

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний університет водного господарства та природокористування
Навчально-науковий інститут автоматики, кібернетики та обчислювальної
техніки

Кафедра обчислювальної техніки

Звіт

З навчальної практики з “Програмування”

Виконали:

студенти II курсу групи КІ-21

спеціальності “Комп’ютерна інженерія”

Алтунін Євген Анатолійович

Перевірив:

проф. Заяць Василь Михайлович

Рівне - 2019

1 – Варіант

Список завдань

Завдання 1. Побудувати програму, яка реалізує наступну задачу:

Розглянемо масив цілих або дійсних чисел a_1, \dots, a_n . Нехай треба переставити елементи цього масиву так, щоб після перестановки вони були впорядковані по не спаданню $a_1 \leq a_2 \leq \dots \leq a_n$. Ця задача має назву сортування. Для розв'язку цієї задачі можна використати наступний алгоритм: знайти елемент масиву, що має найменше значення, переставити його з першим елементом, далі те ж саме виконати, починаючи з другого елементу і тд.

Завдання 2. Побудувати програму, яка реалізує наступну задачу:

Задано n населених пунктів, що пронумеровані від 1 до n . Деякі пари пунктів з'єднані дорогами. Скласти програму, яка визначає, чи можливо з пункту 1 проїхати в пункт n . Використати рекурсію.

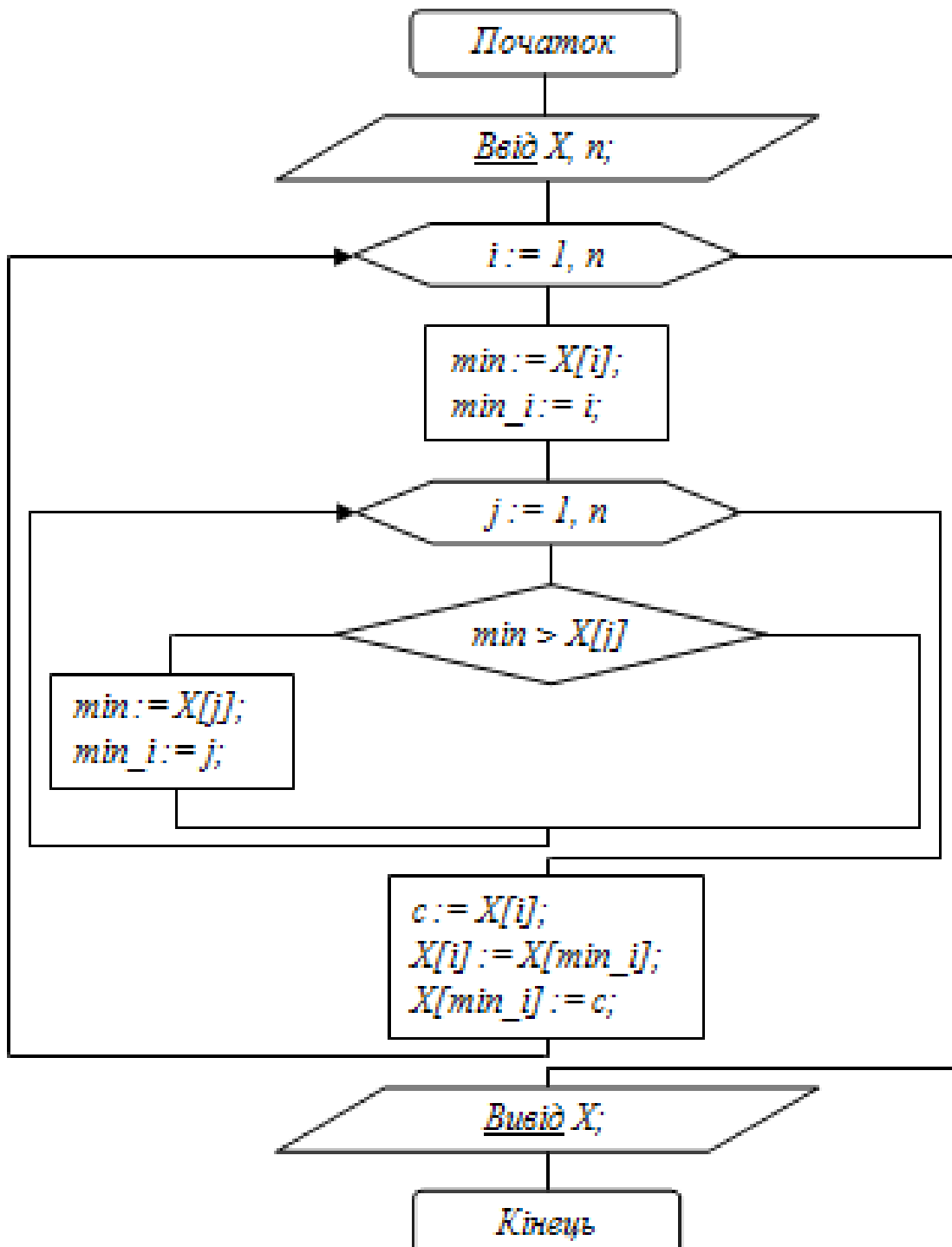
Завдання 3. Побудувати програму, яка реалізує наступну задачу:

Скласти тестову програму для перевірки знань по заданій темі з програмування. Тестів не менше 10, в кожному по 4 відповіді, 1 правильна.

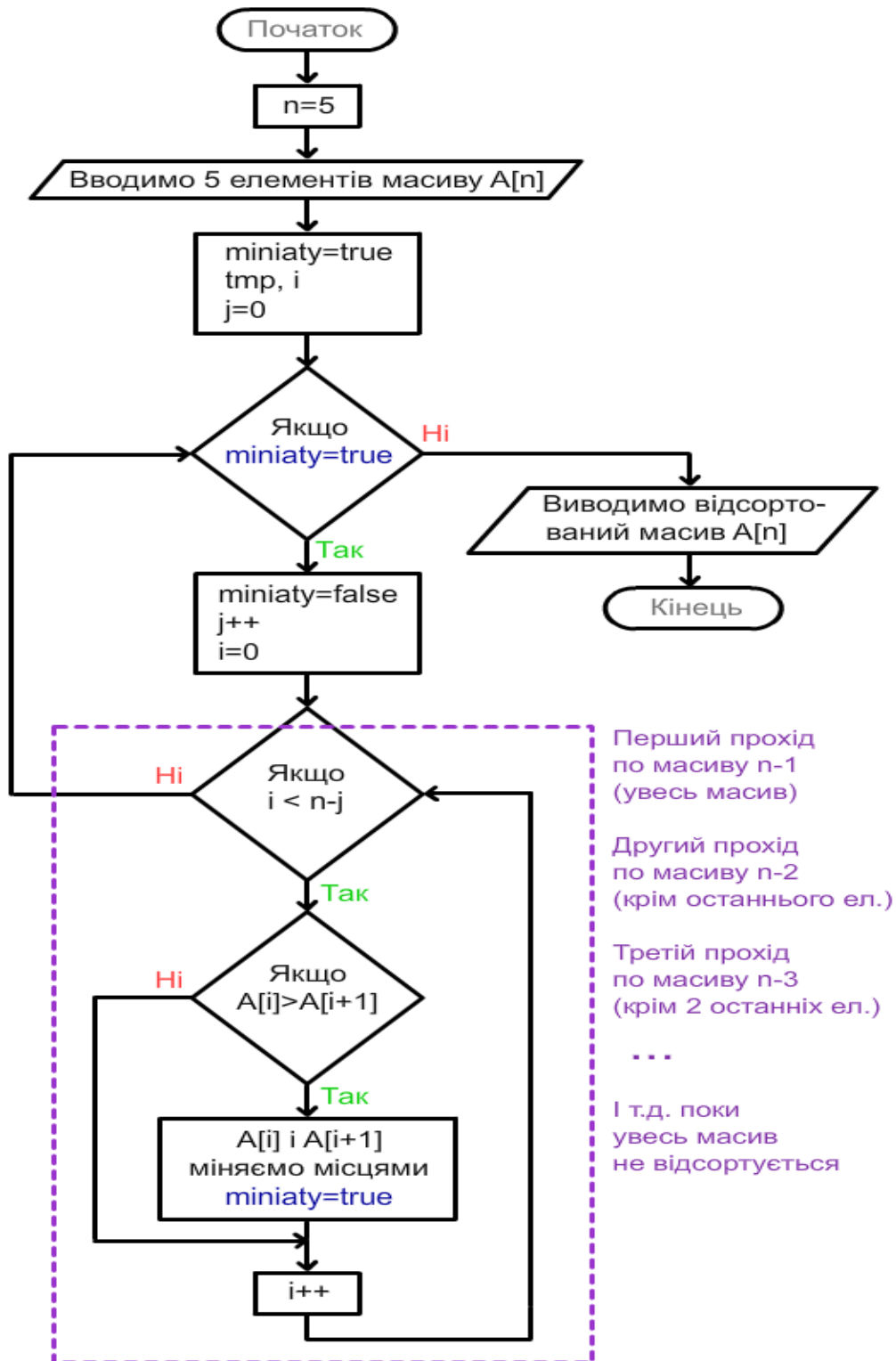
Завдання 1.

Методи сортування

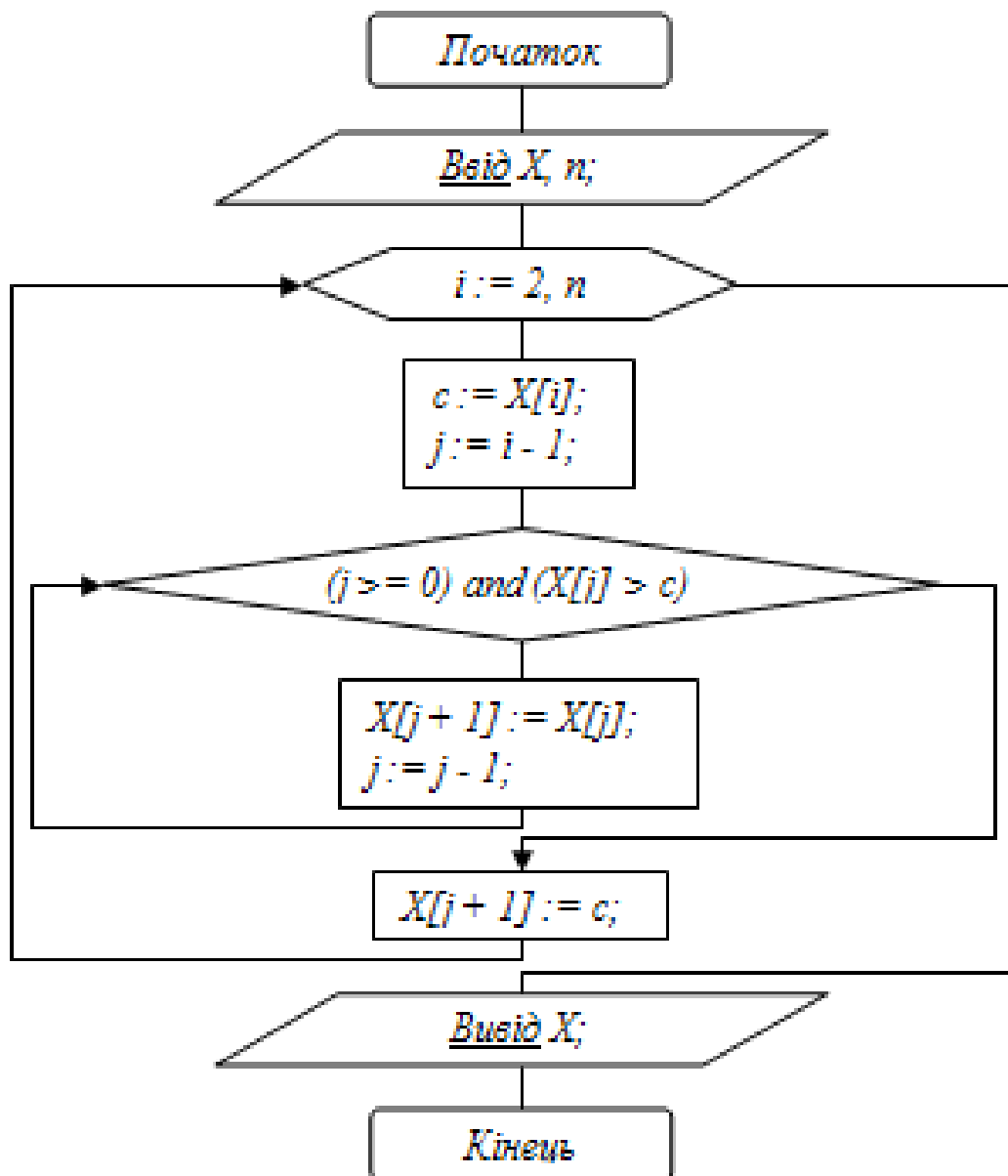
- **Сортування вибором** — простий алгоритм сортування лінійного масиву, на основі вставок. Має ефективність n^2 , що робить його неефективним при сортуванні великих масивів, і в цілому, менш ефективним за подібний алгоритм сортування включенням. Сортування вибором вирізняється більшою простотою, ніж сортування включенням, і в деяких випадках, вищою продуктивністю.



- **Сортування обміном або сортування бульбашкою** - є простим алгоритмом сортування. Алгоритм отримав свою назву від того, що процес сортування за ним нагадує поведінку бульбашок повітря у резервуарі з водою. Оскільки для роботи з елементами масиву він використовує лише порівняння, це сортування на основі порівнянь.



- **Сортування вставками** - простий алгоритм сортування на основі порівнянь. На великих масивах є значно менш ефективним за такі алгоритми, як швидке сортування, пірамідальне сортування та сортування злиттям.



Реалізація програми

```
#include <iostream>
#include <cstdlib>
#include <ctime>
using namespace std;
void SelectSort(int arr[], int SIZE);
void BubleSort(int arr[], int SIZE);
void InsertSort(int arr[], int SIZE);
int main()
{
    setlocale(LC_ALL, "ukrainian");
    srand(time(nullptr));
    const int SIZE = 1000;
    int arr[SIZE];
    cout << "Початковий масив:\n";
    for (int i = 0; i < SIZE; i++)
    {
        arr[i] = rand() % 101 - 50;
        cout << arr[i] << "\t";
    }
    cout << "\n";
    InsertSort(arr, SIZE);
    BubleSort(arr, SIZE);
    SelectSort(arr, SIZE);
    system("pause");
    return 0;
}
```

```

void InsertSort(int arr[],int SIZE)
{
    int *arrayToSort = new int[SIZE];
    arrayToSort = arr;
    clock_t start = clock();
    int j;
    for (int i = 0; i < SIZE; ++i)
    {
        int temp = arrayToSort[i];
        for (j = i - 1; j >= 0 && arrayToSort[j] > temp; --j)
            arrayToSort[j + 1] = arrayToSort[j];
        arrayToSort[j + 1] = temp;
    }
    clock_t end = clock();
    cout << "Результат сортування вставками:\n";
    for (int i = 0; i < SIZE; i++)
        cout << arrayToSort[i] << "\t";
    cout << "\n";
    cout << "Час, витрачений на сортування вставками " << end - start
<< " мс\n";
}

void SelectSort(int arr[], int SIZE)
{
    int *arrayToSort = new int[SIZE];
    arrayToSort = arr;
    clock_t start = clock();
    for (int i = 0; i < SIZE; ++i)
    {
        int minIndex = i, minValue = arrayToSort[i];

```

```

        for (int j = i + 1; j < SIZE; ++j)
            if (arrayToSort[j] < minValue)
            {
                minValue = arrayToSort[j];
                minIndex = j;
            }
        arrayToSort[minIndex] = arrayToSort[i];
        arrayToSort[i] = minValue;
    }
    clock_t end = clock();
    cout << "Результат сортування вибором:\n";
    for (int i = 0; i < SIZE; i++)
        cout << arrayToSort[i] << "\t";
    cout << "\n";
    cout << "Час, витрачений на сортування вибором " << end - start <<
    " мс\n";
}

void BubleSort(int arr[], int SIZE)
{
    int *arrayToSort = new int[SIZE];
    arrayToSort = arr;
    cout << "Результат сортування булькою:\n";
    clock_t start = clock();
    for (int i = 0; i < SIZE; ++i)
    {
        for (int j = SIZE - 1; j > i; --j)
            if (arrayToSort[j] < arrayToSort[j - 1])
            {
                int temp = arrayToSort[j];

```



```

        arrayToSort[j] = arrayToSort[j - 1];
        arrayToSort[j - 1] = temp;
    }
}

clock_t end = clock();
for (int i = 0; i < SIZE; i++)
    cout << arrayToSort[i] << "\t";
cout << "\n";
cout << "Час, витрачений на сортування булькою " << end - start <<
" мс\n";
}

```

Результат виконання програми:

```

Початковий масив:
-16    35    45   -46    29    17   -23    5   -36    32
Результат сортування вставками:
-46   -36   -23   -16    5    17    29    32    35    45
Час, витрачений на сортування вставками 0 мс
Результат сортування булькою:
-46   -36   -23   -16    5    17    29    32    35    45
Час, витрачений на сортування булькою 0 мс
Результат сортування вибором:
-46   -36   -23   -16    5    17    29    32    35    45
Час, витрачений на сортування вибором 0 мс
Для продовження натисніть будь-яку клавішу . . .

```

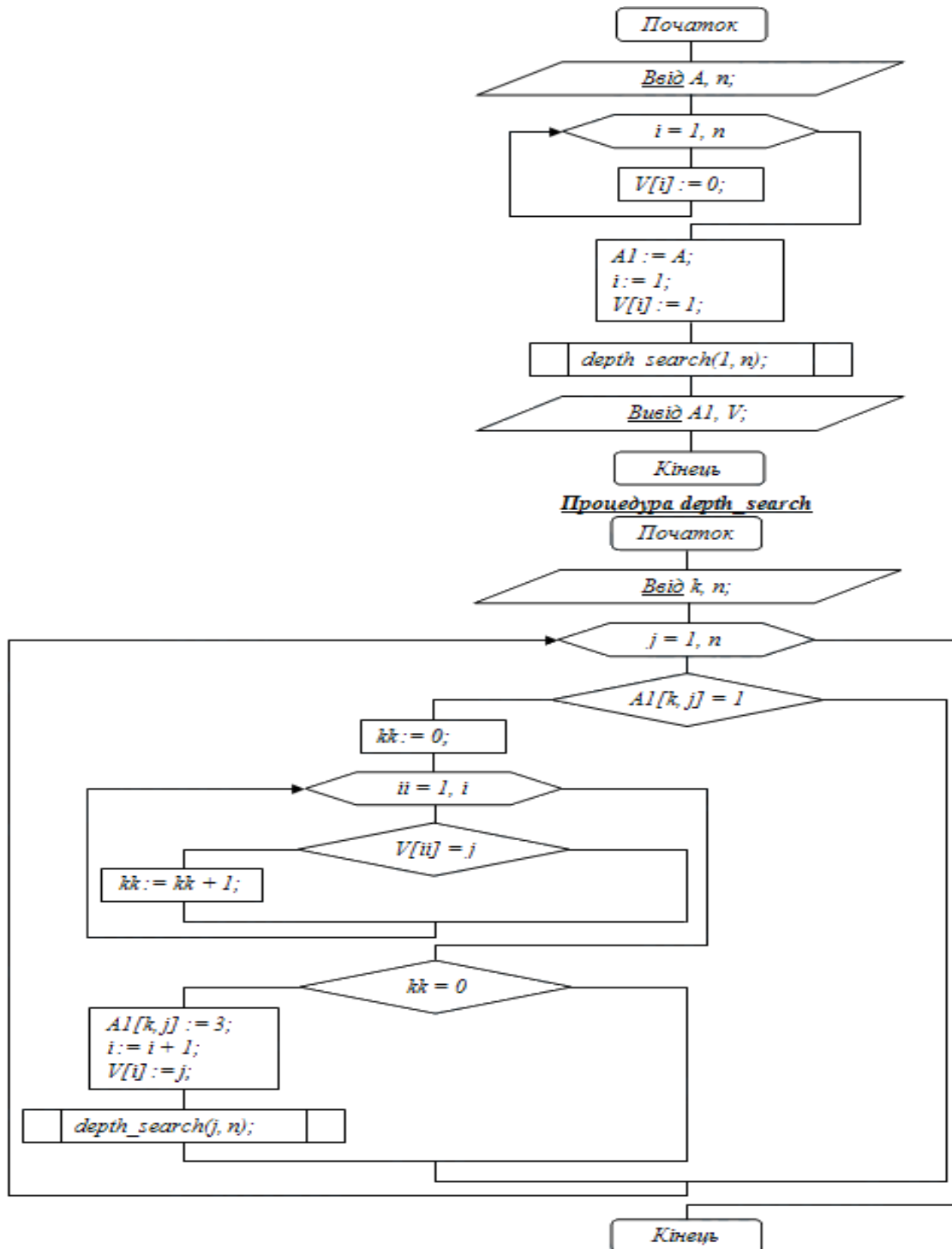
Аналіз та прорахунок результатів:

Іх трьох методів сортування масивів, що представлені в даній програмі, найшвидшим є метод вставок, а найповільнішим - метод вибору. Метод вибору найдоцільніше використовувати для невеликих масивів (забезпечує найбільшу наочність). Метод бульбашок є найбільш простим у реалізації (доцільно використовувати також для невеликих масивів). Метод вставок, завдяки своїй високій швидкості сортування, доцільно використовувати для великих масивів.

Завдання 2.

В основі вирішення даної задачі використано метод обходу графа в глибину, починаючи із заданої точки (старт). Якщо під час обходу не проходимо через точку, введену користувачем як фініш, то шлях між заданими містами відсутній.

Блок-схема алгоритму обходу графа в глибину:



Реалізація програми

```
#include <iostream>
#include <cstdlib>
#include <ctime>
using namespace std;
const int SIZE = 7;
int i, j;
bool *visited = new bool[SIZE];
int map[SIZE][SIZE];
void DFS(int st, int finish)
{
    int r;
    cout << st + 1 << " ";
    visited[st] = true;
    for (r = 0; r <= SIZE; r++)
    {
        if ((map[st][r] != 0) && (!visited[r]))
            DFS(r, finish);
    }
}
int main()
{
    srand(time(nullptr));
    setlocale(LC_ALL, "Rus");
    for (int i = 0; i < SIZE; i++)
    {
        for (int j = i; j < SIZE; j++)
        {
            if (i == j)
```

```

        map[i][j] = 1;
    else
    {
        map[i][j] = rand() % 2;
        map[j][i] = map[i][j];
    }
}

int start, finish;
cout << "Матриця зв'язків між містами: " << endl;
for (i = 0; i < SIZE; i++)
{
    visited[i] = false;
    for (j = 0; j < SIZE; j++)
        cout << " " << map[i][j];
    cout << endl;
}
cout << "Старт >> ";
cin >> start;
cout << "Фініш >> ";
cin >> finish;
cout << "Порядок обходу: ";
DFS(start - 1, finish - 1);
bool isWay = false;
if (visited[finish-1] == true)
    cout << "Існує шлях між містами!\n";
else
    cout << "Шлях між містами відсутній!\n";
delete[]visited;

```

```

    system("pause");
    return 0;
}

```

Результат виконання програми:

```

Матриця зв'язк?в м?ж м?стами:
 1 0 0 0 1 1 0
 0 1 0 0 0 1 0
 0 0 1 1 1 1 0
 0 0 1 1 1 1 0
 1 0 1 1 1 1 1
 1 1 1 1 1 1 0
 0 0 0 0 1 0 1
Старт >> 1
Ф?н?ш >> 7
Порядок обходу: 1 5 3 4 6 2 7 ?снує шлях м?ж м?стами!
Для продовження натисніть будь-яку клавішу . . .

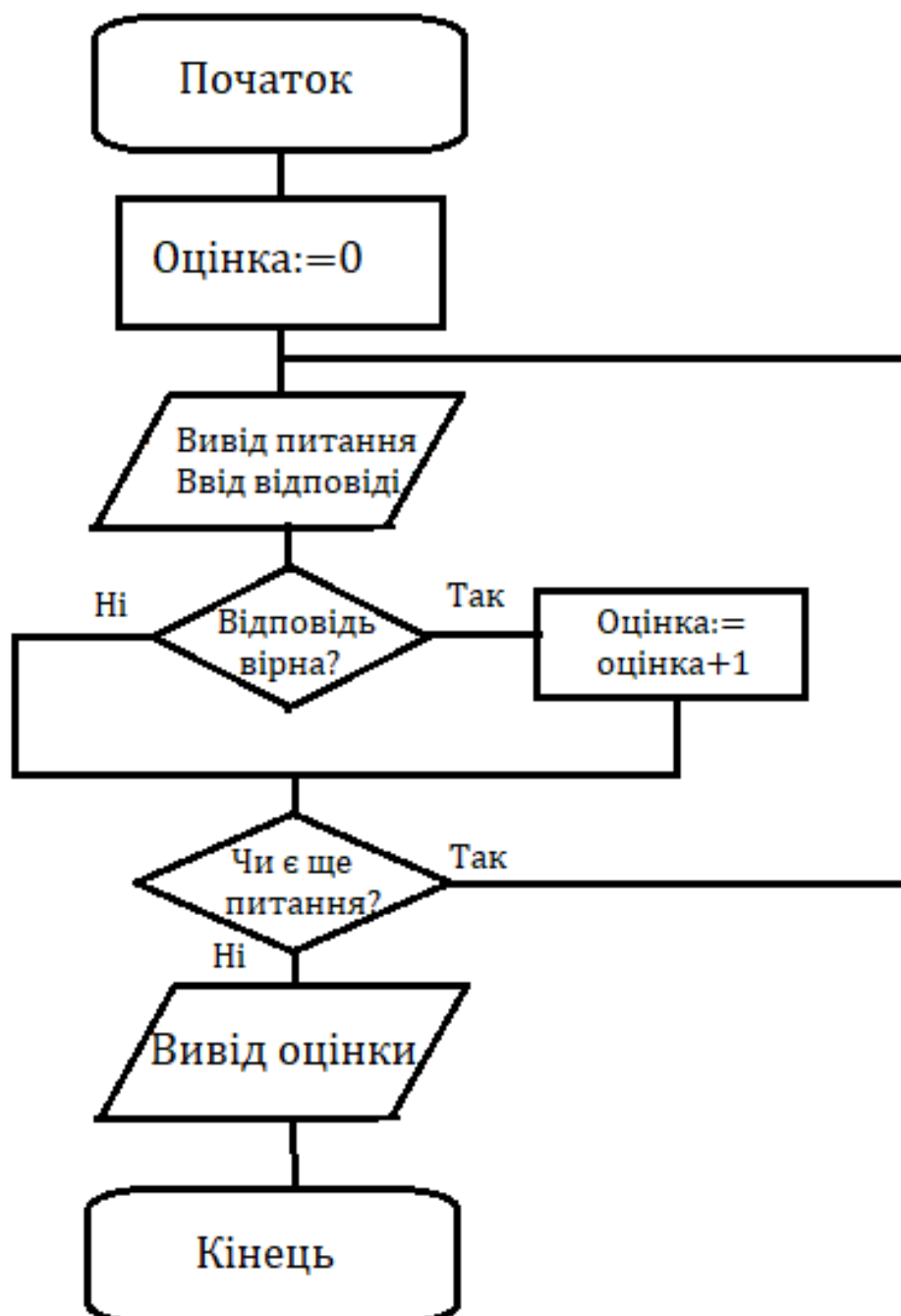
```

Аналіз та прорахунок результатів:

В даній задачі використано алгоритм глибокого обходу графа, починаючи із заданої точки. Якщо під час даного обходу програма не проходить через точку, визначену користувачем як фініш, то . очевидно, що шлях між містами відсутній. Крім глибокого обходу існують також інші алгоритми обходу вершин графа (обхід в ширину, алгоритм Дейкстри, та інші), проте рекурсію доцільно застосовувати саме у методі глибокого обходу.

Завдання 3.

Блок-схема програми:



Реалізація програми

```
#include <iostream>
using namespace std;
int main()
{
    setlocale(LC_ALL, "Ukrainian");
    int rating = 0;
    int answer;
    cout << "Алгоритм, що записано мовою програмування  
називають:\n 1)Блок-схемою\n 2)Програмою\n 3)Модулем\n 4)Командою\n";
    cin >> answer;
    switch (answer)
    {
    case 2:
        rating++;
        break;
    default:
        break;
    }
    cout << "Мова для запису команд у машинних кодах  
називається:\n 1)Мовою програмування низького рівня\n 2)Мовою  
програмування високого рівня\n 3)Машинною мовою\n 4)Об'єктно -  
орієнтованою мовою програмування\n";
    cin >> answer;
    switch (answer)
    {
    case 3:
        rating++;
        break;
```

```

    default:
        break;
}

cout << "Вказівки комп'ютеру на виконання певних дій
називаються ...:\n 1)Вказівниками\n 2)Ідентифікаторами\n
3)Командами\n 4)Умови\n";

cin >> answer;

switch (answer)
{
case 3:
    rating++;
    break;

default:
    break;
}

cout << "У мові Pascal вирази записуються ...:\n 1)В стовпчик\n 2)В
рядок\n 3)В будь-якому вигляді\n 4)По діагоналі\n";

cin >> answer;

switch (answer)
{
case 2:
    rating++;
    break;

default:
    break;
}

cout << "Мови, що описують алгоритми в термінах команд
процесора і наближенні до машинного коду є:\n 1)Мовою програмування
низького рівня\n 2)Мовою програмування високого рівня\n 3)Машинною
мовою\n 4)Об'єсно-орієнтовною мовою програмування\n";

```



```

    cin >> answer;
    switch (answer)
    {
    case 1:
        rating++;
        break;
    default:
        break;
    }

    cout << "Які з наведених записів можуть бути ідентифікаторами?:\n
1)a10t\n 2)if\n 3)end\n 4>true\n";

    cin >> answer;
    switch (answer)
    {
    case 1:
        rating++;
        break;
    default:
        break;
    }

    cout << "Яка із функцій нічого не повертає в мові програмування
C++?:\n 1)int\n 2)main\n 3)bool\n 4)void\n";

    cin >> answer;
    switch (answer)
    {
    case 4:
        rating++;
        break;
    default:

```

```

        break;
    }

    cout << "Якого типу даних не існує?:\n 1)int\n 2)count\n 3)bool\n
4)char\n";

    cin >> answer;

    switch (answer)
    {
    case 2:

        rating++;

        break;

    default:

        break;

    }

    cout << "Який з даних операторів є оператором розгалуження?:\n
1)for\n 2)if\n 3)while\n 4)search\n";

    cin >> answer;

    switch (answer)
    {
    case 2:

        rating++;

        break;

    default:

        break;

    }

    cout << "Що таке компілятор?:\n 1)Програми що перетворюють
текст програми в машинний код, який можна зберегти і після цього
використовувати уже без компілятора\n 2)Програм, що перетворюють
частину програми в машинний код, виконують і після цього переходять
до наступної частини. При цьому щоразу при виконанні програми
використовується компілятор.\n 3)Сервісні програми, що допомагають
керувати програмами\n 4)Конвертери програмного коду, що

```

перетворюють вихідні тексти з однієї мови програмування або його реалізації на іншу\n";

```
    cin >> answer;

    switch (answer)
    {
    case 1:
        rating++;
        break;

    default:
        break;
    }

    cout << "Вітаємо, тест завершено. Ваш результат " << rating << "
    балів із 10\n";

    system("pause");

    return 0;
}
```

Результат виконання програми:

```
1
Яка ?з функцій не повертає в мові програмування C++?:
1)int
2)main
3)bool
4)void
1
Якого типу даних не існує?:
1)int
2)count
3)bool
4)char
1
Який з даних операторів є оператором розгалуження?:
1)for
2)if
3)while
4)search
1
Що таке компілятор?:
1)Програми, що перетворюють текст програми в машинний код, який можна зберегти і після цього використовувати уже без компілятора
2)Програми, що перетворюють частину програми в машинний код, виконують і після цього переходять до наступної частини. При цьому щоразу при виконанні програми використовується компілятор.
3)Сервіси програми, що допомагають керувати програмами
4)Конвертери програмного коду, що перетворюють вихідні тексти з однієї мови програмування або його реалізації на іншу
1
Вітаємо, тест завершено. Ваш результат 3 балів з 10
для продовження натисніть будь-яку клавішу . . .
```

Аналіз та прорахунок результатів:

Вданій програмі використано в якості оператора розгалуження switch, а не if, оскільки таким чином можна уникнути необхідності використовувати вкладені if-інструкції, що спростить код і зробить його більш зрозумілим для користувача.