ДНІПРОВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ОЛЕСЯ ГОНЧАРА
Факультет прикладної математики та інформаційних технологій
Кафедра обчислювальної математики та математичної кібернетики

Лабораторна робота №2

з дисципліни «Алгоритми та структури даних»

на тему «Черга і стек»

Варіант №4

Виконала:
студентка групи ПС-24-1
Анна Гарт

Дніпро, 2025

Відповіді на контрольні запитання.

1. Структура даних, робота з елементами якої організована за принципом FIFO (перший прийшов - перший пішов), це черга.

2. Структура даних, робота з елементами якої організована за принципом FILO (першим прийшов - останнім пішов), це стек.

3. Доступ лише до верхнього елемента (вершини). Операції додавання (push) та видалення (pop) відбуваються з одного кінця. Принцип роботи: LIFO/FILO. Використовується для відстеження викликів функцій, парсингу виразів, відновлення попередніх станів.

4. Видаляється завжди останній доданий елемент (верхівка стека). Операція називається pop.

5. Так, стек можна реалізувати на основі однозв'язного списку, де додавання та видалення елементів відбувається з початку списку (це ефективно, бо не потрібно проходити весь список).

6. Додавання елементів відбувається з одного кінця (хвіст), а видалення – з іншого (голова). Принцип роботи: FIFO. Використовується для планування завдань, буферизації, обробки запитів.

7. Так, можна. Двозв'язний список дозволяє ефективно додавати елементи в кінець та видаляти з початку, що відповідає принципу черги. Також можна легко реалізувати додавання/видалення з обох кінців (двостороння черга, deque).

8. Це черга (queue). Черга гарантує, що перший доданий елемент буде оброблений першим, що ідеально підходить для буферизації завдань або запитів, які чекають на обробку (наприклад, в системах чергування запитів до БД).

Мета виконання завдання: навчитися створювати та опрацьовувати структури даних «черга» і «стек», виконувати операції додавання, вилучення та перегляду елементів, а також закріпити практичні навички роботи з цими структурами.

Постановка завдання. Варіант 4. Розробити програму створення й опрацювання черги і стек. Створити стек з цілих непарних чисел від 3 до 55. Всі числа з черги витягувати по одному і записувати в стек. Числа зі стека знімати по одному і виводити на екран ті з них, що кратні 3. Зобразити створений список у протоколі лабораторної роботи. Схематично зобразити вставлення та вилучення елементів у чергу і стек.

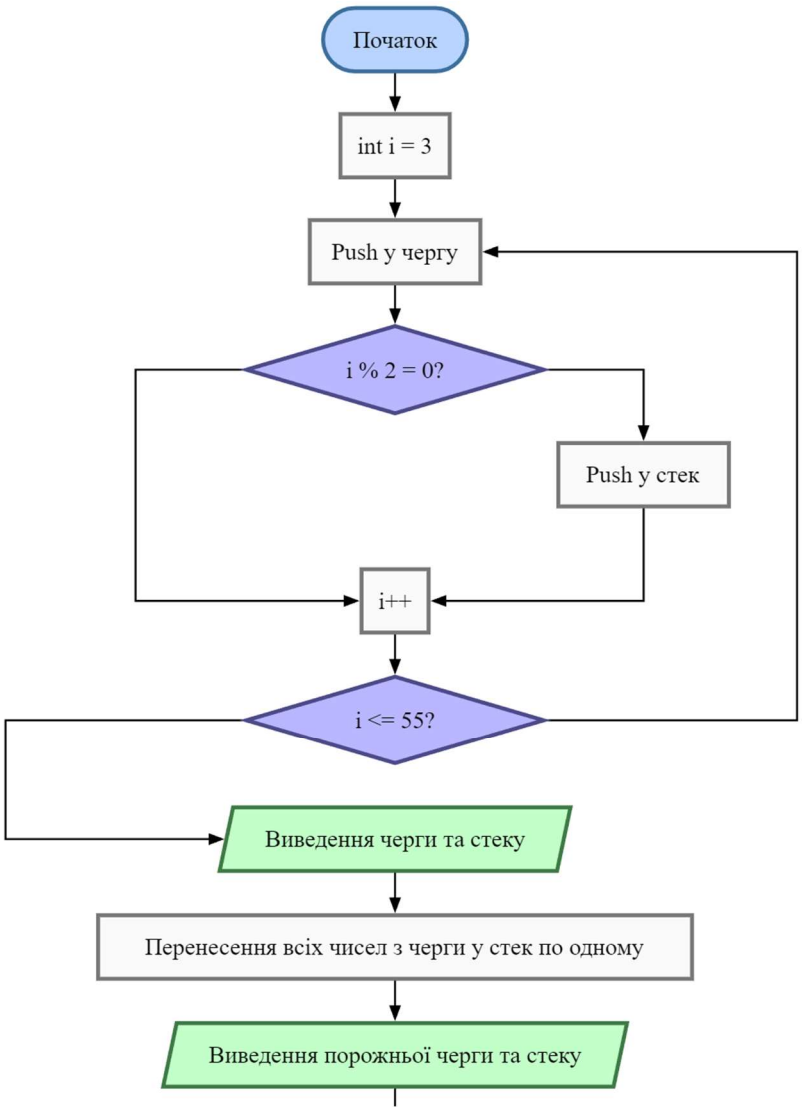
Текст програми:

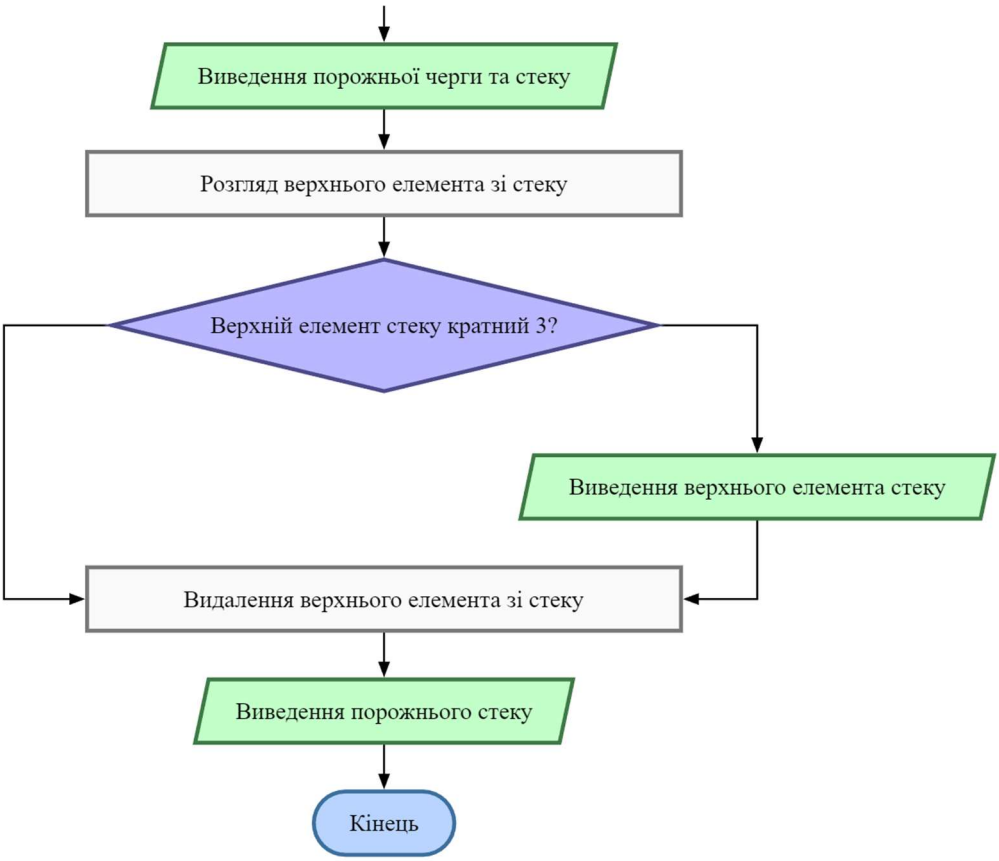
```
/* Гарт Анна Вікторівна, група ПС-24-1, варіант №4
 * Лабораторна робота №2. Черга і стек */

#include <iostream>
```

```cpp
#include <stack>   // для роботи зі стеком
#include <queue>   // для роботи з чергою
using namespace std;

// Виведення черги
void PrintQueue(queue<int> q) {
    cout << "Черга: ";
    if (q.empty()) {
        cout << "порожня\n";
        return;
    }
    else {
        while (!q.empty()) {
            cout << q.front() << " ";
            q.pop();
        }
        cout << "\n";
    }
}

// Виведення стеку
void PrintStack(stack<int> st) {
    cout << "Стек: ";
    if (st.empty()) {
        cout << "порожній\n";
        return;
    }
    else {
        while (!st.empty()) {
            cout << st.top() << " ";
            st.pop();
        }
        cout << "\n";
    }
}

int main() {

    stack<int> st; // стек
    queue<int> q;  // черга

    system("chcp 65001");
    system("cls");

    for (int i = 3; i <= 55; i++) {
        q.push(i);      // додаємо цілі числа від 3 до 55 у чергу
        if (i%2 != 0) {
            st.push(i); // додаємо цілі непарні числа від 3 до 55 у стек
        }
    }
    cout << "\nДодавання цілих чисел від 3 до 55 у чергу" << endl;
    PrintQueue(q);
    cout << "\nДодавання цілих непарних чисел від 3 до 55 у стек" << endl;
    PrintStack(st);

    // Переносимо всі числа з черги у стек по одному
    cout << "\nПеренесення всіх чисел з черги у стек по одному" << endl;
    while (!q.empty()) {
        st.push(q.front()); // додаємо перший елемент черги у стек
```

```
        q.pop();            // видаляємо його з черги
    }
    PrintQueue(q);
    PrintStack(st);

    // Знімаємо всі числа зі стеку по одному і виводимо ті, що кратні 3
    cout << "\nЗнімання всіх чисел зі стеку по одному і виведення тих, що кратні
3:" << endl;
    while (!st.empty()) {
        if (st.top()%3 == 0) {
            cout << st.top() << " "; // виводимо верхній елемент стека
        }
        st.pop();                    // видаляємо його зі стека
    }
    cout << "\n";
    PrintStack(st);

    cout << "\n" << endl;
    system("pause");
    return 0;
}
```

Алгоритм:





Для побудови алгоритму було використано сайт: https://www.yworks.com/yed-live

Результат роботи програми:

```
D:\Projects\Sem3_Algorithms\

Додавання цілих чисел від 3 до 55 у чергу
Черга: 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34
 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55

Додавання цілих непарних чисел від 3 до 55 у стек
Стек: 55 53 51 49 47 45 43 41 39 37 35 33 31 29 27 25 23 21 19 17 15 13 11 9 7 5 3

Перенесення всіх чисел з черги у стек по одному
Черга: порожня
Стек: 55 54 53 52 51 50 49 48 47 46 45 44 43 42 41 40 39 38 37 36 35 34 33 32 31 30 29 28 27 26
 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 55 53 51 49 47 45 43 41 39 37 35
 33 31 29 27 25 23 21 19 17 15 13 11 9 7 5 3

Знімання всіх чисел зі стеку по одному і виведення тих, що кратні 3:
54 51 48 45 42 39 36 33 30 27 24 21 18 15 12 9 6 3 51 45 39 33 27 21 15 9 3
Стек: порожній


Press any key to continue . . .
```

Висновок. У ході виконання роботи ми навчилися створювати та опрацьовувати чергу і стек, виконувати базові операції додавання та вилучення елементів, а також переносити дані між цими структурами. Було побудовано стек непарних чисел, перенесено елементи з черги у стек та відібрано ті з них, що кратні 3. Програма працює коректно й демонструє практичне застосування черги і стека, поставлену мету лабораторної роботи виконано.