Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Кафедра систем штучного інтелекту



**Звіт**

про виконання

**Лабораторних та практичних робіт № 6**

***з дисципліни:*** «Мови та парадигми програмування»

***з розділу***: «Динамічні структури (Черга, Стек, Списки, Дерево). Алгоритми обробки динамічних структур»

***Виконав(ла):***

студент групи ШІ-12

Стешенко Юрій Євгенович

# **Тема роботи:**

Динамічні структури (Черга, Стек, Списки, Дерево). Алгоритми обробки

динамічних структур

# **Мета роботи:**

Опрацювати та засвоїти динамічні структури і їх алгоритми

# **Теоретичні відомості:**

**Тема №1: Черга (Queue)**

Черга - це динамічна структура даних, що дозволяє здійснювати доступ до елементів в порядку "перший прийшов, перший обслугований" (FIFO).

Джерела Інформації:

Книжка: "Data Structures and Algorithms" by Michael T. Goodrich, Roberto Tamassia, Michael H. Goldwasser.

Відео: "Introduction to Queues" on YouTube.

Стаття: "Queue Data Structure" на GeeksforGeeks.

Курс: "Data Structures and Algorithms Specialization" на Coursera.

Що опрацьовано:

Ознайомлення з базовими операціями черги: enqueue (додати елемент), dequeue (вилучити елемент), front (отримати елемент на передньому кінці).

Розгляд основних застосувань черг у програмуванні.

Статус: Ознайомлений частково.

Початок опрацювання теми: 19.12.2023.

Звершення опрацювання теми: 22.12.2023.

**Тема №2: Стек (Stack)**

Стек - це динамічна структура даних, що дозволяє здійснювати доступ до елементів в порядку "останній прийшов, перший обслугований" (LIFO).

Джерела Інформації:

Книжка: "Algorithms" by Robert Sedgewick, Kevin Wayne.

Відео: "Stack Data Structure" on YouTube.

Стаття: "Stack Data Structure" на GeeksforGeeks.

Курс: "Data Structures and Algorithms" на edX.

Що опрацьовано:

Ознайомлення з операціями стеку: push (додати елемент), pop (вилучити елемент), top (отримати елемент на вершині).

Розгляд практичних застосувань стеків в програмуванні.

Статус: Ознайомлений.

Початок опрацювання теми: 19.12.2023.

Звершення опрацювання теми: 22.12.2023.

**Тема №3: Списки (Lists)**

Список - це динамічна структура даних, яка дозволяє зберігати та управляти послідовністю елементів.

Джерела Інформації:

Книжка: "Introduction to Algorithms" by Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein.

Відео: "Linked Lists" on YouTube.

Стаття: "Linked List Data Structure" на GeeksforGeeks.

Курс: "Data Structures" на Udacity.

Що опрацьовано:

Ознайомлення з різними видами списків: зв'язаний список, масив, двозв'язний список.

Розгляд операцій над списками: вставка, видалення, пошук елементів.

Статус: Ознайомлений частково.

Початок опрацювання теми: 19.12.2023.

Звершення опрацювання теми: 22.12.2023.

**Тема №4: Дерево (Tree)**

Дерево - це ієрархічна структура даних, що складається з вузлів, з'єднаних гілками, де кожен вузол може мати декілька дочірніх вузлів.

Джерела Інформації:

Книжка: "Data Structures and Algorithms in Python" by Michael T. Goodrich, Roberto Tamassia, Michael H. Goldwasser.

Відео: "Introduction to Trees" on YouTube.

Стаття: "Tree Data Structure" на GeeksforGeeks.

Курс: "Algorithms and Data Structures" на Khan Academy.

Що опрацьовано:

Ознайомлення з основними поняттями: корінь, листя, вузол, рівень, висота.

Розгляд різних видів дерев: бінарне дерево, бінарне дерево пошуку, AVL-дерево.

Статус: Ознайомлений частково.

Початок опрацювання теми: 19.12.2023.

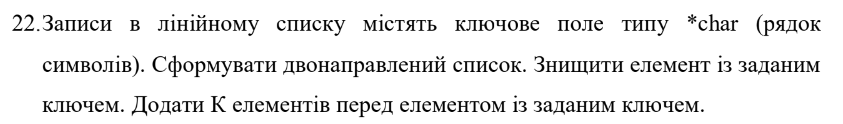
Звершення опрацювання теми: 22.12.2023.

# **Виконання роботи:**

## **1. Опрацювання завдання та вимог до програм та середовища:**

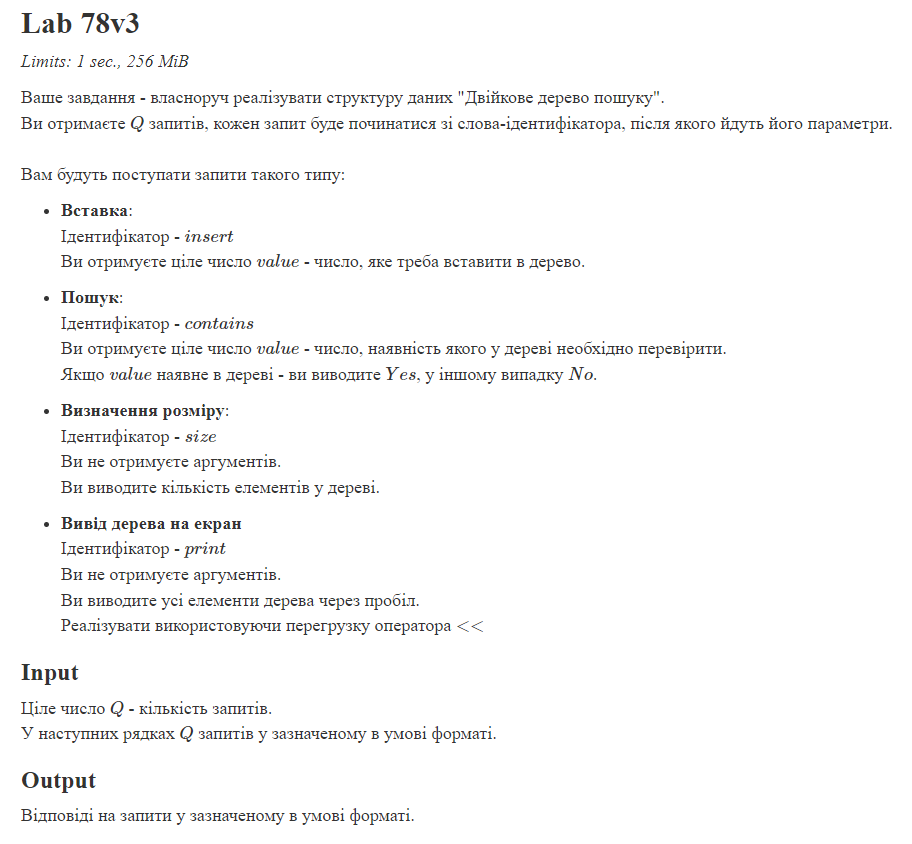
Завдання №1 VNS lab 10

* Варіант завдання 22
* Деталі завдання



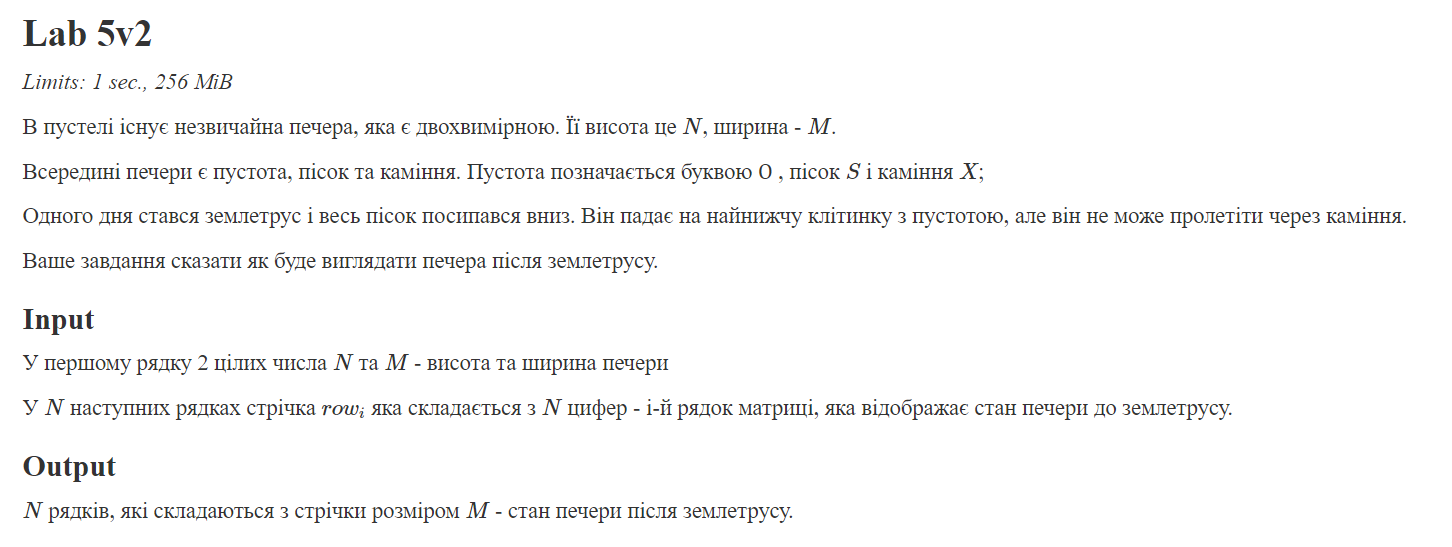
Завдання №2 Algotester Lab 7-8

* Варіант завдання - 3
* Деталі завдання



Завдання №3 Algotester Lab 5

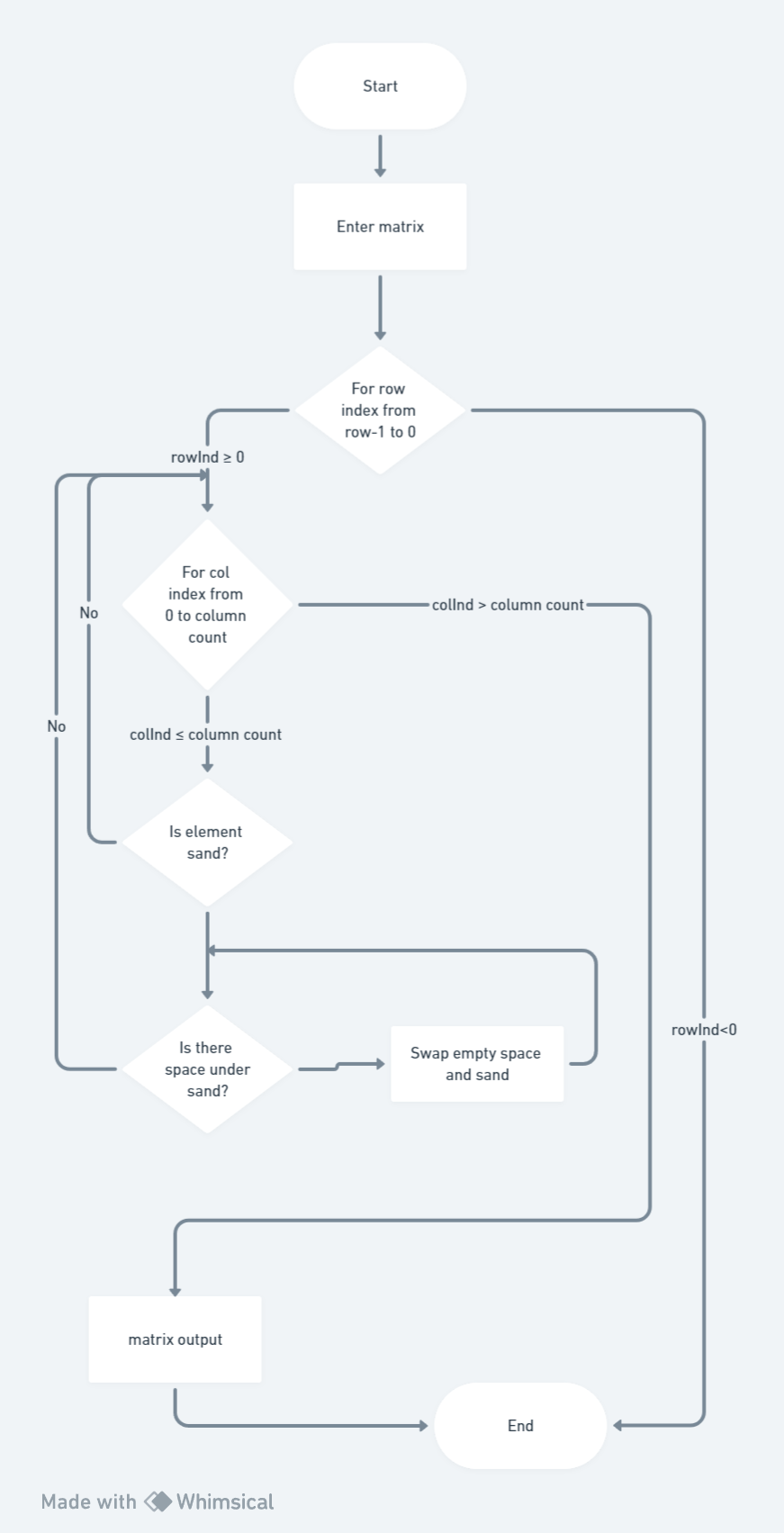
* Варіант завдання - 2
* Деталі завдання



## **2. Дизайн та планована оцінка часу виконання завдань:**

Програма №1 Algo lab 5v2

* Блок-схема



* Планований час на реалізацію - 20 хвилин

## **3. Код програм з посиланням на зовнішні ресурси:**

Завдання №1 VNS lab 10

[Посилання на файл програми у пул-запиті GitHub](https://github.com/artificial-intelligence-department/ai_programming_playground/pull/1310/files#diff-91baacfd6f43eadf633ecea5c50e1431920331f2ff717a93d3191187992be352)

#include <iostream>

#include <ctime>

using namespace std;

struct Node

{

int Value;

Node \*Previous;

Node \*Next;

Node(int value)

{

Value = value;

Previous = NULL;

Next = NULL;

}

};

class Linked\_list

{

Node \*start;

Node \*end;

public:

Linked\_list()

{

start = end = NULL;

}

void add(int value)

{

Node \*newNode = new Node(value);

if (!start)

start = end = newNode;

else

{

end->Next = newNode;

newNode->Previous = end;

end = newNode;

}

}

void Delete(int token)

{

Node \*node = start;

while (node != NULL)

{

if (node->Value == token)

{

if (node->Previous)

node->Previous->Next = node->Next;

else

start = node->Next;

if (node->Next)

node->Next->Previous = node->Previous;

else

end = node->Previous;

delete node;

break;

}

node = node->Next;

}

}

void AddBeforeToken(int token, int count)

{

Node \*node = start;

while (node)

{

if (node->Value == token)

{

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

{

Node \*newNode = new Node(0);

newNode->Next = node;

newNode->Previous = node->Previous;

if (node->Previous)

node->Previous->Next = newNode;

else

start = newNode;

node->Previous = newNode;

}

break;

}

node = node->Next;

}

}

void Print()

{

Node \*node = start;

while (node)

{

cout << node->Value << " ";

node = node->Next;

}

cout << "\n";

}

Node \*GetNode(int Position)

{

Node \*node = start;

for (int i = 0; i < Position && node != NULL; i++)

node = node->Next;

return node;

}

};

int main()

{

Linked\_list list;

srand(time(0));

for (int j = 0; j < 10; j++)

{

list.add(rand() % 20 + 1);

}

cout << "List: ";

list.Print();

list.Delete(list.GetNode(5)->Value);

cout << "List: ";

list.Print();

list.AddBeforeToken(list.GetNode(7)->Value, 7);

cout << "List: ";

list.Print();

return 0;

}

Завдання №2 Algotester Lab 7-8

[Посилання на файл програми у пул-запиті GitHub](https://github.com/artificial-intelligence-department/ai_programming_playground/pull/1310/files#diff-d5d9b0b7ea61d3f6558a9bab08ec3e5f78bbff9021765ef2f8b890b51536ed73)

#include <iostream>

using namespace std;

template <class Type>

class binary\_tree

{

private:

struct \_node

{

public:

Type val;

struct \_node \*\_upper;

struct \_node \*\_left;

struct \_node \*\_right;

\_node(Type val, \_node \*upper, \_node \*left, \_node \*right)

: val(val), \_upper(upper), \_left(left), \_right(right) {}

\_node() : \_left(NULL), \_right(NULL), \_upper(NULL) {}

\_node(Type val)

: \_node(val, NULL, NULL, NULL) {}

};

// Helper for finalizer

void finalize(\_node \*\_root)

{

if (\_root == NULL)

return;

finalize(\_root->\_left);

finalize(\_root->\_right);

delete \_root;

}

\_node \*find(Type val)

{

\_node \*current = treeRoot;

while (current != NULL)

{

if (current->val == val)

return current;

if (val < current->val)

current = current->\_left;

else

current = current->\_right;

}

return NULL;

}

void putNode(\_node \*node, \_node \*\_root)

{

if (node == NULL)

return;

if (\_root == NULL)

{

treeRoot = node;

treeSize++;

return;

}

if (node->val < \_root->val)

{

if (\_root->\_left == NULL)

{

\_root->\_left = node;

node->\_upper = \_root;

treeSize++;

}

else

putNode(node, \_root->\_left);

}

else if (node->val > \_root->val)

{

if (\_root->\_right == NULL)

{

\_root->\_right = node;

node->\_upper = \_root;

treeSize++;

}

else

putNode(node, \_root->\_right);

}

}

void destroyNode(\_node \*node)

{

if (node->\_upper == NULL)

{

treeRoot = NULL;

\_node \*left = node->\_left, \*right = node->\_right;

delete node;

if (left != NULL)

{

treeRoot = left;

left->\_upper = NULL;

putNode(right, treeRoot);

}

else if (right != NULL)

{

treeRoot = right;

right->\_upper = NULL;

}

treeSize--;

return;

}

if (node->\_upper->\_left == node)

node->\_upper->\_left = NULL;

else

node->\_upper->\_right = NULL;

\_node \*left = node->\_left, \*right = node->\_right;

delete node;

treeSize--;

putNode(left, treeRoot);

putNode(right, treeRoot);

}

void printToStream(ostream &os, \_node \*currentParent)

{

if (!currentParent)

return;

printToStream(os, currentParent->\_left);

os << currentParent->val << ' ';

printToStream(os, currentParent->\_right);

}

\_node \*treeRoot;

int treeSize;

public:

binary\_tree()

{

treeRoot = NULL;

treeSize = 0;

}

~binary\_tree()

{

finalize(treeRoot);

}

int get\_size()

{

return treeSize;

}

void put(Type val)

{

putNode(new \_node(val), treeRoot);

}

bool has(Type val)

{

return find(val) != NULL;

}

void destroy(Type val)

{

\_node \*node = find(val);

if (node == NULL)

return;

destroyNode(node);

}

friend ostream &operator<<(ostream &os, binary\_tree<Type> &t)

{

t.printToStream(os, t.treeRoot);

return os;

}

};

int main()

{

binary\_tree<int> \_tree;

int Q;

string line;

cin >> Q;

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

{

cin >> line;

if (line == "insert")

{

int element;

cin >> element;

\_tree.put(element);

}

else if (line == "size")

{

cout << \_tree.get\_size() << endl;

}

else if (line == "print")

{

cout << \_tree << "\n";

}

else if (line == "contains")

{

int element;

cin >> element;

cout << (\_tree.has(element) ? "Yes" : "No") << endl;

}

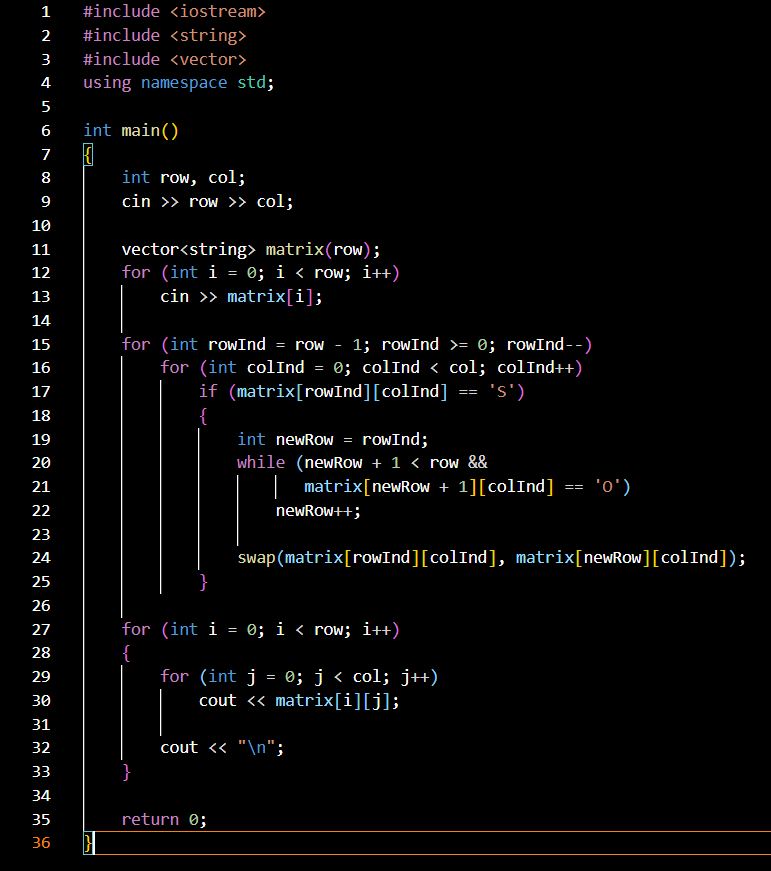
}

return 0;

}

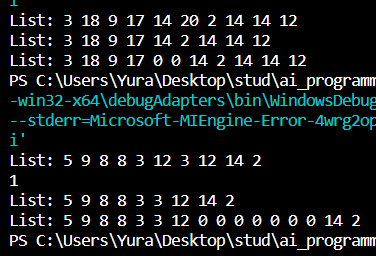
Завдання №3 Algotester Lab 5

[Посилання на файл програми у пул-запиті GitHub](https://github.com/artificial-intelligence-department/ai_programming_playground/pull/1310/files#diff-6c7e944a87a33274e995f8087cc06fc6c2ca1ad92605f8b1d4f0d20258172c36)

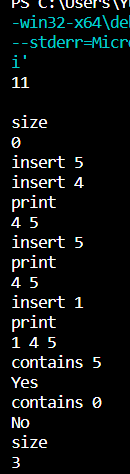


## **4. Результати виконання завдань, тестування та фактично затрачений час:**

Завдання №1 [VNS lab 10](https://github.com/artificial-intelligence-department/ai_programming_playground/pull/1310/files#diff-91baacfd6f43eadf633ecea5c50e1431920331f2ff717a93d3191187992be352)



Завдання №2 [Algotester Lab 7-8](https://github.com/artificial-intelligence-department/ai_programming_playground/pull/1310/files#diff-d5d9b0b7ea61d3f6558a9bab08ec3e5f78bbff9021765ef2f8b890b51536ed73)



Завдання №3 [Algotester Lab 5](https://github.com/artificial-intelligence-department/ai_programming_playground/pull/1310/files#diff-6c7e944a87a33274e995f8087cc06fc6c2ca1ad92605f8b1d4f0d20258172c36)

