 МІНІСТЕРСТВО ОСВІТИ I НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ   
ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

ФАКУЛЬТЕТ БІОМЕДИЧНОЇ ІНЖЕНЕРІЇ

КАФЕДРА БІОМЕДИЧНОЇ КІБЕРНЕТИКИ

**Лабораторна робота №11**

з дисципліни «Алгоритмізація та програмування

на тему: «Стек і черга»

Варіант №

**Виконав:**

**Перевірив:**

Зараховано від \_\_\_.\_\_\_.\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис викладача)

Київ-2022

**Завдання**

1. Ознайомитися з теоретичними відомостями створення, обробки та видалення однозв’язних списків.

2. Відповідно до варіанту визначити структуру стеку або черги та розробити функції, що необхідні для роботи із заданим контейнером; функцію додавання елемента *push* (пам’ять під черговий елемент контейнера слід виділяти динамічно), функцію вилучення елемента *pop*, функцію друку змісту контейнера на екран, функцію видалення контейнера.

3. Розробити програмний застосунок, в якому створюється контейнер і заповнюється елементами, що в якості даних зберігають унікальні числа діапазоном від -50 до 50, та реалізовано виконання індивідуального завдання, після чого всі створені контейнери видаляються (глобальні змінні не використовувати, функція main має бути призначена тільки для виклику функцій користувача):

**Завдання за варіантом:** Визначити скільки елементів стека мають значення, що менші за середнє значення всіх елементів стека.

**Програмна реалізація:**

#include <iostream>

#include<cstdlib>

using namespace std;

struct stack

{

int data;

stack\* next;

};

void push(stack\*\* head, int data)

{

stack\* element = new stack;

element->data = data;

element->next = \*head;

\*head = element;

}

void pop(stack\*\* cur)

{

stack\* element = (\*cur);

(\*cur) = (\*cur)->next;

delete[] element;

}

void deleteStack(stack\*\* head)

{

stack\* element = new stack;

while ((\*head) != nullptr)

{

stack\* element = (\*head);

(\*head) = (\*head)->next;

delete[] element;

}

}

void displayStack(stack\* head)

{

cout << "The stack's elements are: ";

stack\* tmp = head;

while (tmp != nullptr)

{

cout << tmp->data << " ";

tmp = tmp->next;

}

}

double findAvrg(stack\* head)

{

float sum = 0, i = 0;

stack\* tmp = head;

while (tmp != nullptr)

{

sum += tmp->data;

i++;

tmp = tmp->next;

}

float res = sum / i;

return res;

}

int countLowerThen(stack\* head, float num)

{

stack\* tmp = head;

int i = 0;

while (tmp != nullptr)

{

if (tmp->data < num)

i++;

tmp = tmp->next;

}

return i;

}

int main()

{

stack\* head = nullptr;

int n;

srand((unsigned)time(NULL));

cout << "How much elements do you want to be in a stack?: ";

cin >> n;

for (int i = 0; i < n; i++)

push(&head, (rand() % 100) - 50);

displayStack(head);

double num = findAvrg(head);

int count = countLowerThen(head, num);

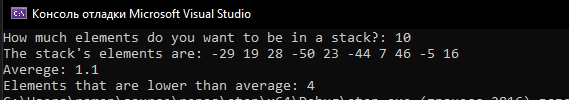
cout << endl << "Averege: " << num << endl << "Elements that are lower than average: " << count;

deleteStack(&head);

return 0;

}

**Результат виконання програми:**



**Контрольні питання:**

**1. Що таке список? Як здійснюється зв'язок між елементами списку?**

Список — це шаблон класу контейнерів послідовностей, який зберігає свої елементи в лінійному упорядкуванні та дозволяє виконувати операції вставки та видалення у будь-якому розташуванні в послідовності.

У С++ список реалізується завдяки структурам, в яких є необхідна для зберігання інформація, посилання на наступний/попередній/наступний і попередній елементи. Зв’язок між елементами здійснюється завдяки посиланням.

**2. Що є ознаками кінця та порожнечі списку різних типів?**

Останнім елементом однозв’язного списку може бути:

* Покажчик, що зберігає нульове значення(в однозв’язному лінійному списку та в двозв’язному списку)
* Покажчик, що вказує на перший елемент списку(кільцевий однозв’язний список)
* Покажчик next останнього елемента зберігає адресу первого елемента, а значення покажчика Previous першого елемента зберігає адресу останнього елмента списку(двозв’язний кільцевий список)

**3. Опишіть алгоритм виведення на екран елементів лінійного та кільцевого списків.**

Для виведення елементів лінійного списку нам потрібно взяти початковий елемент, вивести його значення, отримати за вказівником наступний елемент, вивести його значення тощо. Якщо вказівник наступного елементу буде нульовим – зупинити виведення.

Для виведення елементів кільцевого списку нам потрібно зберегти вказівник на поточний (умовно перший/останній) елемент і виводити усі елементи, поки не дійдемо до цього елемента (перевірку робити по вказівнику).

**4. Поясніть принцип вставки та видалення елементів у однозв'язному та двозв'язному списках.**

У однозв’язному списку вставка відбувається створенням нового елементу списку та створенням всередині цього елементу вказівки на попередній елемент списку. Видалення відбувається тимчасовим збереженням вказівки на наступний елемент, видаленням поточного елементу та заміною вказівки попереднього елементу(на наступний) на тимчасово збережену вказівку. Якщо елемент крайній, то просто видаленням елементу, та заміною вказівки на наступний елемент на нульове значення, якщо елемент останній.

У двозв’язному списку створення нового елементу відбувається так само, як і в однозв’зяному, але ми у цьому елементу ще й зберігаємо вказівку на попередній елемент. Видаляємо також скожим чином за вийнятком того, що ми тимчасово зберігаємо вказівку ще й на попередній елемент та замінюємо в наступному елементі вказівку на попередній тимчасовою вказівкою.

**5. Які операції зі списками найчастіше виконуються на практиці?**

Додавання елементу, видалення елементу, прохід по списку, очищення списку.

**6. Чим відрізняється організація списків типу «стек» і «черга»?**

Стек працює за принципом LIFO («last in – first out») - останній прийшов – перший вийшов.

Черга працює за принципом FIFO («first in – first out») - перший прийшов – перший вийшов.

**7. Які операції можливі над стеком та над чергою?**

Ті самі, що і для звичайних списків.

**8. Для моделювання яких реальних процесів зручно використовувати стек, а для яких чергу?**

**Стек:** виконання рекурсії, управління динамічною пам’яттю, калькулятори, де запис виразів організовано в інверсній польській нотації.

**Черга:** послідовність обробки подій в операційних системах, буферизація завдань принтера в системі вхідних і вихідних потоків, організація буфера введення з клавіатури, моделювання процесів обслуговування клієнті тощо.

**9. Чи можливо організувати стек та чергу не на основі зв'язного списку? Поясніть.**

Можна створити на основі динамічного масиву, але тоді буде витрачатись значна кількість ресурсів на оновлення довжини масиву, тому краще реалізовувати стек та чергу на основі зв’язного списку.

**10. В чому застосовуються принципи стеку та черги в процесі виконання програмного коду?**

Програмний вид стека використовується для обходу структур даних, наприклад, дерево чи граф. При використанні рекурсивних функцій також застосовуватиметься стек, але апаратний його вигляд.

Зазвичай черга використовується для задач, де потрібно буде накопичувати відповідь. Наприклад потужні калькулятори, величини при обрахунках яких набагато більші за можливу пам’ять однієї змінної.