Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Кафедра систем штучного інтелекту



**Звіт**

**про виконання лабораторних та практичних робіт блоку № 6**

На тему:  «Динамічні структури (Черга, Стек, Списки, Дерево). Алгоритми обробки динамічних структур.»

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

до:

ВНС Лабораторної Роботи № 10

Алготестер Лабораторної Роботи № 5

Алготестер Лабораторної Роботи № 7-8

Практичних Робіт № 6

**Виконала:**

Студентка групи ШІ-13

Гошовська Діана Олегівна

# **Тема роботи:**

Ознайомлення з теорією про динамічні структури. Дослідження алгоритмів обробки динамічних структур. Робота з бінарними деревами.

# **Мета роботи:**

Ознайомитися з навчальною теорією про динамічні структури, роботу з їх типами та принцип їхньої роботи в коді, використавши для цього доступні джерела інформації; закріпити отриманні знання на практичних заняттях; оформити звіт до лабораторної роботи №6.

# **Теоретичні відомості:**

1. Теоретичні відомості з переліком важливих тем:

* Тема №1: Ознайомлення з теорією про динамічні структури.
* Тема №2: Дослідження алгоритмів обробки динамічних структур.

1. Індивідуальний план опрацювання теорії:
   * Тема №1: Ознайомлення з теорією про динамічні структури.
   * Джерела Інформації
     + Документ: <https://vns.lpnu.ua/pluginfile.php?file=%2F1116827%2Fmod_resource%2Fcontent%2F1%2Fkonspekt_lekcyi_ukr_1.pdf>
   * Що опрацьовано:
     + Опрацьовано джерела, що містять інформацію про динамічні структури та методи їх застосування у коді.
     + Закріплено отриману інформацію, використавши її для виконання завдань.

* Статус: Ознайомлена
  + Початок опрацювання теми: 28.11.2023
  + Звершення опрацювання теми: 06.12.2023
* Тема №2: Дослідження алгоритмів обробки динамічних структур.
  + Джерела Інформації:
    - Документ: <https://vns.lpnu.ua/pluginfile.php?file=%2F1116827%2Fmod_resource%2Fcontent%2F1%2Fkonspekt_lekcyi_ukr_1.pdf>
  + Що опрацьовано:
    - Опрацьовано інформацію про роботу динамічних структур та їх імплементацію в програмі.
    - Закріплено отриману інформацію, використавши її для виконання завдань.
  + Статус: Ознайомлена
  + Початок опрацювання теми: 28.11.2023
  + Звершення опрацювання теми: 06.12.2023

# **Виконання роботи:**

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

Завдання №3: написання коду до VNS Lab 10

* Варіант завдання: 9
* Деталі завдання:   
  Записи в лінійному списку містять ключове поле типу int. Сформувати

двонаправлений список. Знищити з нього К елементів перед елементом із

заданим номером, додати К елементів у кінець списку.

* Важливі деталі для врахування в імплементації програми: зрозуміти принцип роботи з двонаправленими списками та пояснити отримані результати.

Завдання №4: написання коду до Algotester Lab 5v2

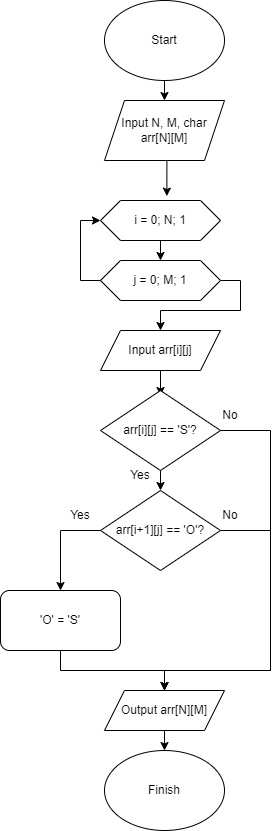
* Варіант завдання: 2
* Деталі завдання: В пустелі існує незвичайна печера, яка є двохвимірною. Її висота це N, ширина - M. Всередині печери є пустота, пісок та каміння. Пустота позначається буквою О, пісок S і каміння X. Одного дня стався землетрус і весь пісок посипався вниз. Він падає на найнижчу клітинку з пустотою, але він не може пролетіти через каміння. Ваше завдання сказати як буде виглядати печера після землетрусу.
* Важливі деталі для врахування в імплементації програми: виконати завдання та пояснити отримані результати.
* 

Figure 1: Блок схема

Завдання №5: написання коду до Algotester Lab 78v2

* Варіант завдання: 2
* Деталі завдання: <https://algotester.com/uk/ContestProblem/DisplayWithEditor/134648>
* Важливі деталі для врахування в імплементації програми: дотримуватися умов завдання.

Завдання №6: написання коду до Class Practice Task

* Деталі завдання: написати код, використовуючи закріплені раніше знання про створення динамічних структур та роботу з ними.
* Важливі деталі для врахування в імплементації програми: використати здобуті знання на практиці.

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

Програма №3: VNS Lab 10

* Планований час на реалізацію: 4 години.
* Важливі деталі для врахування в імплементації: використати динамічні структури для виконання завдання.

Програма №4: Algotester Lab 5v2

* Планований час на реалізацію: 3 години.
* Важливі деталі для врахування в імплементації: дотримуватися умов завдання.

Програма №5: Algotester Lab 78v2

* Планований час на реалізацію: 6 годин.
* Важливі деталі для врахування в імплементації: дотримуватися умов завдання.

Програма №6: Class Practice Task

* Планований час на реалізацію: 4 години.
* Важливі деталі для врахування в імплементації: дотримуватися умов завдання.

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

Завдання №3: VNS Lab 10

https://github.com/artificial-intelligence-department/ai\_programming\_playground/pull/1041/files#diff-3eb801dbdd6284ab6dea0b86bdc8194d39dde45b55ff85563a7bca7fe6f5afa8

#include <iostream>

using namespace std;

struct Node {

    int data;

    Node\* prev;

    Node\* next;

    Node(int value) : data(value), prev(nullptr), next(nullptr) {}

};

void appendNode(Node\*& head, int value) {

    Node\* newNode = new Node(value);

    if (!head) {

        head = newNode;

    } else {

        Node\* temp = head;

        while (temp->next) {

            temp = temp->next;

        }

        temp->next = newNode;

        newNode->prev = temp;

    }

}

void processList(Node\*& head, int targetIndex, int K) {

    if (!head) {

        cout << "The list is empty." << endl;

        return;

    }

    Node\* temp = head;

    int currentIndex = 0;

    while (temp && currentIndex != targetIndex) { //пошук елемент зі заданим номером

        temp = temp->next;

        currentIndex++;

    }

    if (temp) { //елемент знайдено

        for (int i = 0; i < K && temp->prev; ++i) { //видалення К елементів перед елементом

            Node\* toDelete = temp->prev;

            temp->prev = toDelete->prev;

            if (toDelete->prev) {

                toDelete->prev->next = temp;

            }

            delete toDelete;

        }

        for (int i = 0; i < K; ++i) { //додавання елементів у кінець списку

            int newValue;

            cout << "Enter a value for each element: ";

            cin >> newValue;

            appendNode(head, newValue);

        }

    } else {

        cout << "Element is not found." << endl;

    }

}

void printList(const Node\* head) {

    const Node\* temp = head;

    while (temp) {

        cout << temp->data << " ";

        temp = temp->next;

    }

    cout << endl;

}

int main() {

    Node\* head = nullptr;

    for (int i = 1; i <= 10; ++i) {

        appendNode(head, i);

    }

    cout << "Output list: ";

    printList(head);

    int targetIndex, K;

    cout << "Enter a number of element and a quantity of elements to change: ";

    cin >> targetIndex >> K;

    processList(head, targetIndex, K);

    cout << "Result: ";

    printList(head);

    return 0;

}

Ввід програми VNS Lab 10

Завдання №4: Algotester Lab 5v2

https://github.com/artificial-intelligence-department/ai\_programming\_playground/pull/1041/files#diff-07ce95f68d2b9470d9bd60bf2c6895e312eeac8d38e387c487c3ab2fa7d067cd

#include <iostream>

#include <string>

using namespace std;

int main(){

    int N, M;

    cin >> N >> M;

    char cave[N][M];

    for (int i = 0; i < N; i++){

        for (int j = 0; j < M; j++){

            cin >> cave[i][j];

        }

    }

    cout << endl;

    for (int j = 0; j < M; j++){

        for (int i = N-1; i > -1; i--){

            if (cave[i][j] == 'S'){ //пісок, що рухається

                if (cave[i+1][j] == 'O'){ //клітинка під піском вільна

                    cave[i+1][j] = 'S';

                    cave[i][j] = 'O';

                    i=N-1;

                }

            }

        }

    }

    for (int i = 0; i < N; i++){

        for (int j = 0; j < M; j++){

            cout << cave[i][j];

        }

        cout << endl;

    }

    return 0;

}

Ввід програми Algotester Lab 5v2

Завдання №5: Algotester Lab 78v2

https://github.com/artificial-intelligence-department/ai\_programming\_playground/pull/1041/files#diff-d6b23e47cf3a4c1dddbba7f96c999b84a7e15b8453bba33f983906bb3032e840

#include <iostream>

#include <string>

using namespace std;

template <typename nameType>

struct templateArray {

    int size = 0, capacity = 1;

    nameType\* Structure\_1;

    templateArray() {

        Structure\_1 = new nameType[1];

    }

    void enterElements(int index, int mainarray[], int Q) {

        if (capacity > size + Q) {

            for (int i = size - 1; i > index - 1; i--) {

                Structure\_1[i + Q] = Structure\_1[i];

            }

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

                Structure\_1[i + index] = mainarray[i];

            }

            size += Q;

        }

        else {

            while (capacity <= size + Q) {

                capacity = capacity \* 2;

            }

            nameType\* Structure\_2 = new nameType[capacity];

            for (int i = 0; i < index; i++) {

                Structure\_2[i] = Structure\_1[i];

            }

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

                Structure\_2[i + index] = mainarray[i];

            }

            for (int i = index; i < size; i++) {

                Structure\_2[i + Q] = Structure\_1[i];

            }

            delete[] Structure\_1;

            Structure\_1 = Structure\_2;

            size += Q;

        }

    }

    void Erase(int index, int q) {

        for (int i = index; i < size; ++i) {

            Structure\_1[i] = Structure\_1[i + q];

        }

        size -= q;

    }

    int Size() {

        return size;

    }

    int Capacity() {

        return capacity;

    }

    nameType& operator[](int index) {

        return Structure\_1[index];

    }

    friend ostream& operator<<(ostream& os, const templateArray& array) {

        for (int i = 0; i < array.size; ++i) {

            os << array.Structure\_1[i] << ' ';

        }

        return os;

    }

};

int main() {

    templateArray<int> Structure\_1;

    int H;

    cin >> H;

    for (int i = 0; i < H; i++) {

        string trigger;

        cin >> trigger;

        if (trigger == "insert") {

            int index,Q;

            cin >> index >> Q;

            int\* values = new int[Q];

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

                cin >> values[i];

            }

            Structure\_1.enterElements(index, values, Q);

        }

        else if (trigger == "erase") {

            int index, q;

            cin >> index >> q;

            Structure\_1.Erase(index, q);

        }

        else if (trigger == "size") {

            cout << Structure\_1.Size() << endl;

        }

        else if (trigger == "capacity") {

            cout << Structure\_1.Capacity() << endl;

        }

        else if (trigger == "get") {

            int index;

            cin >> index;

            cout << Structure\_1[index] << endl;

        }

        else if (trigger == "set") {

            int index;

            cin >> index;

            cin >> Structure\_1[index];

        }

        else if (trigger == "print") {

            cout << Structure\_1 << endl;

        }

    }

    return 0;

}

Ввід програми Algotester Lab 78v2

Завдання №6: Class Practice Task 1

https://github.com/artificial-intelligence-department/ai\_programming\_playground/pull/1041/files#diff-41a336f29daf4c8abb52fe5c5fe2b7e7fcc1f84c3979acf772ec5ef01c7ab464

#include <iostream>

using namespace std;

struct Node {

    int data; //всі елементи списку є int

    Node\* next; //вказівник на наступний елемент

    Node(int value) : data(value), next(nullptr) {} //конструктор data - значення елемента

};

void printList(Node\* head) {

    while (head) {

        cout << head->data << " ";

        head = head->next;

    }

    cout << endl;

}

Node\* reverse(Node\* head) {

    Node\* prev = nullptr;

    Node\* current = head;

    Node\* next = nullptr;

    while (current != nullptr) {

        next = current->next; //next привласнює вказівник next, що знаходиться в current

        current->next = prev; //current отримав вказівник попереднього і записав його як вказівник наступного елементу

        prev = current; //вказівник current став prev

        current = next;

    }

    return prev; //новий початок списку після реверсу

}

int main() {

    Node\* head = new Node(1);

    head->next = new Node(2);

    head->next->next = new Node(3);

    head->next->next->next = new Node(4);

    head->next->next->next->next = new Node(5);

    cout << "Input list: ";

    printList(head);

    head = reverse(head);

    cout << "Output list: ";

    printList(head);

    return 0;

}

Ввід програми Class Practice Task 1

Завдання №7: Class Practice Task 2

https://github.com/artificial-intelligence-department/ai\_programming\_playground/pull/1041/files#diff-a0c06ede76f78278e25eec0370db3b9e29fdf2c5d911967350dc9f8e81f4fb58

#include <iostream>

using namespace std;

struct Node {

    int data; // всі елементи списку є int

    Node\* next; // вказівник на наступний елемент

    Node(int value) : data(value), next(nullptr) {} // конструктор data - значення елемента

};

void printList(Node\* head) {

    while (head) {

        cout << head->data << " ";

        head = head->next;

    }

    cout << endl;

}

bool compare(Node\* h1, Node\* h2) {

    while (h1 != nullptr && h2 != nullptr) {

        if (h1->data != h2->data) {

            return false;

        }

        h1 = h1->next;

        h2 = h2->next;

    }

    if (h1 != nullptr || h2 != nullptr) {//один список більший за інший

        return false;

    }

    return true;

}

int main() {

    Node\* head1 = nullptr;

    Node\* head2 = nullptr;

    int n, m, value;

    cout << "Enter the number of elements for list 1: ";

    cin >> n;

    cout << "Enter the elements for list 1: ";

    for (int i = 0; i < n; i++) {

        cin >> value;

        if (head1 == nullptr) { //перевірка, чи список порожній

            head1 = new Node(value);

        } else {

            Node\* temp = head1;

            while (temp->next != nullptr) {

                temp = temp->next;

            }

            temp->next = new Node(value);

        }

    }

    cout << "Enter the number of elements for list 2: ";

    cin >> m;

    cout << "Enter the elements for list 2: ";

    for (int i = 0; i < m; i++) {

        cin >> value;

        if (head2 == nullptr) {

            head2 = new Node(value);

        } else {

            Node\* temp = head2;

            while (temp->next != nullptr) {

                temp = temp->next;

            }

            temp->next = new Node(value);

        }

    }

    cout << "List 1: ";

    printList(head1);

    cout << "List 2: ";

    printList(head2);

    if (compare(head1, head2)) {

        cout << "Lists are equal." << endl;

    } else {

        cout << "Lists are not equal." << endl;

    }

    return 0;

}

Ввід програми Class Practice Task 2

Завдання №8: Class Practice Task 3

https://github.com/artificial-intelligence-department/ai\_programming\_playground/pull/1041/files#diff-905305f9f7ddb55da945f55e8415de572a18068a9b11b7691224f1a8cda87709

#include <iostream>

using namespace std;

struct Node {

    int data; // всі елементи списку є int

    Node\* next; // вказівник на наступний елемент

    Node(int value) : data(value), next(nullptr) {} // конструктор data - значення елемента

};

void printList(Node\* head) {

    while (head) {

        cout << head->data << " ";

        head = head->next;

    }

    cout << endl;

}

Node\* add(Node\* n1, Node\* n2) {

    Node\* result = nullptr;

    Node\* current = nullptr;

    int carry = 0;

    while (n1 != nullptr || n2 != nullptr || carry != 0) {

        int sum = carry;

        if (n1 != nullptr) {

            sum += n1->data;

            n1 = n1->next;

        }

        if (n2 != nullptr) {

            sum += n2->data;

            n2 = n2->next;

        }

        int digit = sum % 10;

        carry = sum / 10;

        if (result == nullptr) {

            result = new Node(digit);

            current = result;

        }

        else {

            current->next = new Node(digit);

            current = current->next;

        }

    }

    return result;

}

int main() {

    Node\* num1 = new Node(3);

    num1->next = new Node(7);

    num1->next->next = new Node(9);

    Node\* num2 = new Node(4);

    num2->next = new Node(2);

    num2->next->next = new Node(8);

    cout << "Number 1: ";

    printList(num1);

    cout << "Number 2: ";

    printList(num2);

    Node\* result = add(num1, num2);

    cout << "Sum: ";

    printList(result);

    return 0;

}

Ввід програми Class Practice Task 3

Завдання №9: Class Practice Task 4

https://github.com/artificial-intelligence-department/ai\_programming\_playground/pull/1041/files#diff-f72f699497c85a91e2913d1714507a5d24a145d24705a33ef7af4fbb022dcf38

#include <iostream>

using namespace std;

struct TreeNode {

    int val;

    TreeNode\* left;

    TreeNode\* right;

    TreeNode(int x) : val(x), left(nullptr), right(nullptr) {}

};

TreeNode\* create\_mirror\_flip(TreeNode\* root) {

    if (root == nullptr) {

        return nullptr;

    }

    TreeNode\* newRoot = new TreeNode(root->val);

    newRoot->right = create\_mirror\_flip(root->left);

    newRoot->left = create\_mirror\_flip(root->right);

    return newRoot;

}

void inorderTraversal(TreeNode\* root) {

    if (root != nullptr) {

        inorderTraversal(root->left);

        cout << root->val << " ";

        inorderTraversal(root->right);

    }

}

int main() {

    TreeNode\* root = new TreeNode(1);

    root->left = new TreeNode(2);

    root->right = new TreeNode(3);

    root->left->left = new TreeNode(4);

    root->left->right = new TreeNode(5);

    cout << "Original tree: ";

    inorderTraversal(root);

    cout << endl;

    TreeNode\* mirroredTree = create\_mirror\_flip(root);

    cout << "Mirrored tree: ";

    inorderTraversal(mirroredTree);

    cout << endl;

    return 0;

}

Ввід програми Class Practice Task 4

Завдання №10: Class Practice Task 5

https://github.com/artificial-intelligence-department/ai\_programming\_playground/pull/1041/files#diff-94009839f88e240e39765ecd82c6f42c62bdaba3b4ce358d7905dde59122eee8

#include <iostream>

#include <stack>

using namespace std;

struct TreeNode {

    int val;

    TreeNode \*left;

    TreeNode \*right;

    TreeNode(int x) : val(x), left(nullptr), right(nullptr) {}

};

void tree\_sum(TreeNode \*root) {

    if (!root) {

        return;

    }

    std::stack<TreeNode\*> nodeStack;

    TreeNode \*current = root;

    TreeNode \*prev = nullptr;

    while (current || !nodeStack.empty()) {

        while (current) {

            nodeStack.push(current);

            current = current->left;

        }

        current = nodeStack.top();

        if (current->right && current->right != prev) {  //якщо поточний вузол має правого підвузла і він не оброблений

            current = current->right;

        } else {

            int sum = current->val; //обчислюємо суму підвузлів і зберігаємо в поточному вузлі

            if (current->left) {

                sum += current->left->val;

            }

            if (current->right) {

                sum += current->right->val;

            }

            current->val = sum;

            nodeStack.pop(); //батьківський вузол

            prev = current;

            current = nullptr;

        }

    }

}

void printTree(TreeNode\* root) {//вивести значення дерева для перевірки

    if (root) {

        printTree(root->left);

        cout << root->val << " ";

        printTree(root->right);

    }

}

int main() {

    TreeNode \*root = new TreeNode(1);

    root->left = new TreeNode(2);

    root->right = new TreeNode(3);

    root->left->left = new TreeNode(4);

    root->left->right = new TreeNode(5);

    root->right->left = new TreeNode(6);

    root->right->right = new TreeNode(7);

    cout << "Original tree: ";

    printTree(root);

    cout << endl;

    tree\_sum(root);

    cout << "Modified tree: ";

    printTree(root);

    cout << endl;

    return 0;

}

Ввід програми Class Practice Task 5

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

Завдання №3: VNS Lab 10

Original tree: 4 2 5 1 6 3 7

Modified tree: 4 11 5 28 6 16 7

Вивід програми VNS Lab 10

Час затрачений на виконання завдання: 6 годин.

Завдання №4: Algotester Lab 5v2

5 5

SSOSS

OOOOO

SOOXX

OOOOS

OOSOO

OOOOO

OOOSS

OOOXX

SOOOO

SSSOS

Вивід програми Algotester Lab 5v2

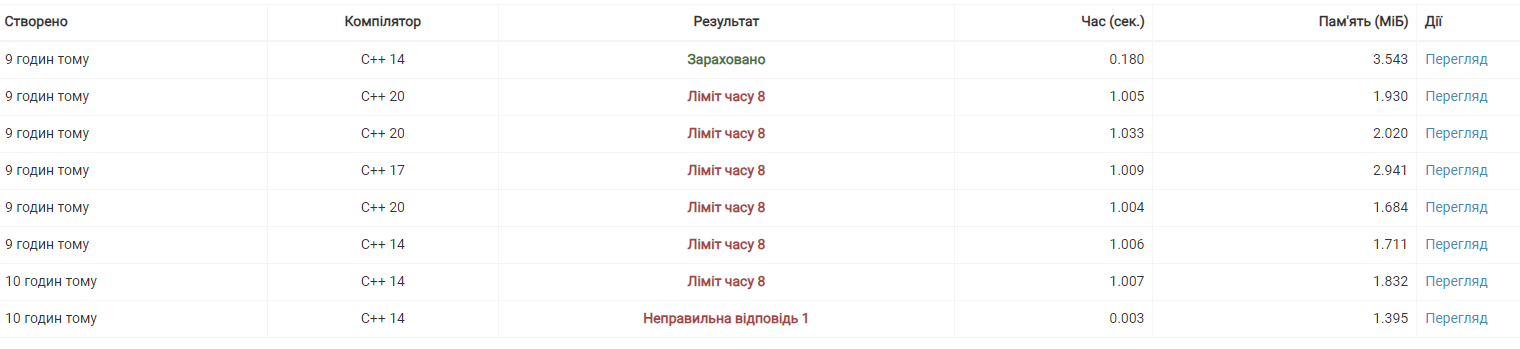


Figure 2: Algotester 5v2

Час затрачений на виконання завдання: 4 години.

Завдання №5: Algotester Lab 78v2

12

size

0

capacity

1

insert 0 2

100 100

size

2

capacity

4

insert 0 2

102 102

size

4

capacity

8

insert 0 2

103 103

size

6

capacity

8

print

103 103 102 102 100 100

Вивід програми Algotester Lab 78v2

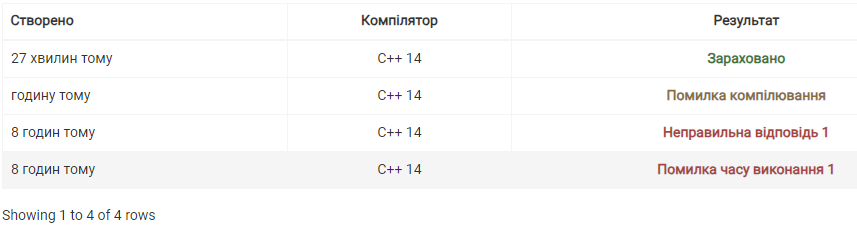


Figure 3 Algotester 78v2

Час затрачений на виконання завдання: 7 годин.

Завдання №6: Class Practice Task 1

Input list: 1 2 3 4 5

Output list: 5 4 3 2 1

Вивід програми Class Practice Task 1

Час затрачений на виконання завдання: 2 години.

Завдання №7: Class Practice Task 2

Enter the number of elements for list 1: 5

Enter the elements for list 1: 1 2 3 4 5

Enter the number of elements for list 2: 4

Enter the elements for list 2: 1 2 3 4

List 1: 1 2 3 4 5

List 2: 1 2 3 4

Lists are not equal.

Вивід програми Class Practice Task 2

Час затрачений на виконання завдання: 2 години.

Завдання №8: Class Practice Task 3

Number 1: 3 7 9

Number 2: 4 2 8

Sum: 7 9 7 1

Вивід програми Class Practice Task 3

Час затрачений на виконання завдання: 3 години.

Завдання №9: Class Practice Task 4

Original tree: 4 2 5 1 3

Mirrored tree: 3 1 5 2 4

Вивід програми Class Practice Task 4

Час затрачений на виконання завдання: 3 години.

Завдання №10: Class Practice Task 5

Original tree: 4 2 5 1 6 3 7

Modified tree: 4 11 5 28 6 16 7

Вивід програми Class Practice Task 5

Час затрачений на виконання завдання: 3 години.

# **Висновки:**

В ході виконання лабораторної роботи №6 я ознайомилася з навчальною теорією про динамічні структури та їх різновид, а також дослідила їх реалізацію в коді; це дало мені можливість покращити знання програмування на мові C++. Виконала одне лабораторних завдання із застосуванням знань про двонаправлені списки. Написала два коди у середовищі Algotester, а також виконала практичне завдання. Оформила звіт за зразком. Ця лабораторна робота розвинула мої навички розуміння роботи комп’ютера з кодом, принцип побудови блок-схем та написання програм на мові C++.