**Звіт**

Про виконання лабораторної роботи №1

З дисципліни «Структури даних та алгоритми»

**Виконав:**

Зимовець Руслан (МП-1, ІПЗ)

***Завдання на лабораторну роботу***

Розробити програму, яка читає з клавіатури послідовність N цілих чисел ( 1<N<256), жодне з яких не повторюється, зберігає їх до структури даних (згідно з завданням) та видає на екран такі характеристики: − сума елементів послідовності; 192 − середнє арифметичне елементів послідовності; − три мінімальні та максимальні елементи; − елемент послідовності з номером [N/2]. Усі перелічені характеристики потрібно визначити із заповненої структури даних. Дозволено використовувати лише ті операції, що притаманні заданій структурі. Наприклад, заборонено отримувати доступ до елемента із довільною позицією у черзі, яку реалізовано на базі масиву. Використовувати готові реалізації структур даних (наприклад, STL) заборонено. Указати складність алгоритмів для пошуку кожної характеристики.

Формат вхідних даних.

Перший рядок – число N , далі N цілих чисел, розділених пробілами.

A screenshot of a computer

Description automatically generated

Варіант: Черга + Стек

**Опис роботи:**

У межах даної роботи було реалізовано Стек та Чергу, базовані на динамічному масиві та зв’язному списку (загалом - 4 структури даних). Також визначено та протестовано наступні операції на обох структурах даних: сума елементів послідовності, середнє арифметичне елементів послідовності, знаходження трьох мінімальних та максимальних елементів, елемент послідовності з номером [N/2].

**Мета роботи:**

Метою роботи було не тільки розроблення Стеку та Черги як структур даних та операцій на них, а ще й порівняння продуктивності їх реалізацій на масивах та списках.

**Аналіз роботи:**

Було розроблено структури даних з наступними складностями їх базових операцій:

* Стек на базі списку:
  + Додавання елементу – **О(1)**
  + Вилучення елементу – **О(1)**
  + Отримання розміру – **О(1)**
  + Отримання основного елементу – **О(1)**
* Стек на базі масиву
  + Додавання елементу – **амортизована О(1) (О(N) – в найгіршому випадку [N – кількість елементів у стеці])**
  + Вилучення елементу – **О(1)**
  + Отримання розміру – **О(1)**
  + Отримання основного елементу – **О(1)**
* Черга на базі списку
  + Додавання елементу – **О(1)**
  + Вилучення елементу – **О(1)**
  + Отримання розміру – **О(1)**
  + Отримання основного елементу – **О(1)**
* Черга на базі масиву
  + Додавання елементу – **амортизована О(1) (О(N) – в найгіршому випадку [N – кількість елементів у стеці])**
  + Вилучення елементу – **О(1)**
  + Отримання розміру – **О(1)**
  + Отримання основного елементу – **О(1)**

Також було розроблено операції на розглянутих структурах даних, які мають наступні складності:

* сума елементів послідовності – **O(N) [N – кількість елементів у структурі даних]**
* середнє арифметичне елементів послідовності - **O(N) [N – кількість елементів у структурі даних]**
* знаходження трьох мінімальних та максимальних елементів - **O(N) [N – кількість елементів у структурі даних]**
* елемент послідовності з номером [N/2] - **O(N) [N – кількість елементів у структурі даних]**

Порівняння продуктивності («бенчмарк») базових операцій різних реалізацій розглянутих структур даних:

A screen shot of a computer program

Description automatically generated

З порівнянь продуктивності випливає, що стеки та черги реалізовані на основі масивів є значно більш швидкими за стеки та черги базовані на зв’язних списках. Це можна пояснити тим, що масиви вимагають меншої кількості алокацій (хоч і більших за розміром), ніж зв’язні списки. І, що напевно найголовніше, масиви розміщуються у пам’яті послідовно, тому ймовірність того, що більшість їх елементів буде зчитано у кеш процесора, є значно більшою, ніж у списків. У списків елементи розміщуються у випадкових ділянках пам’яті (за це відповідає системний алокатор), тому їх кешова локальність програє масивам.

**Висновок:**

Було розглянуто та проаналізовано різні реалізації Стеку та черги, складності їх базових операцій, розширені операції на них та їх складності. Також було порівняно продуктивність базових операцій розглянутих структур даних.