МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний аерокосмічний університет ім. М. Є. Жуковського
«Харківський авіаційний інститут»

Факультет систем управління літальних апаратів
Кафедра систем управління літальних апаратів

Лабораторна робота № 9

з дисципліни «Алгоритмізація та програмування»
на тему «Робота з рядками на С ++»

ХАІ.301.173.310.1 ЛР

Виконав студент гр. _____310_____

_____Микола АНДРЮШКІН_____
(підпис, дата) (П.І.Б.)

Перевірив
_____ к.т.н., доц. Олена ГАВРИЛЕНКО
(підпис, дата) (П.І.Б.)

2025

МЕТА РОБОТИ

Вивчити теоретичний матеріал з основ роботи з низькорівневими рядками на C++ і документацію до класу string, а також алгоритми пошуку в рядку, а також реалізувати обробку рядків на C++ в середовищі Visual Studio.

СХЕМИ РОБОТИ КОДУ

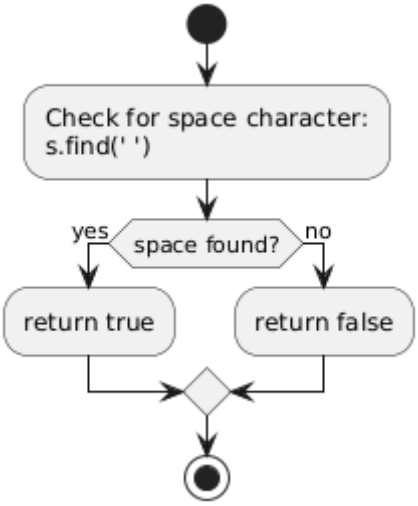

Рисунок 1 — chkLine()

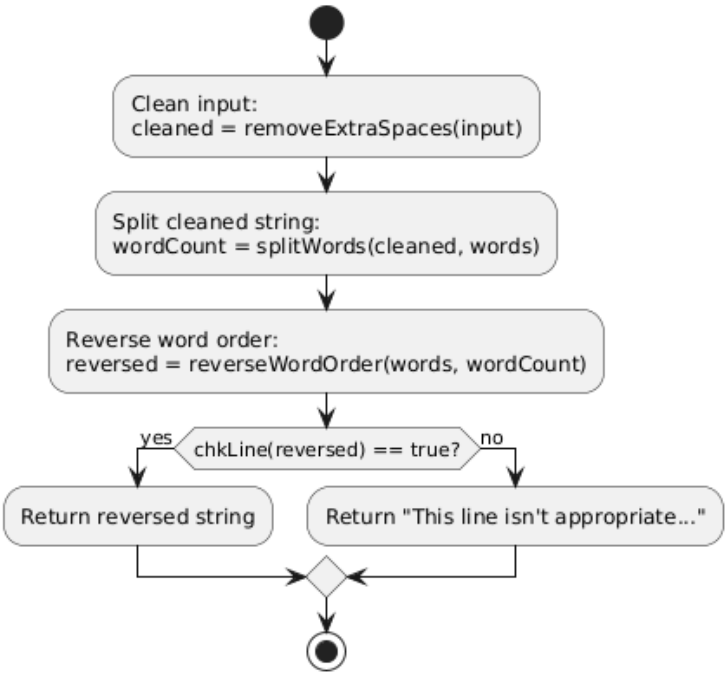

Рисунок 2 — reverseWords()

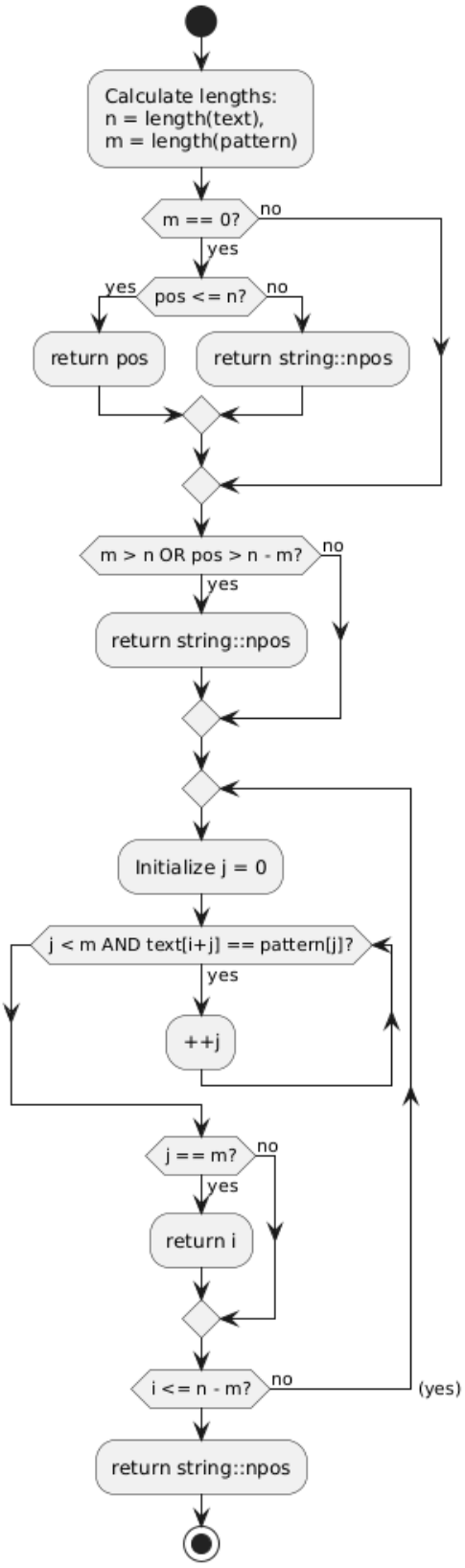

Рисунок 3 — findSubstringPosition()

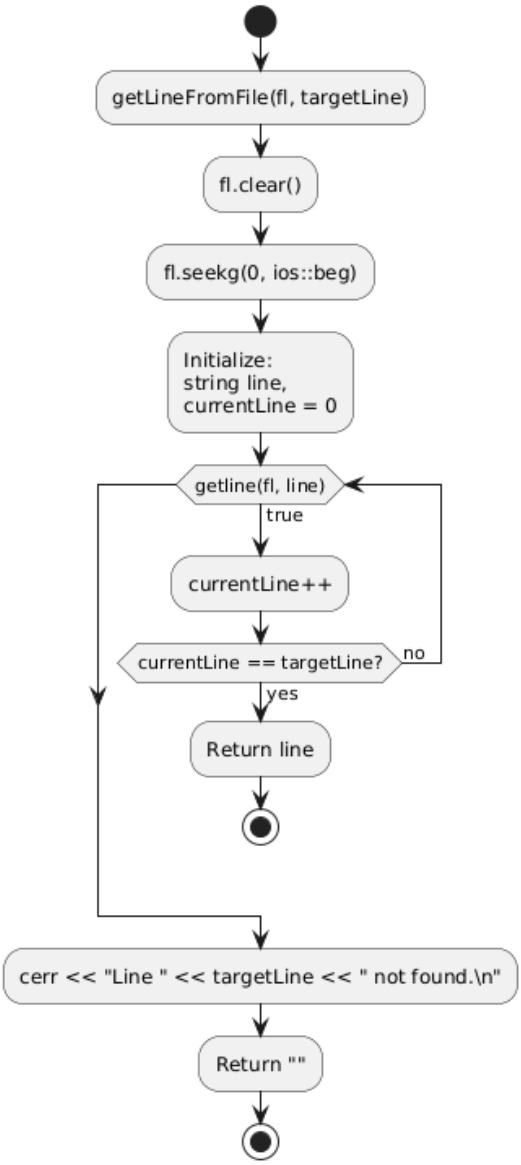

Рисунок 4 — getLineFromFile()

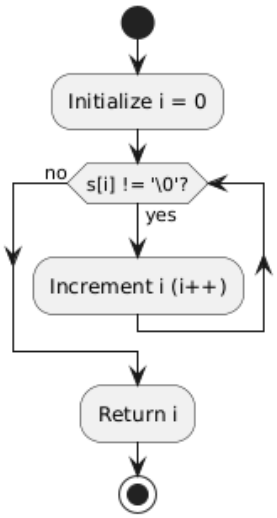

Рисунок 5 — lenght()

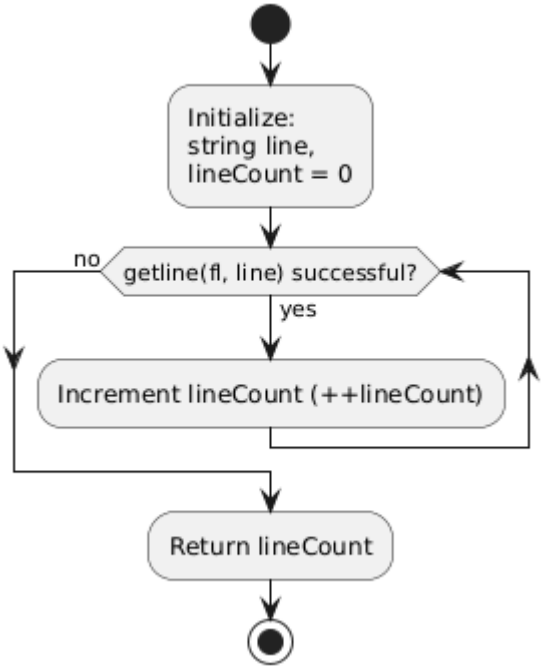

Рисунок 6 — lineNum()

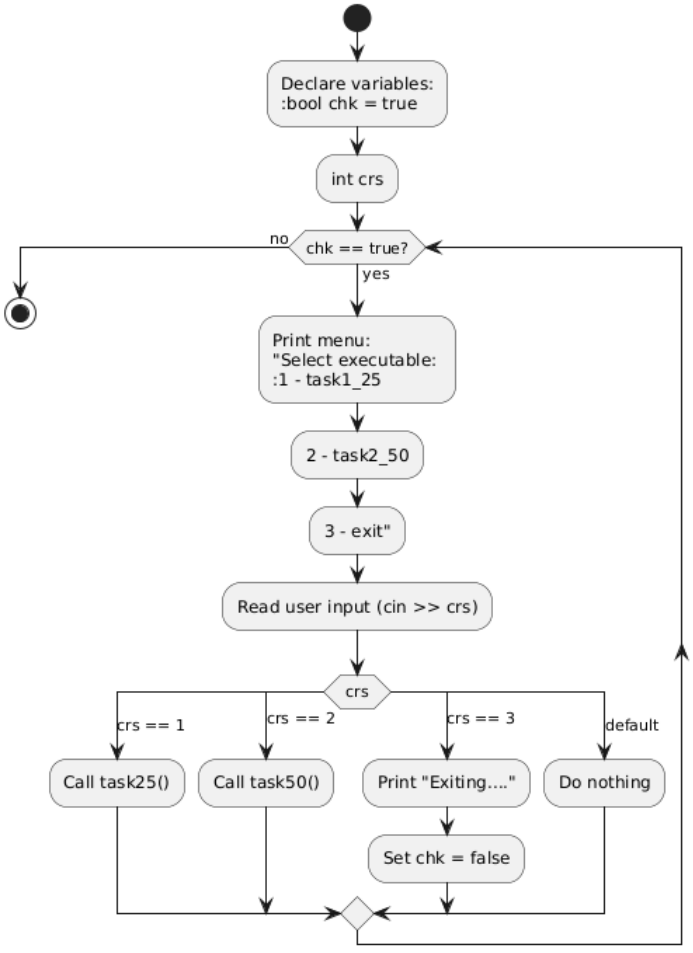

Рисунок 7 —main()

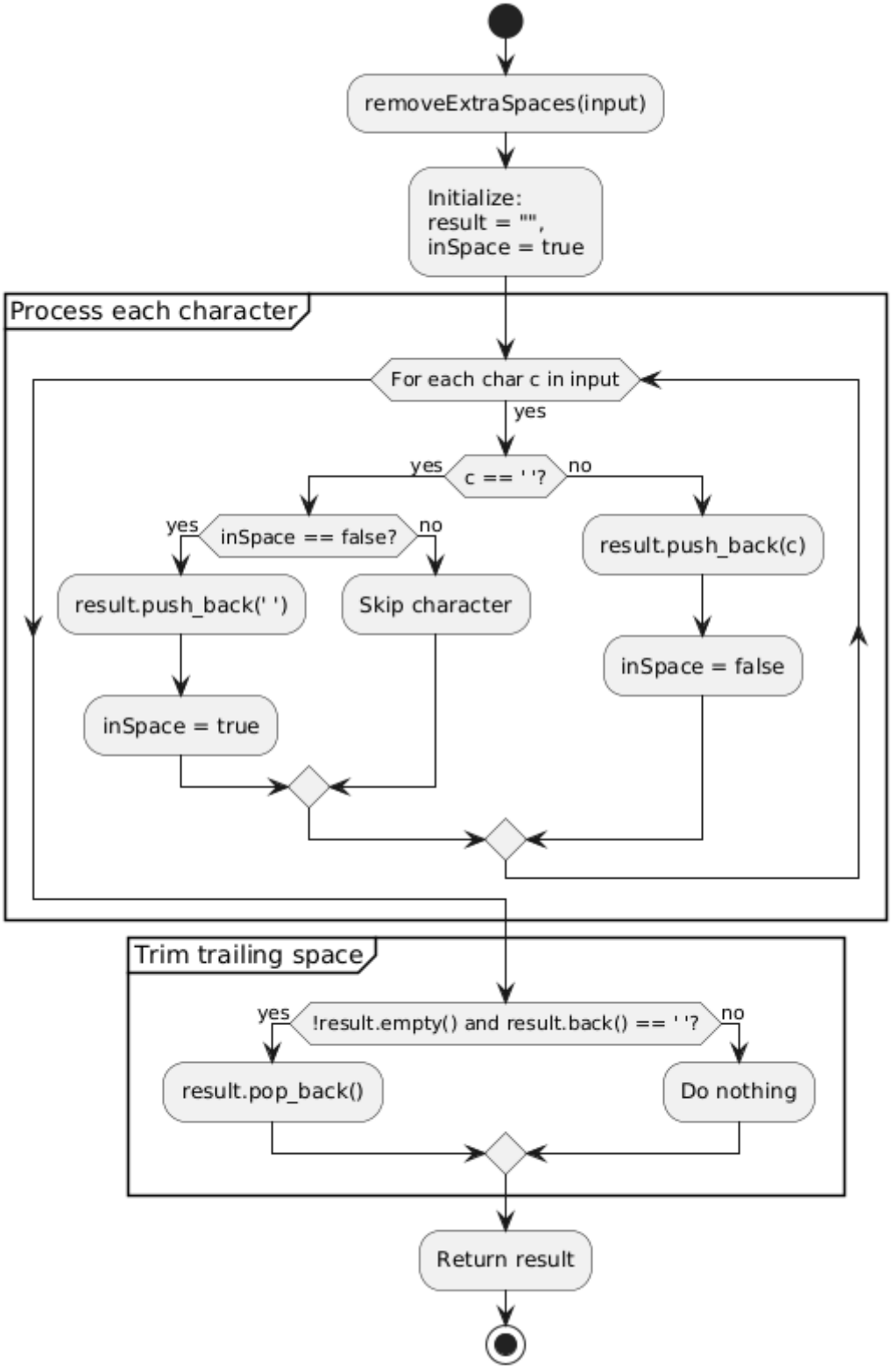

Рисунок 8 —removeExtraSpaces()

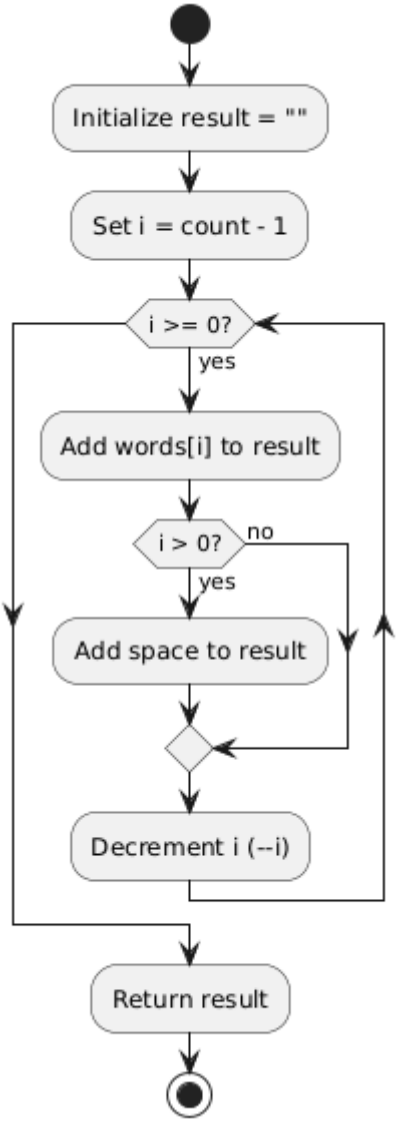

Рисунок 9 —reverseWordOrder()

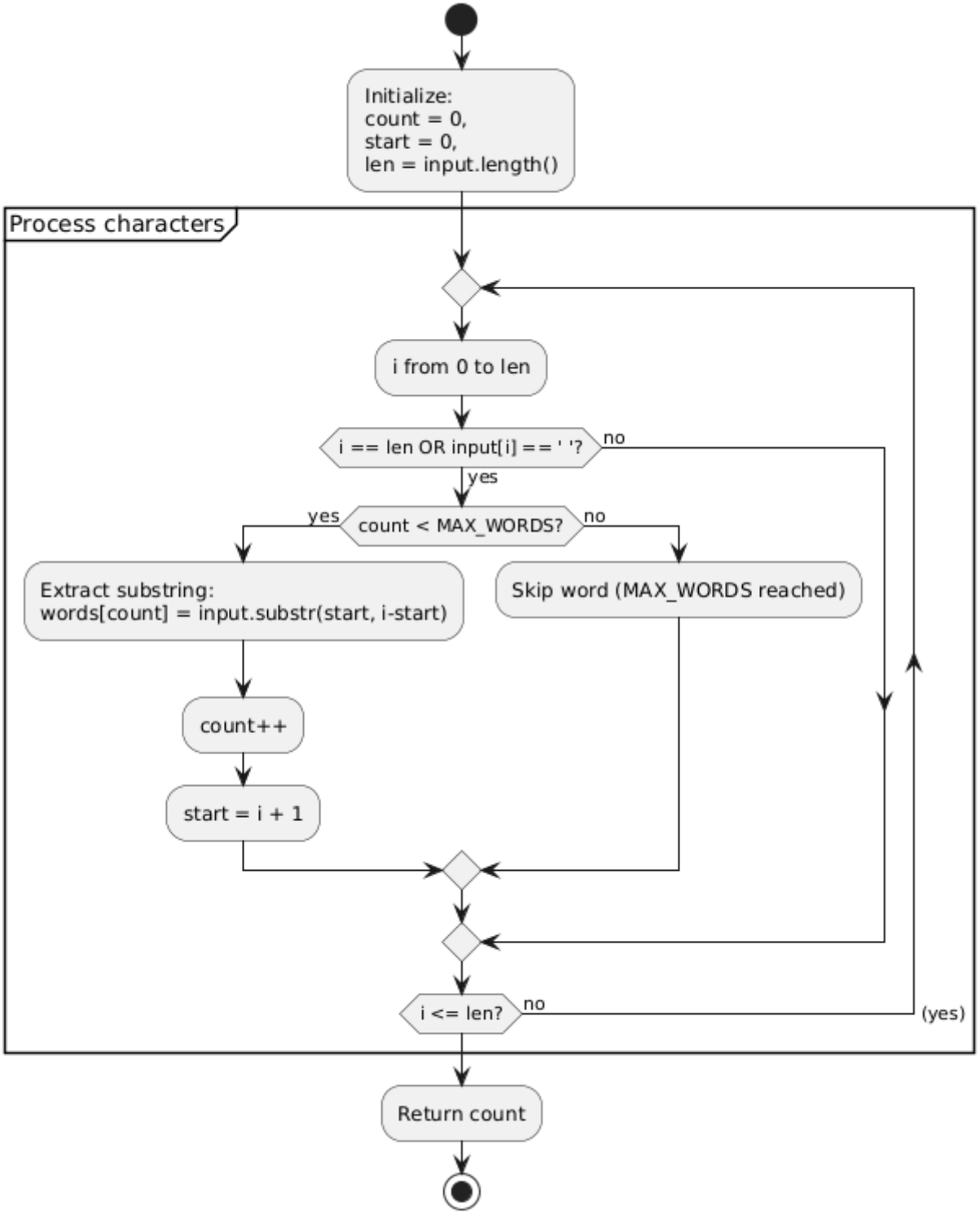

Рисунок 10 —splitWords()

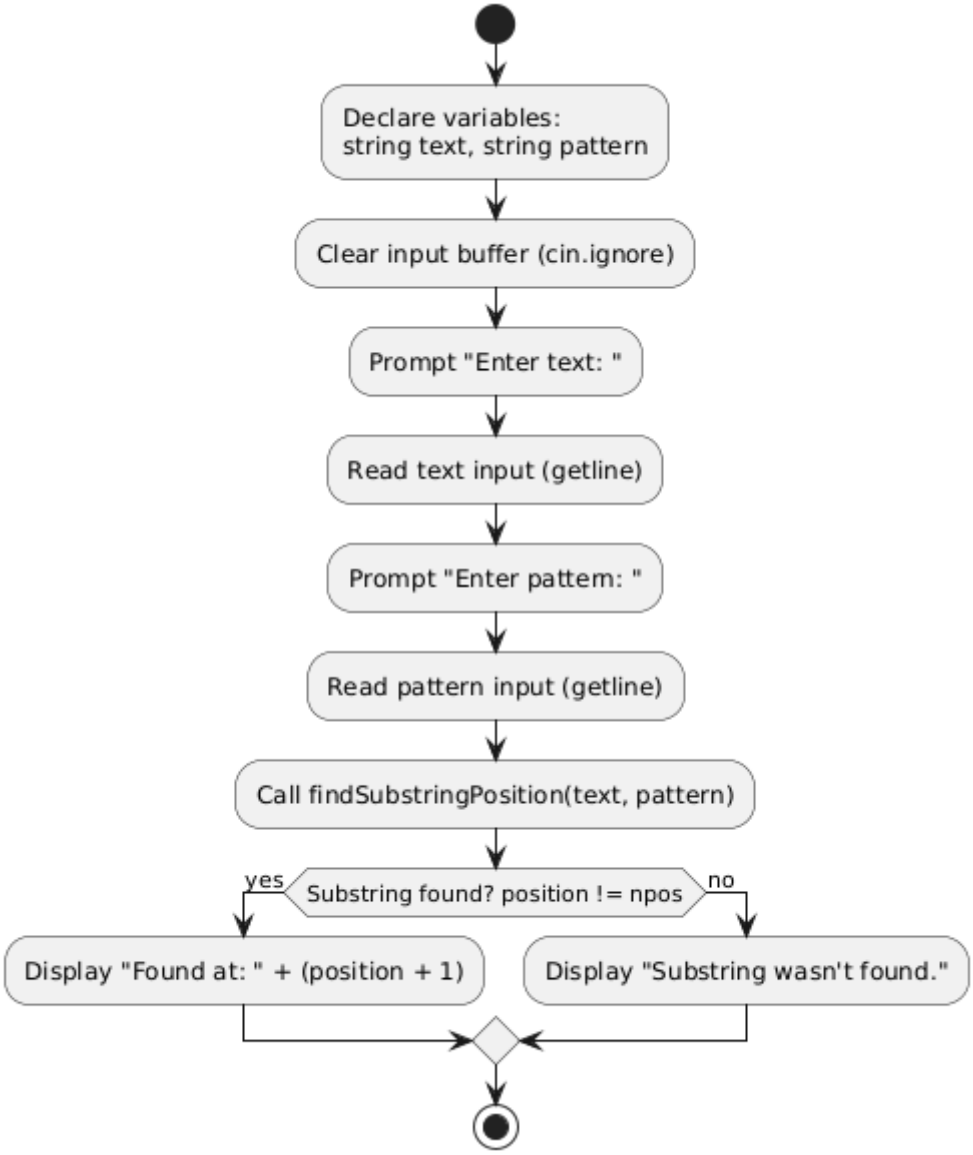

Рисунок 11 —task25()

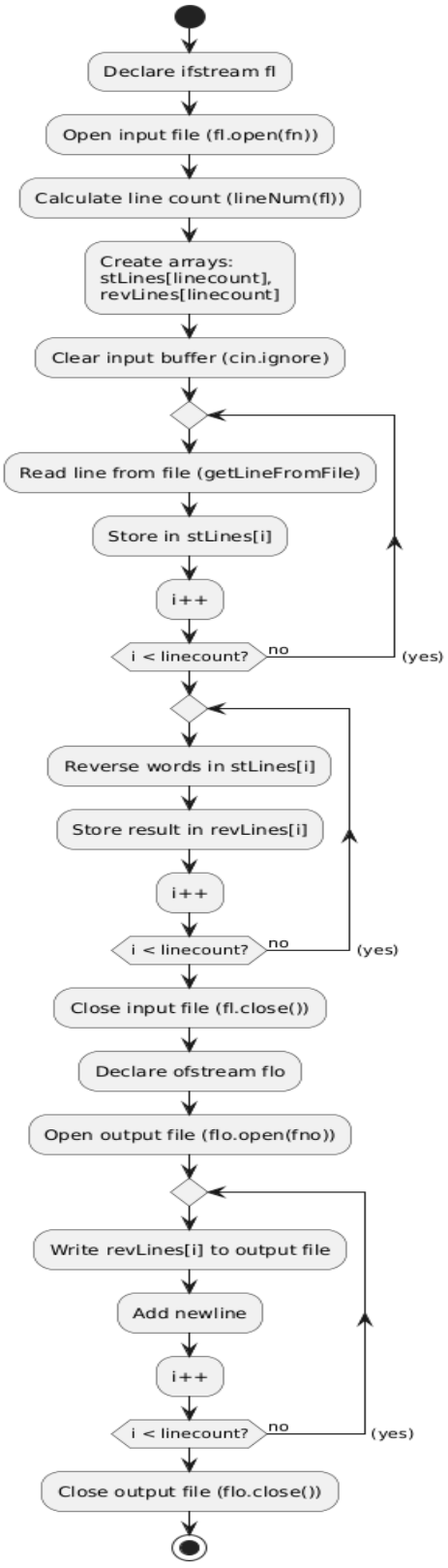

Рисунок 12 —task50()

ВИСНОВКИ

Закріплена на практиці робота з рядками та файлами. Відпрацьована в коді програми обробка рядків. Виникли труднощі з обробкою рядків з файлу.

ДОДАТОК А

Лістинг коду програми

```
   кmain.cpp
#include <iostream>
#include <string>
#include "task1_25.h"
#include "task2_50.h"
using namespace std;

// Task 1_25 main function
void task25() {
    // Dectaration of variables
    string text;
    string pattern;

    // String input
    cin.ignore(256, '\n');
    cout << "Enter text: ";
    getline(cin, text);
    cout << "Enter pattern: ";
    getline(cin, pattern);

    // Displaying results
      size_t position = findSubstringPosition(text, pattern); // починаємо з
позиції 5
    if (position != string::npos) {
        cout << "Found at: " << position + 1 << endl;
    } else {
        cout << "Substring wasn't found." << endl;
    }
}

//Task 2_50 main function
void task50(){
    // Declaration of variables
    ifstream fl;
    fl.open(fn); //file open
    size_t linecount;
    linecount = lineNum(fl); // linecout is how many lines in file
    string stLines[linecount];
    string revLines[linecount];

    cin.ignore(256, '\n');

    // Getting lines from file
    for(int i = 0; i < linecount; i++){
```

```
        stLines[i] = getLineFromFile(fl, i + 1);
    }

    // Reversing of lines
    for(int i = 0; i < linecount; i++){
        revLines[i] = reverseWords(stLines[i]);
    }

    fl.close(); // File closed


    ofstream flo; //file ouput open
    flo.open(fno);

    // writing results to file
    for(int i = 0; i < linecount; i++){
        flo << revLines[i] << endl;
    }
    flo.close(); // file output close
}

int main(){

    // declaration of variables
    bool chk;
    chk = true;
    int crs;

    // menu
    while(chk){
         cout << "Select executable: 1 - task1_25; 2 - task2_50; 3 - exit from
program: ";
        cin >> crs;
        switch (crs)
        {
        case 1:
            task25();
            break;
        case 2:
            task50();
            break;
        case 3:
            cout << "Exiting...." << endl;
            chk = false;
            break;
        default:
            break;
        }
    }
}
```

```
task1_25.h
#include <iostream>
#include <string>
using namespace std;

//function that returns lenght of string
size_t lenght(string s){
    size_t i;
    while (s[i]) i++;
    return i;
}

//function that returns position of pattern in text(if pattern not found - returns npos)
size_t findSubstringPosition(const string& text, const string& pattern, size_t pos = 0) {
    // declaration of variables
    size_t n = lenght(text);
    size_t m = lenght(pattern);

    //checks
    if (m == 0) {
        return pos <= n ? pos : string::npos;
    }
    if (m > n || pos > n - m) {
        return string::npos;
    }

    // find for cycle
    for (size_t i = pos; i <= n - m; ++i) {
        size_t j = 0;
        while (j < m && text[i + j] == pattern[j]) {
            ++j;
        }
        if (j == m) {
            return i; // знайдено входження
        }
    }

    return string::npos; // не знайдено
}

task2_50.h
#include <iostream>
#include <string>
#include <fstream>
using namespace std;

// declaration of constants
```

```
const char * fn = "bukavy.txt";
const char * fno = "bukavy_out.txt";
const int MAX_WORDS = 100;

// function that reads file and returns number of lines in text file
size_t lineNum(ifstream& fl){
    //declaration of variables
    string line;
    size_t lineCount = 0;

    // linecount while loop
    while (getline(fl, line)) {
        ++lineCount;
    }
    return lineCount;
}

// function that checks if string has spaces
bool chkLine(string s){
    return s.find(' ') != string::npos;
}


string removeExtraSpaces(const string& input) {
    //declaration of variables
    string result;
    bool inSpace = true; // ingnores spaces in beginning of string

    // space deletion cycle
    for (char c : input) { // char c == input[i]; i++
        if (c == ' ') {
            if (!inSpace) {
                result.push_back(' ');
                inSpace = true;
            }
        } else {
            result.push_back(c);
            inSpace = false;
        }
    }

    // delete space in the end if it exist and string isn't empty
    if (!result.empty() && result.back() == ' ') {
        result.pop_back();
    }

    return result;
}

// function that splits string in array of words
```

```
int splitWords(const string& input, string words[]) {

    //declaration of variables
    int count = 0;
    int start = 0;
    int len = input.length();

    //cycle that splits
    for (int i = 0; i <= len; ++i) {
        if (i == len || input[i] == ' ') {
            if (count < MAX_WORDS) {
                words[count++] = input.substr(start, i - start);
                start = i + 1;
            }
        }
    }

    return count;
}

// function that receives array of strings and sticks them to one string on
reverse order
string reverseWordOrder(string words[], int count) {
    string result;

    for (int i = count - 1; i >= 0; --i) {
        result += words[i];
        if (i > 0) {
            result += ' ';
        }
    }

    return result;
}

// function that combines all functions from this c header file
string reverseWords(string& input) {
    string cleaned = removeExtraSpaces(input);
    string words[MAX_WORDS];
    int wordCount = splitWords(cleaned, words);
    if (chkLine(reverseWordOrder(words, wordCount))){
        return reverseWordOrder(words, wordCount);
    }
    return "This line isn't appropriate, please provide strings with spaces.";
}

// get specific line from text file
string getLineFromFile(ifstream& fl, int targetLine) {
    fl.clear();
    fl.seekg(0, ios::beg);
```

```
    string line;
    int currentLine = 0;

    while (getline(fl, line)) {
        ++currentLine;
        if (currentLine == targetLine) {
            return line;
        }
    }

    // if at the end and line don't found
    cerr << "Line " << targetLine << " not found.\n";
    return "";
}
```

ДОДАТОК Б

Скрін-шоти вікна виконання програми

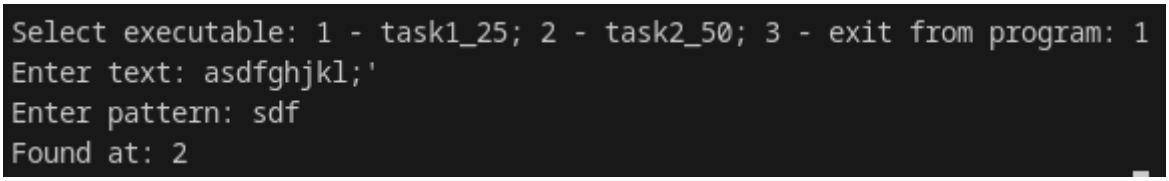

Рисунок Б.1 – Екран виконання програми для вирішення завдання Task1_25

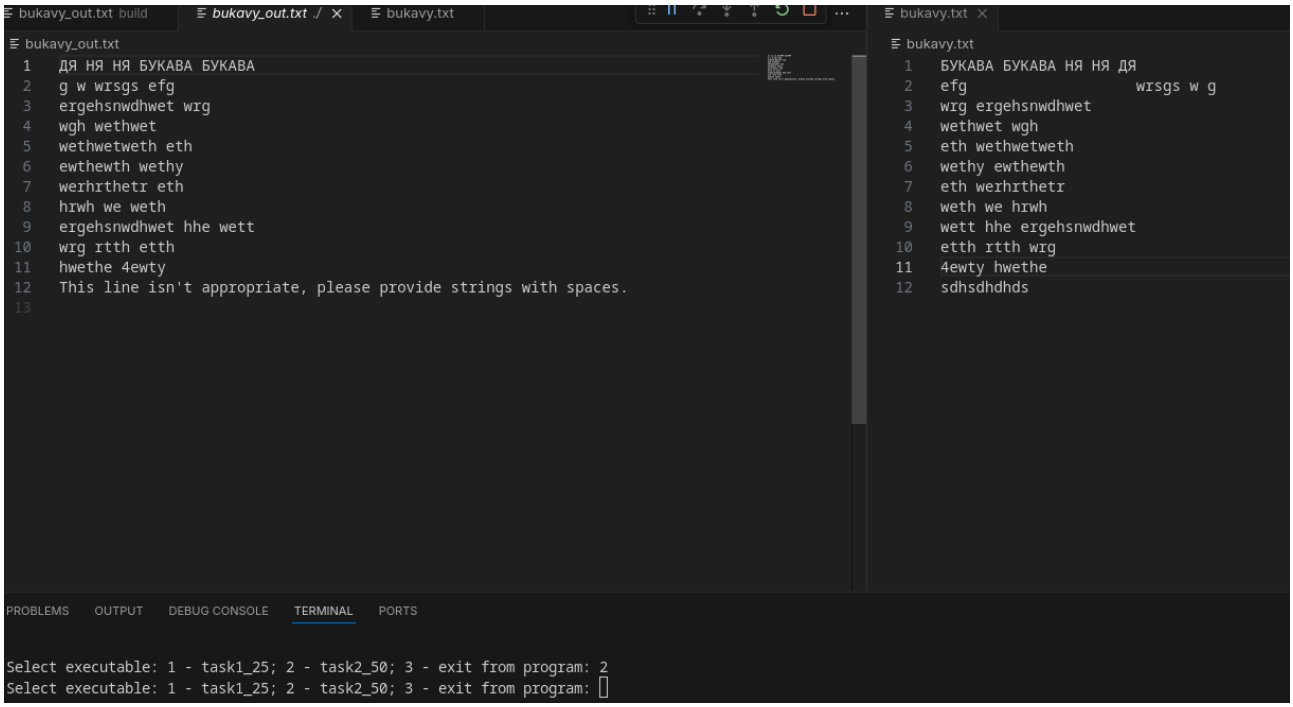

Рисунок Б.2 – Екран виконання програми для вирішення завдання Task2_50