Міністерство освіти і науки України
Національний університет «Львівська політехніка»
Кафедра систем штучного інтелекту

Звіт

про виконання лабораторних та практичних робіт блоку № 6
На тему: «Програмування: алгоритм, програма, код. Системи числення. Двійкова система числення. Розробка та середовище розробки програми.»
з дисципліни: «Основи програмування»
до:
Практичних Робіт до блоку № 6

Виконав:
Студент групи ШІ-11
Бубельник Юрій Олегович

Львів 2024

Тема: Динамічні структури (Черга, Стек, Списки, Дерево). Алгоритми обробки динамічних структур.

Мета: Засвоїти основи роботи з динамічними структурами даних, такими як черга, стек, списки та дерева. Ознайомитися з алгоритмами їх обробки для розв'язання різноманітних задач.

Теоретичні відомості та джерела:

Динамічні структури

-Урок №89. Динамічне виділення пам’яті;
-Урок №90. Динамічні масиви;

-Стек;
C++ • Теорія • Урок 58 • Стек, Куча, Статична пам'ять
https://www.youtube.com/watch?v=B3VHHfMW0Pg

-Черга;
#4
#5

-Списки:
#1
#2
Урок #133
Урок #134
Урок #135

-Дерева:
#3
C++ • Теорія • Урок 144 • ADT • Бінарне дерево

Виконання роботи:

VNS Lab 10v1:

Написати програму, у якій створюються динамічні структури й виконати їхню обробку у відповідності зі своїм варіантом.

Для кожного варіанту розробити такі функції:

1. Створення списку.
2. Додавання елемента в список (у відповідності зі своїм варіантом).
3. Знищення елемента зі списку (у відповідності зі своїм варіантом).
4. Друк списку.
5. Запис списку у файл.
6. Знищення списку.
7. Відновлення списку з файлу.

4. Варіанти завдань

1. Записи в лінійному списку містять ключове поле типу int. Сформувати однонаправлений список. Знищити з нього елемент із заданим номером, додати елемент із заданим номером;

Algotester Lab 5v2:

В пустелі існує незвичайна печера, яка є двохвимірною. Її висота це N, ширина - M.
Всередині печери є пустота, пісок та каміння. Пустота позначається буквою O , пісок S і каміння X;
Одного дня стався землетрус і весь пісок посипався вниз. Він падає на найнижчу клітинку з пустотою, але він не може пролетіти через каміння.
Ваше завдання сказати як буде виглядати печера після землетрусу.

Algotester Lab 7-8 v1:

Class Practice Task:

Ваше завдання - власноруч реалізувати структуру даних "Двозв’язний список".
Ви отримаєте Q запитів, кожен запит буде починатися зі слова-ідентифікатора, після якого йдуть його аргументи.

Вам будуть поступати запити такого типу:

Вставка:

Ідентифікатор - insert

Ви отримуєте ціле число index елемента, на місце якого робити вставку.

Після цього в наступному рядку рядку написане число N - розмір списку, який треба вставити.

У третьому рядку N цілих чисел - список, який треба вставити на позицію index.

Видалення:

Ідентифікатор - erase

Ви отримуєте 2 цілих числа - index, індекс елемента, з якого почати видалення та nn - кількість елементів, яку треба видалити.

Визначення розміру:

Ідентифікатор - size

Ви не отримуєте аргументів.

Ви виводите кількість елементів у списку.

Отримання значення i-го елементу

Ідентифікатор - get

Ви отримуєте ціле число - index, індекс елемента.

Ви виводите значення елемента за індексом.

Модифікація значення ii-го елементу

Ідентифікатор - set

Ви отримуєте 2 цілих числа - індекс елемента, який треба змінити, та його нове значення.

Вивід списку на екран

Ідентифікатор - print

Ви не отримуєте аргументів.

Ви виводите усі елементи списку через пробіл.

Реалізувати використовуючи перегрузку оператора <<

Class Practice Task:

Задача №1 - Реверс списку (Reverse list)

Реалізувати метод реверсу списку: Node* reverse(Node *head);
Умови задачі:

- використовувати цілочисельні значення в списку;
- реалізувати метод реверсу;
- реалізувати допоміжний метод виведення вхідного і обернутого списків;

Задача №2 - Порівняння списків

bool compare(Node *h1, Node *h2);
Умови задачі:

- використовувати цілочисельні значення в списку;
- реалізувати функцію, яка ітеративно проходиться по обох списках і порівнює дані в кожному вузлі;
- якщо виявлено невідповідність даних або якщо довжина списків різна (один список закінчується раніше іншого), функція повертає ***false***.

Задача №3 – Додавання великих чисел

Node* add(Node *n1, Node *n2);
Умови задачі:

- використовувати цифри від 0 до 9 для значень у списку;
- реалізувати функцію, яка обчислює суму двох чисел, які збережено в списку; молодший розряд числа записано в голові списка (напр. 379 $\Longrightarrow$ 9→7→3);
- функція повертає новий список, передані в функцію списки не модифікуються.

Задача №4 - Віддзеркалення дерева

TreeNode *create_mirror_flip(TreeNode *root);
Умови задачі:

- використовувати цілі числа для значень у вузлах дерева
- реалізувати функцію, що проходить по всіх вузлах дерева і міняє місцями праву і ліву вітки дерева

- функція повертає нове дерево, передане в функцію дерево не модифікується

Задача №5 - Записати кожному батьківському вузлу суму підвузлів

void tree_sum(TreeNode *root);

Умови задачі:

- використовувати цілочисельні значення у вузлах дерева;
- реалізувати функцію, яка ітеративно проходить по бінарному дереві і записує у батьківський вузол суму значень підвузлів
- вузол-листок не змінює значення
- значення змінюються від листків до кореня дерева

Self Practice

Task Algotester Lab 5v2:

У вас є карта гори розміром N×M.
Також ви знаєте координати {x,y} , у яких знаходиться вершина гори.
Ваше завдання - розмалювати карту таким чином, щоб найнижча точка мала число 0, а пік гори мав найбільше число.
Клітинкі які мають суміжну сторону з вершиною мають висоту на один меншу, суміжні з ними і не розфарбовані мають ще на 1 меншу висоту і так далі.

Код програми з посиланням на зовнішні ресурси

VNS Lab 10v1: затратність ~3 години
ai_11/yurii_bubelnyk/epic_6/Code/vns_lab_10_task_1_variant_1_yurii_bubelnyk.cpp

Algotester Lab 5v2: затратність ~ 2 години
ai_11/yurii_bubelnyk/epic_6/Code/algotester_lab_5_variant_2_yurii_bubelnyk.cpp

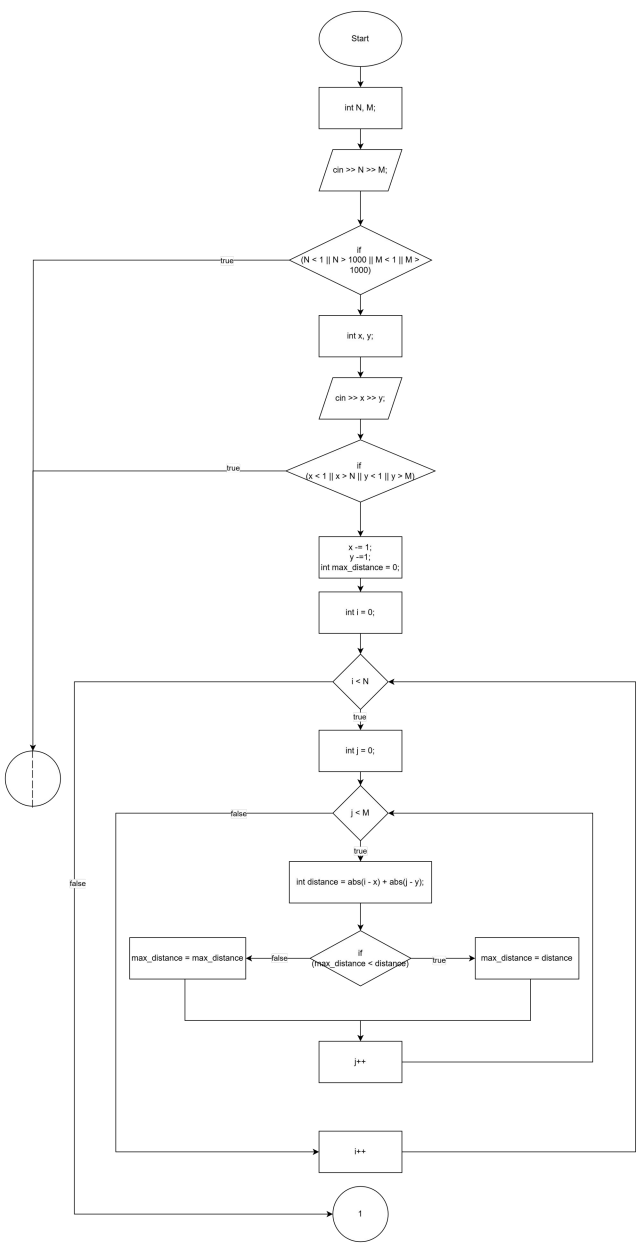
Algotester Lab 7-8 v1: затратність ~ 6 години
ai_11/yurii_bubelnyk/epic_6/Code/algotester_lab_7_8_variant_1_yurii_bubelnyk.cpp

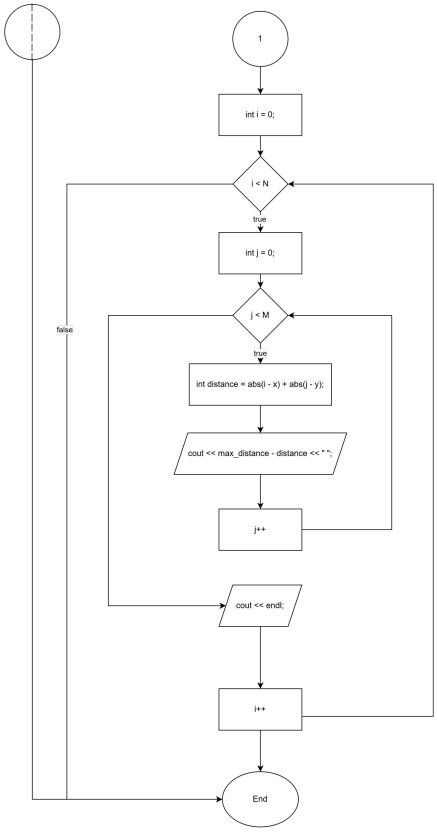
Class Practice Task: затратність ~ 4 години
ai_11/yurii_bubelnyk/epic_6/Code/practice_work_task_1_yurii_bubelnyk.cpp
ai_11/yurii_bubelnyk/epic_6/Code/practice_work_task_4-5_yurii_bubelnyk.cpp

Task Algotester Lab 5v3: затратність ~ 2 години
ai_11/yurii_bubelnyk/epic_6/Code/algotester_practice_work_Lab-5v3_yurii_bubelnyk.drawio

SelfPractice A prime number: ~30 хвилин
ai_11/yurii_bubelnyk/epic_6/Code/pRIME.cpp

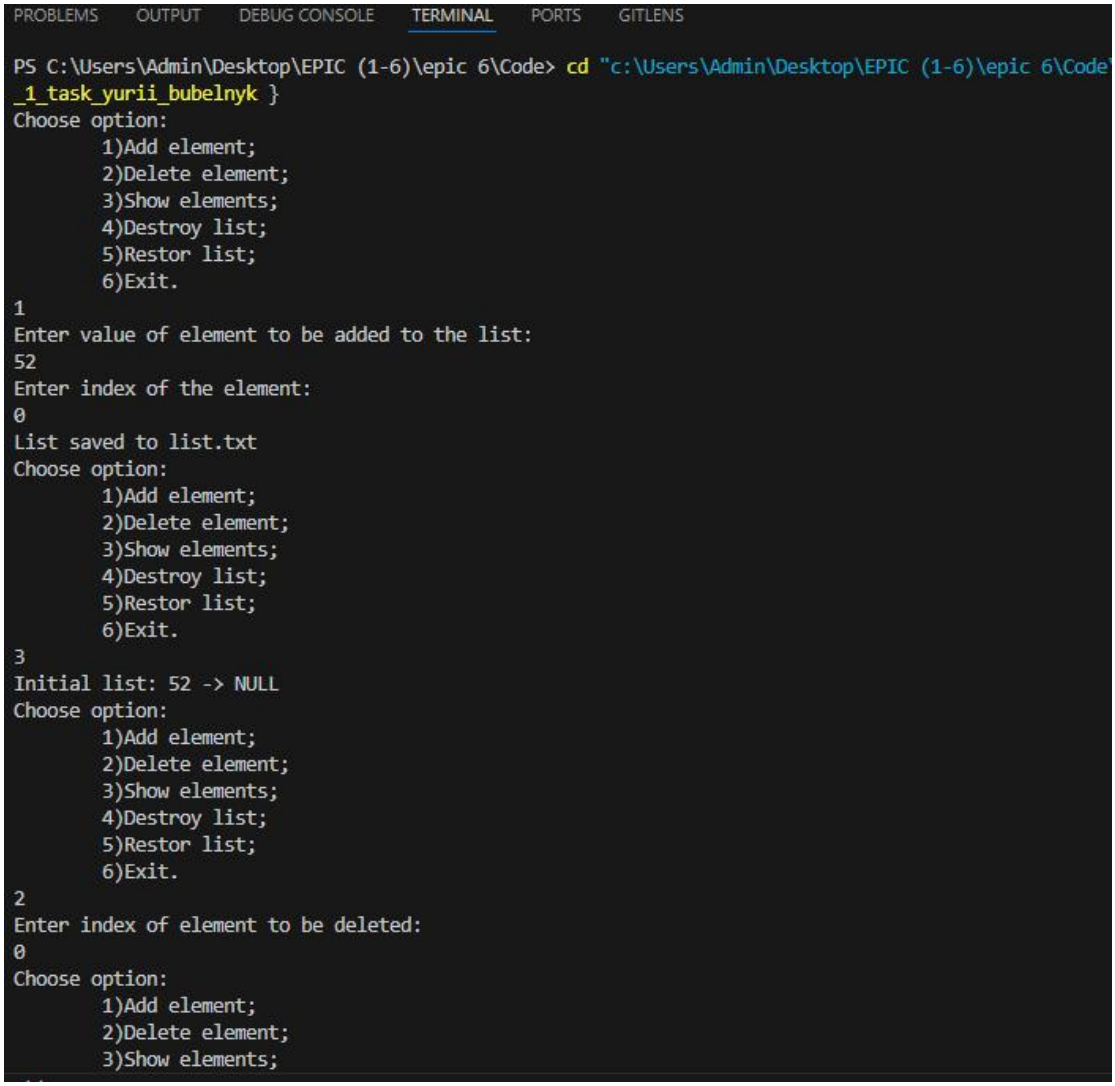
Block-scheme for tast Algotester Lab 5v3





Результати виконаних завдань, тестування та фактично затрачений час

VNS Lab 10v1: ~4 години



```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  GITLENS

PS C:\Users\Admin\Desktop\EPIC (1-6)\epic 6\Code> cd "c:\Users\Admin\Desktop\EPIC (1-6)\epic 6\Code"
_1_task_yurii_bubelnyk }
Choose option:
        1)Add element;
        2)Delete element;
        3)Show elements;
        4)Destroy list;
        5)Restor list;
        6)Exit.
1
Enter value of element to be added to the list:
52
Enter index of the element:
0
List saved to list.txt
Choose option:
        1)Add element;
        2)Delete element;
        3)Show elements;
        4)Destroy list;
        5)Restor list;
        6)Exit.
3
Initial list: 52 -> NULL
Choose option:
        1)Add element;
        2)Delete element;
        3)Show elements;
        4)Destroy list;
        5)Restor list;
        6)Exit.
2
Enter index of element to be deleted:
0
Choose option:
        1)Add element;
        2)Delete element;
        3)Show elements;
```

```
Choose option:
        1)Add element;
        2)Delete element;
        3)Show elements;
        4)Destroy list;
        5)Restor list;
        6)Exit.
5
List restored from list.txt
Choose option:
        1)Add element;
        2)Delete element;
        3)Show elements;
        4)Destroy list;
        5)Restor list;
        6)Exit.
3
Initial list: 52 -> NULL
Choose option:
        1)Add element;
        2)Delete element;
        3)Show elements;
        4)Destroy list;
        5)Restor list;
        6)Exit.
4
List destroyed.
Choose option:
        1)Add element;
        2)Delete element;
        3)Show elements;
        4)Destroy list;
        5)Restor list;
        6)Exit.
```

Algotester Lab 5v2: ~2 години

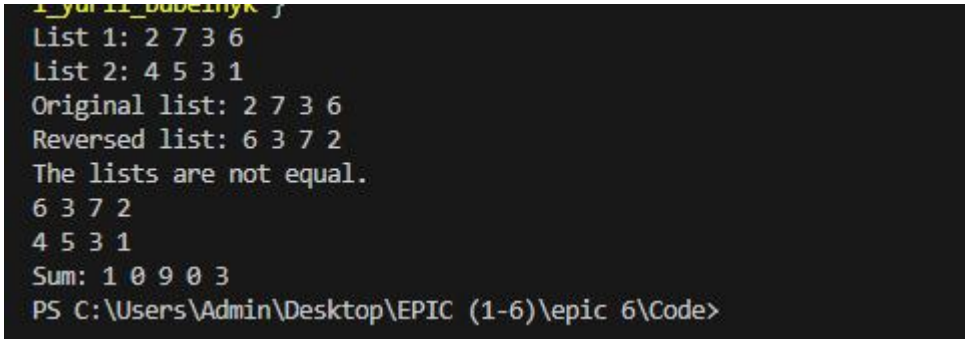
```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\Admin\Desktop\EPIC (1-6)\epic 6\Code> cd "
ter_lab_5_variant_2_yurii_bubelnyk }
3 5
SOSOO
XSOSS
OOXOO
SOOOO
XOSOO
OSXSS
PS C:\Users\Admin\Desktop\EPIC (1-6)\epic 6\Code>
```

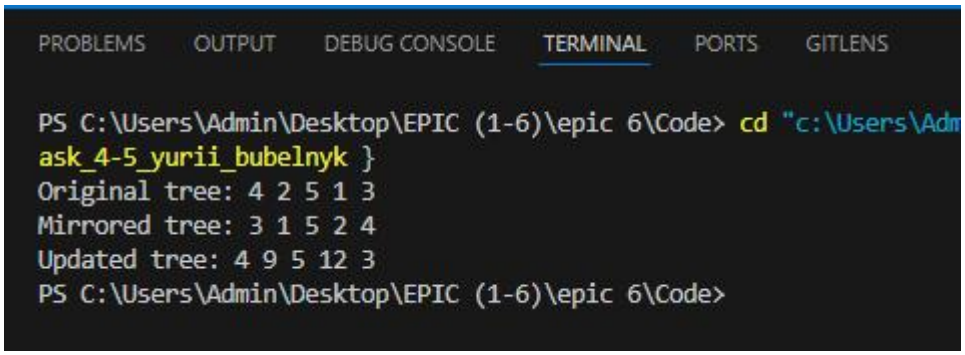
Created	Compiler	Result	Time (sec.)	Memory (MiB)	Actions
5 days ago	C++ 23	Accepted	0.037	1.898	View

Algotester Lab 7-8 v1: ~5 годин

2 days ago	C++ 23	Accepted	0.008	1.422	View

Class Practice Task: ~3 години

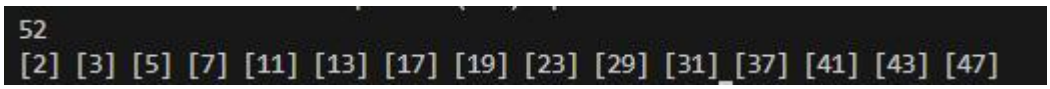




Task Algotester Lab 5v3: ~3 години

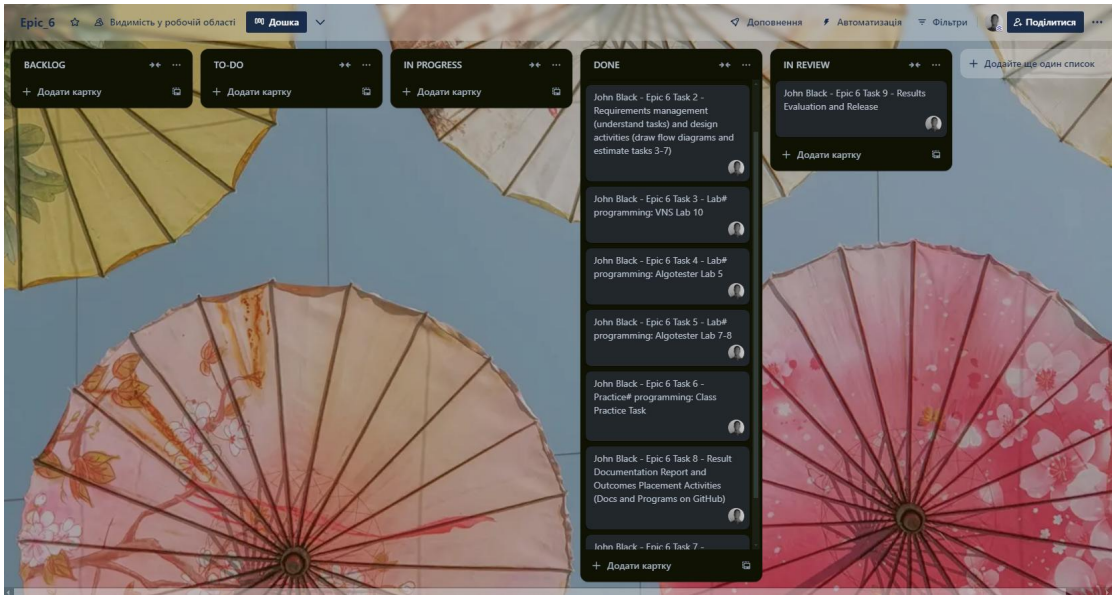
Created	Compiler	Result	Time (sec.)	Memory (MiB)	Actions
3 days ago	C++ 23	Accepted	0.099	4.531	View
3 days ago	C++ 23	Accepted	0.097	4.527	View

SelfPractice A prime number: ~20 хвилин



Робота з командою:

Налаштували Trello для Epic 6:



Висновки:

Отже, в межах цього епіка я старався зрозуміти, що таке списки, дерева та як їх реалізовувати в коді. Практикувався з записом даних у файли, а також покращив роботу з масивами та алгоритмами.

Посилання на pull request