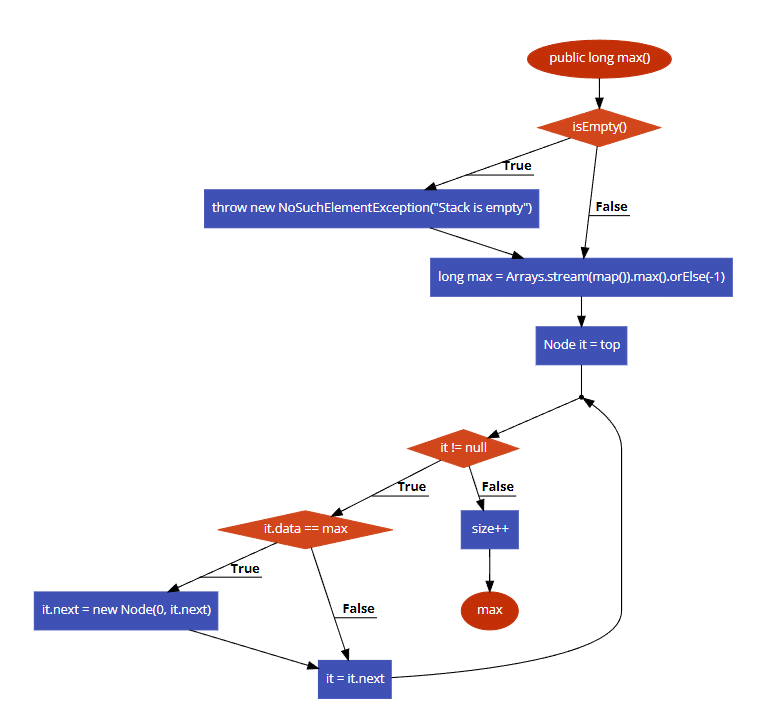
1. deleteOddIndexElements() – видалення кожного другого елемента стека.

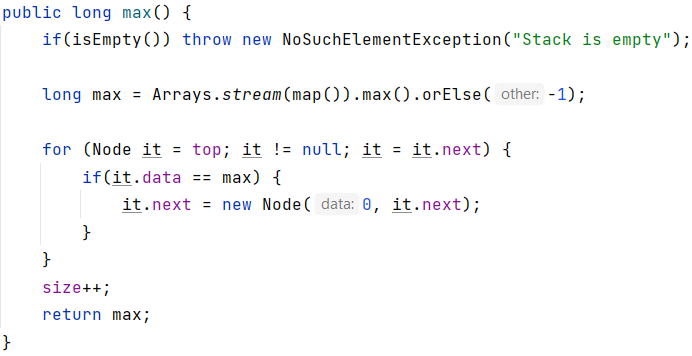


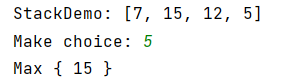




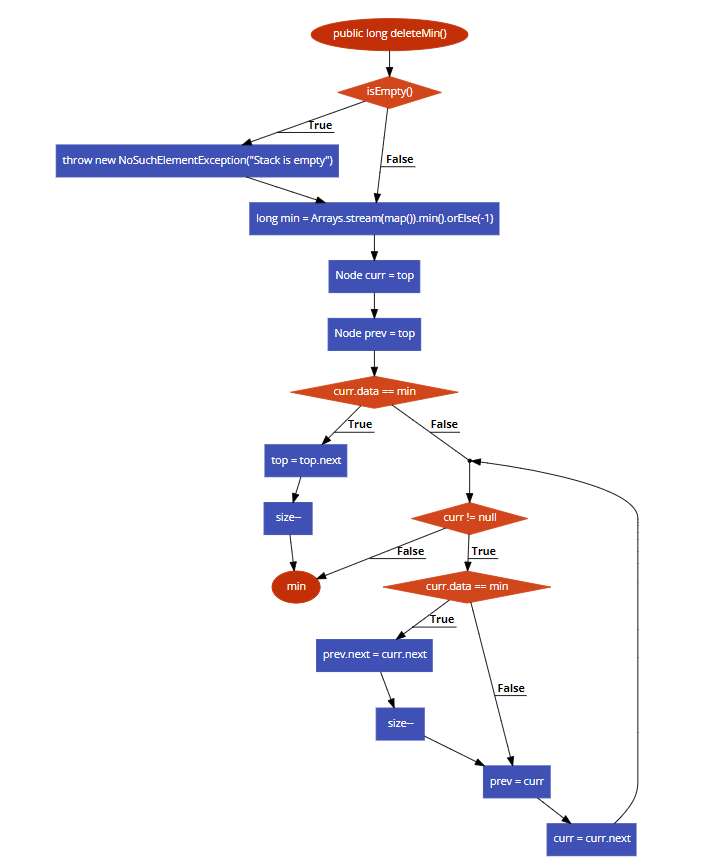
1. max() – знаходження максимального елемента стеку.

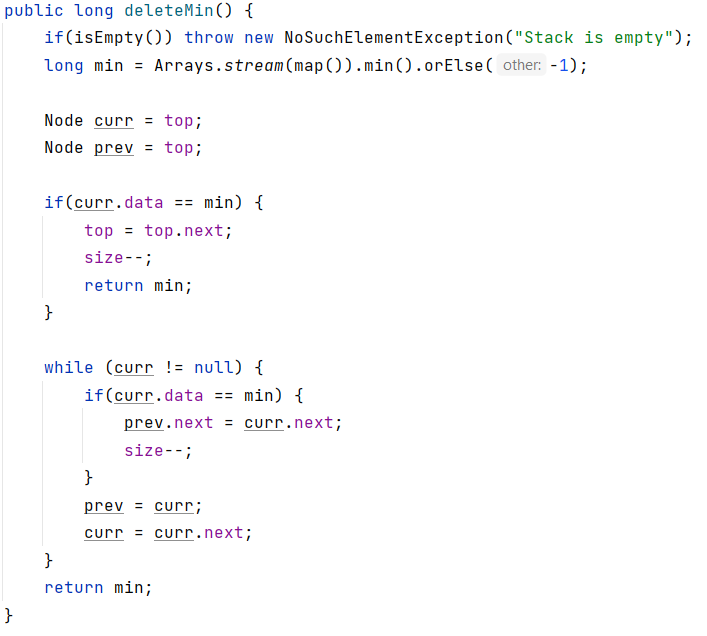


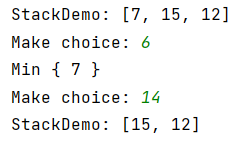




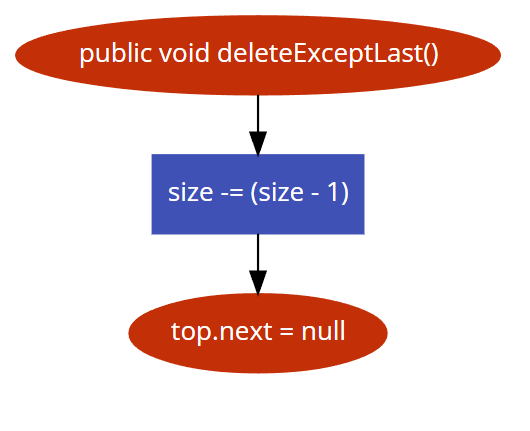
1. deleteMin() – знаходження і видалення мінімального елемента зі стека.

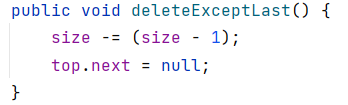


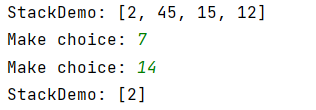




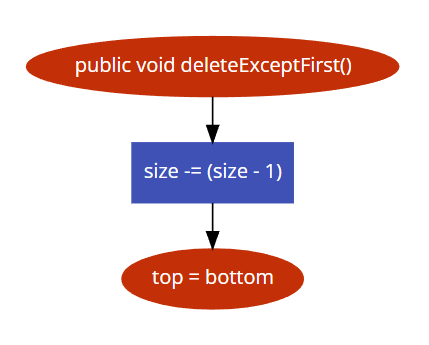
1. deleteExceptLast() – видалення усіх елементів стека окрім останнього.

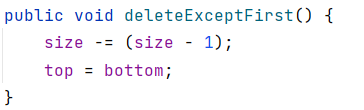


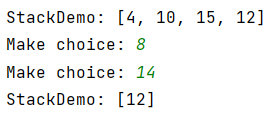




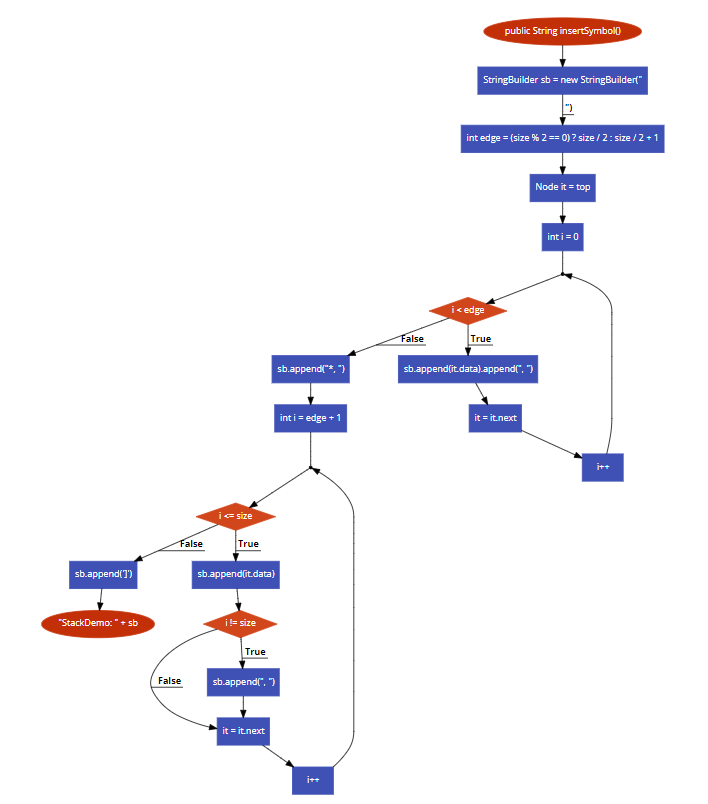
1. deleteExceptFirst() – видалення усіх елементів стека окрім першого.

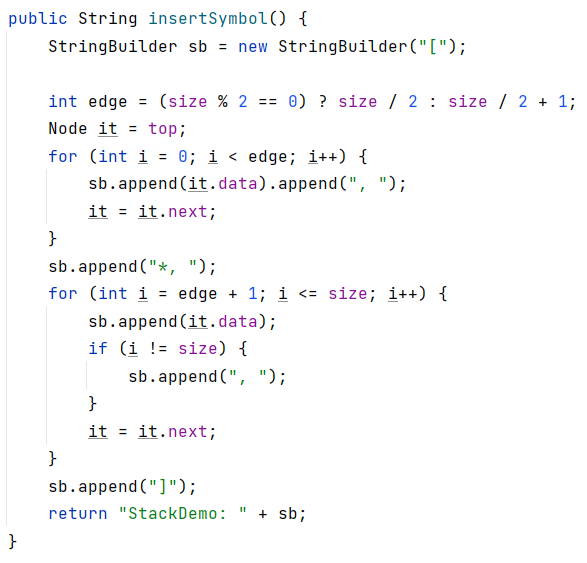


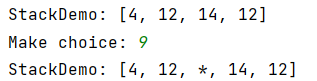




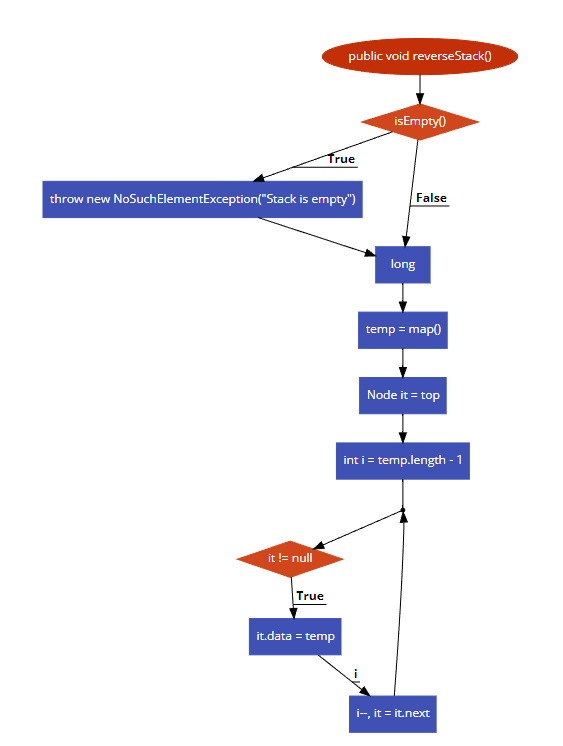
1. insertSymbol() – вставлення символу '\*'.

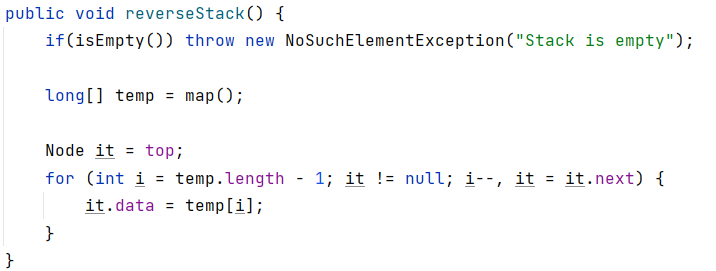


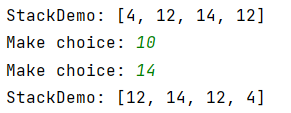




1. reverseStack() – розгорнути стек.

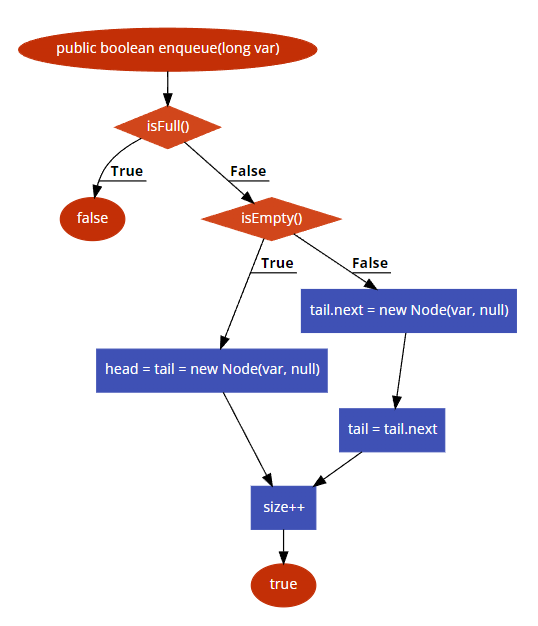


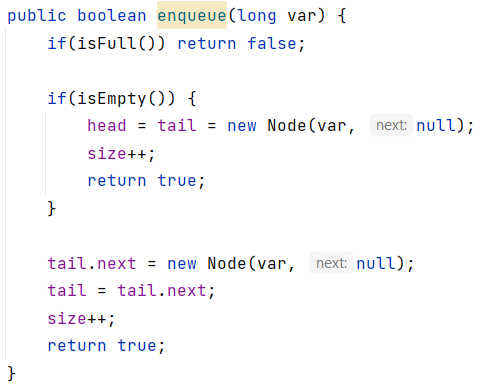


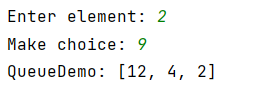


**Блок-схеми методів для роботи із чергами:**

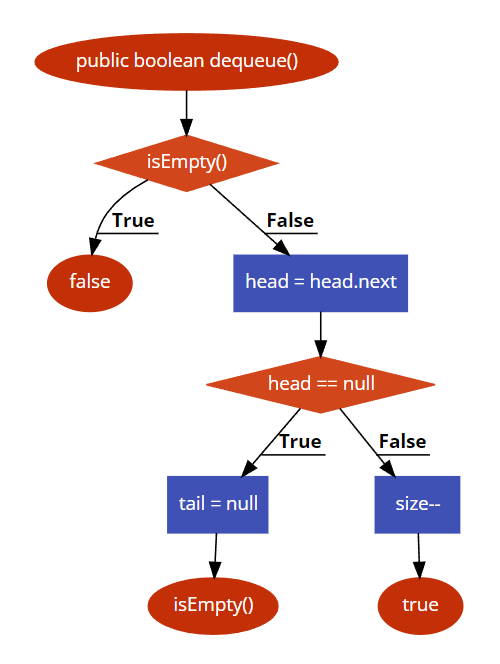
1. enqueue(long var) – вставлення нового елемента в чергу.

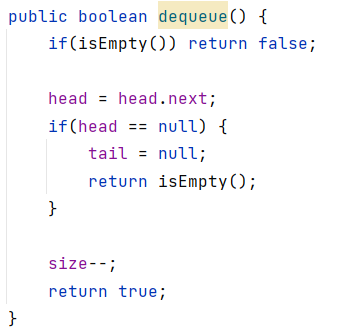


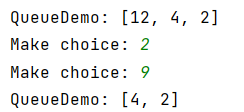




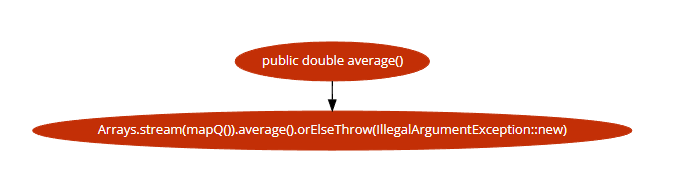
1. dequeue() – видалення елемента із черги.

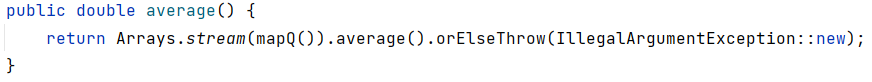


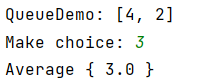




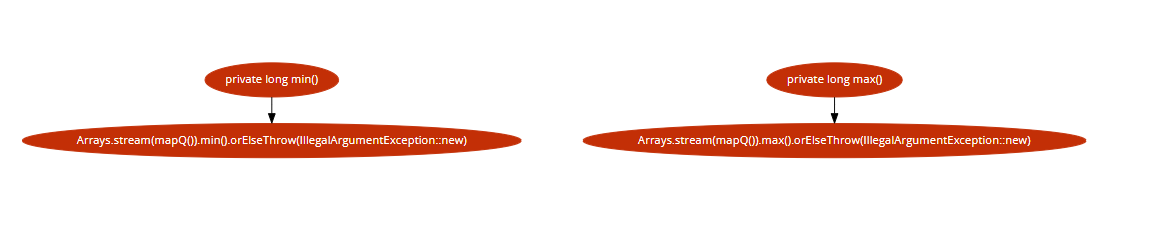
1. average() – обчислення середнього значення.

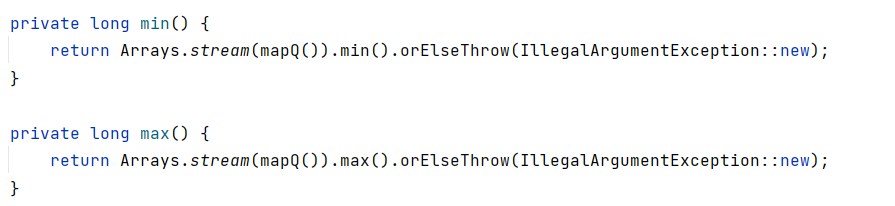


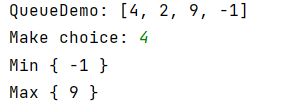




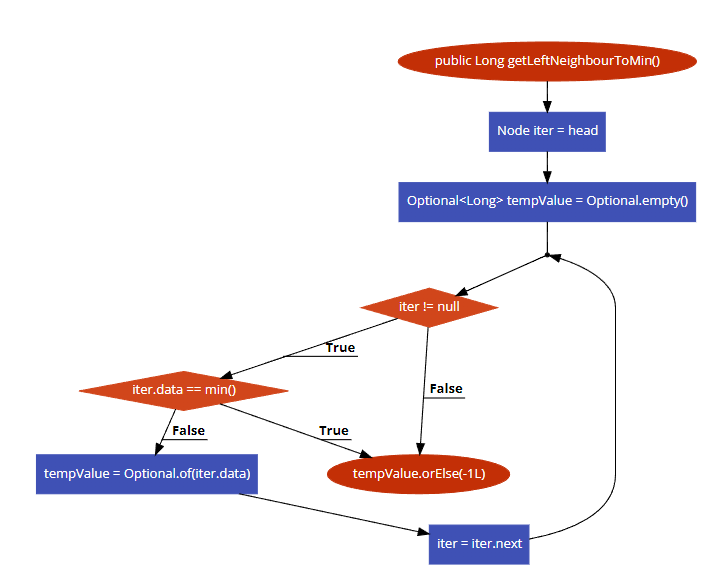
1. min(), max() – знаходження мінімального та максимального елементів.

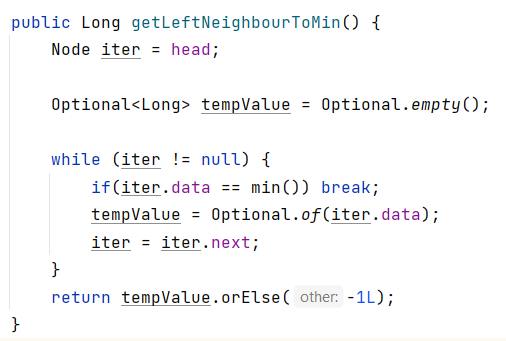


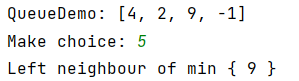




1. getLeftNeighbourToMin() – визначення сусіднього елемента зліва від мінімального.

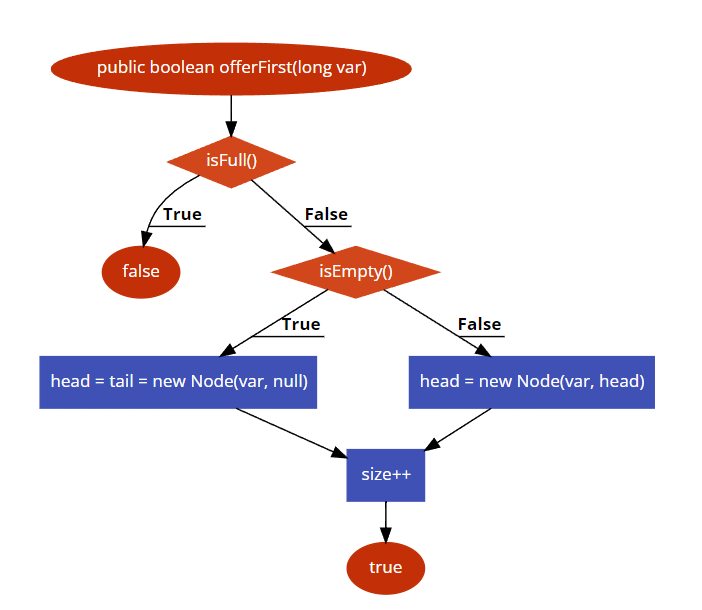


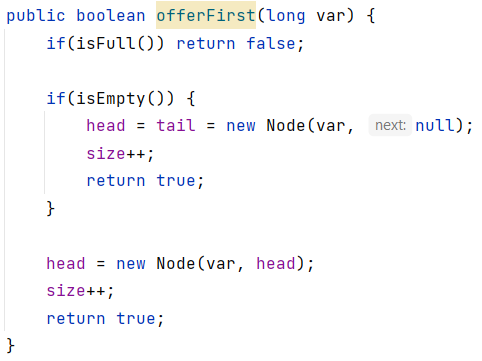


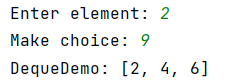


**Блок-схеми методів для роботи із деками:**

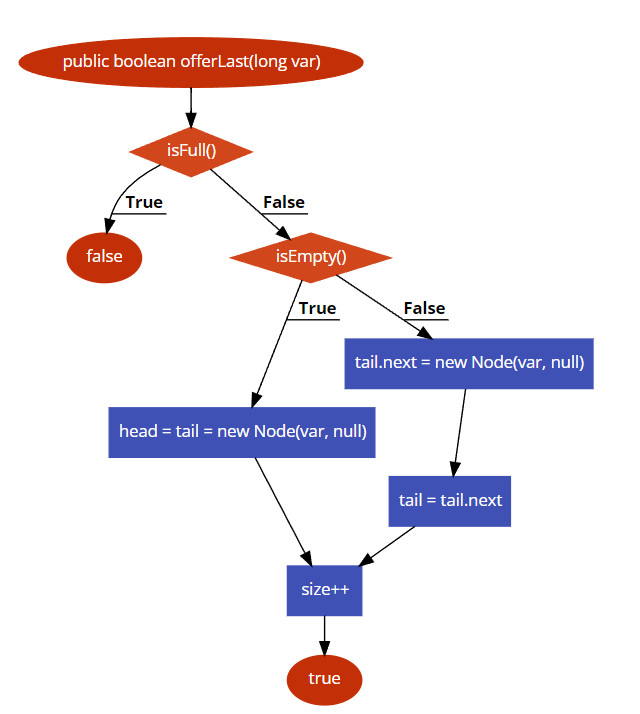
1. offerFirst(long var) – вставлення елемента на початок дека.

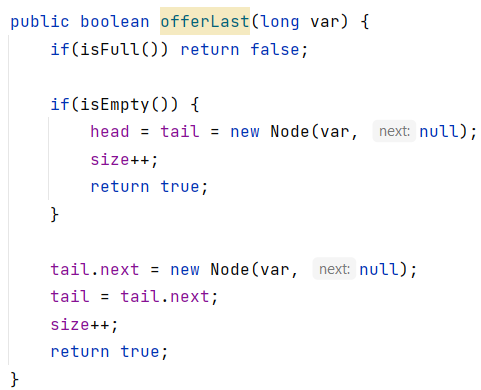


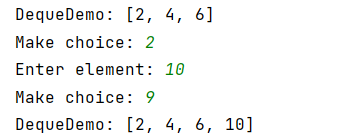




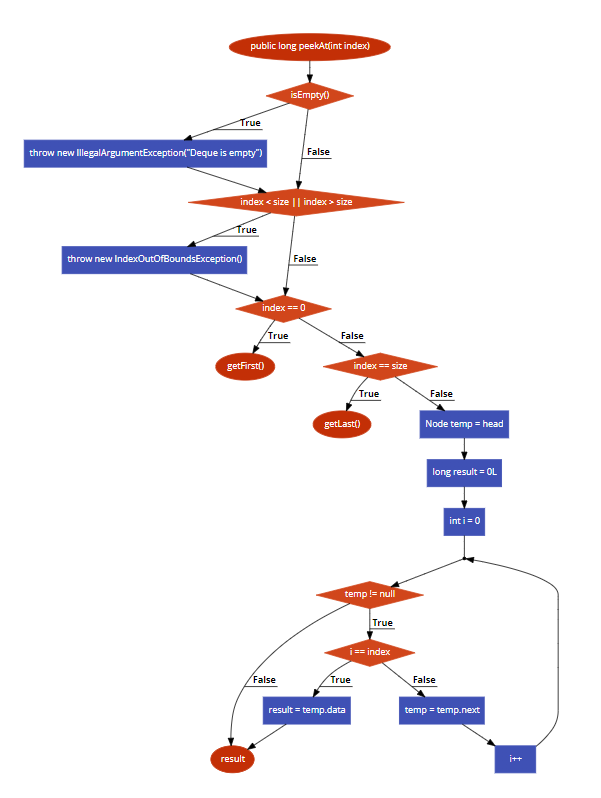
1. offerLast(long var) – вставлення елемента в кінець дека.

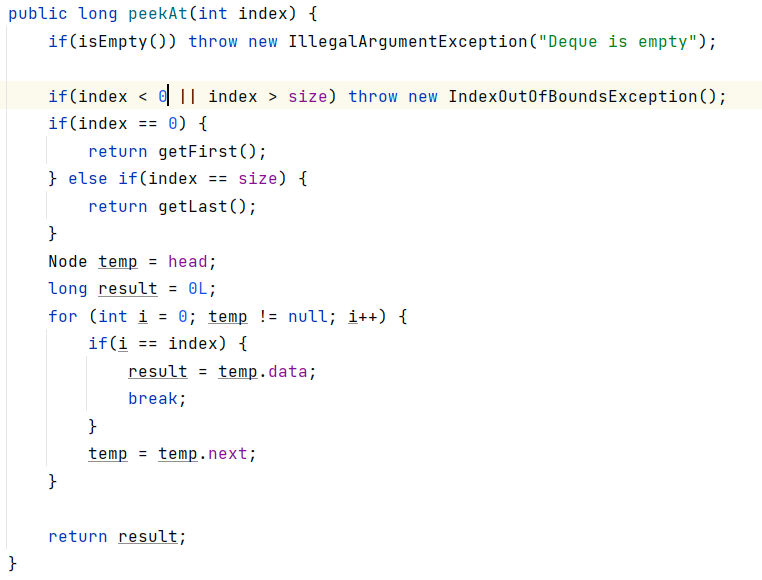


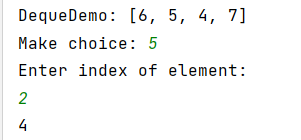




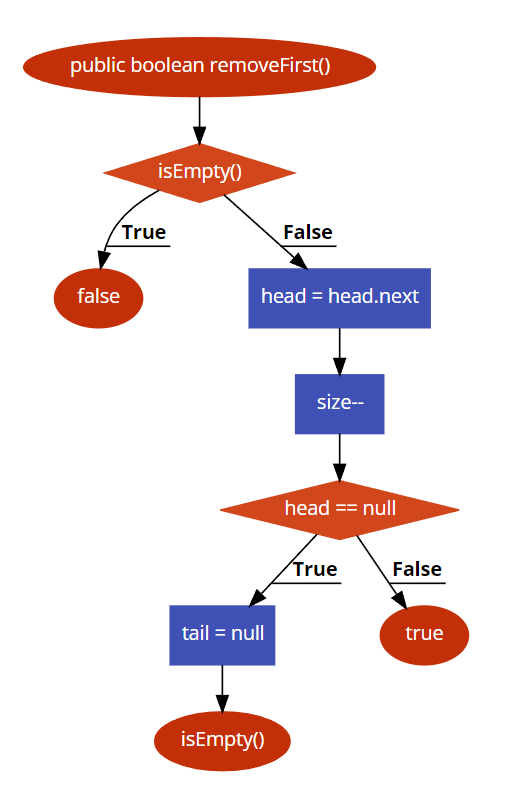
1. peekAt(int index) – отримання елемента на позиції по даному індексу.

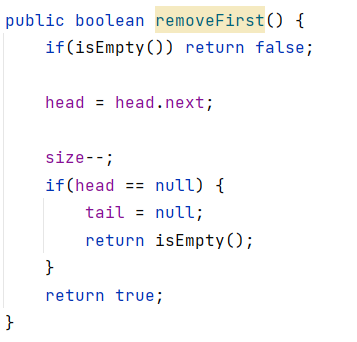


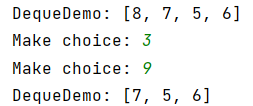




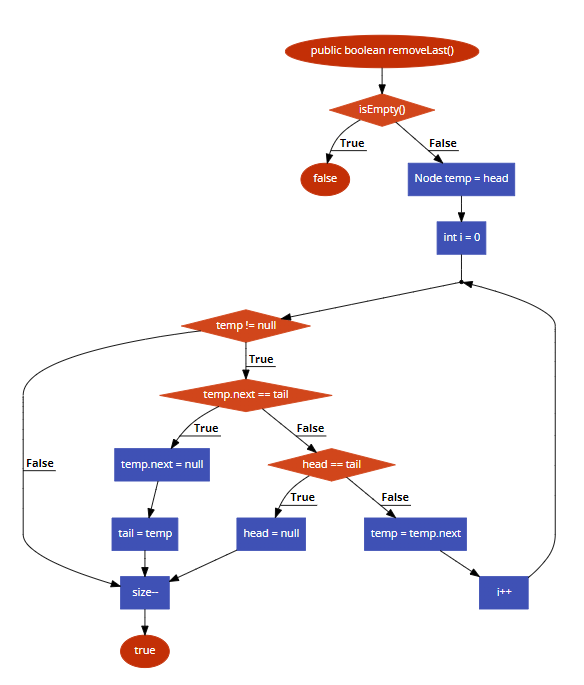
1. removeFirst() – видалення першого елемента.

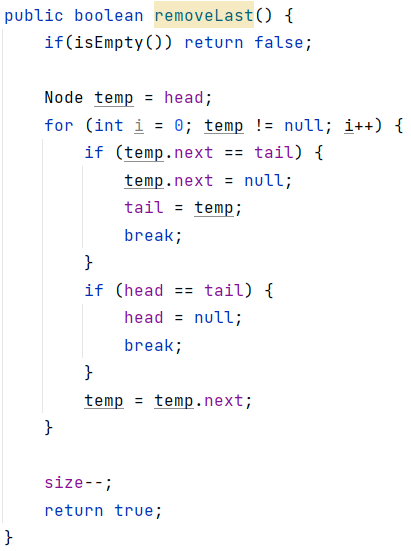


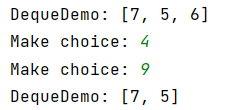




1. removeLast() – видалення останнього елемента.







**Висновки:**

На цій лабораторній роботі, я опанував навички у роботі із стеком, чергою та деком і їх основними методами.

**Стек** - це колекція, елементи якої отримують за принципом "останній

увійшов, перший вийшов" (Last-In-First-Out або LIFO) . Це означає, що ми

матимемо доступ лише до останнього доданого елементу.

На відміну від списків ми не можемо отримати доступ до довільного

елемента стека. Ми можемо лише додавати або видаляти елементи за

допомогою спеціальних методів. У стека немає також способу Contains, як у

списків. Крім того, стек не має ітератора. Для того, щоб розуміти, чому на

стек накладаються такі обмеження, давайте подивимося, як він працює і як

використовується.

Найчастіше зустрічається аналогія пояснення стека — стос тарілок.

Незалежно від того, скільки тарілок у стосі, ми завжди можемо зняти верхню. Чисті тарілки так само кладуться на верх стопки, і ми завжди будемо першою брати ту тарілку, яка була покладена останньою.

**Черги** дуже схожі на стеки. Вони також не дають доступу до довільного елементу, але, на відміну від стека, елементи кладуться (enqueue) і забираються (dequeue) з різних кінців. Такий метод називається "перший

увійшов, перший вийшов" (First-In-First-Out або FIFO) . Тобто забирати

елементи з черги ми будемо в тому самому порядку, що й клали. Як реальна

черга чи конвеєр.

Черги часто використовуються у програмах для реалізації буфера, у який

можна покласти елемент для подальшої обробки, зберігаючи порядок

надходження. Наприклад, якщо база даних підтримує лише одне з'єднання,

можна використовувати чергу потоків, які будуть, як не дивно, чекати на свою чергу на доступ до БД.

**Двостороння черга** (Double-ended queue) , або дек (Deque) , розширює поведінку черги. У дек можна додавати чи видаляти елементи як із початку, і з кінця черги. Така поведінка корисна у багатьох завданнях, наприклад, планування виконання потоків чи реалізація інших структур даних.