

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 № 3

з дисципліни «Алгоритмізація та програмування»
на тему "Реалізація алгоритмів з розгалуженням мовою C ++"

XAI.301. 141. 319a. 16 ЛР

Виконав студент гр. 319a

Моїсеєнко Євген
(підпис, дата) (П.І.Б.)

Перевірів

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

2023

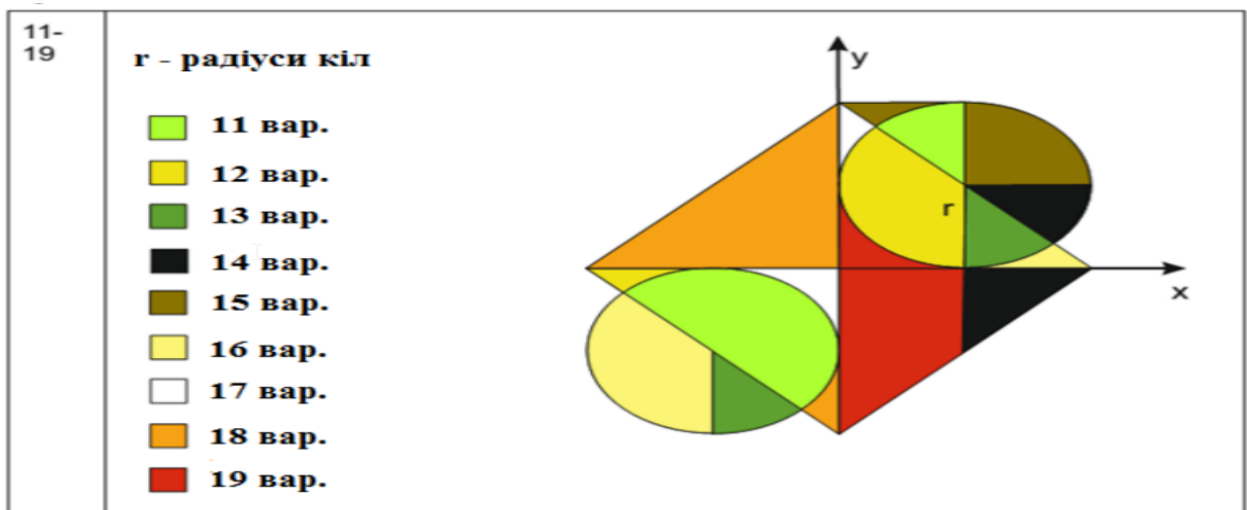
МЕТА РОБОТИ

Вивчити теоретичний матеріал щодо синтаксису у мові C++ і подання у вигляді UML діаграм активності алгоритмів з розгалуженням та реалізувати алгоритми з використанням інструкцій умовного переходу і вибору мовою C++ в середовищі Visual Studio. Також опанувати та відпрацювати навички структурування програми з функціями.

ПОСТАНОВКА ЗАДАЧІ

Завдання 1. Вирішити завдання на алгоритми з розгалуженням.If9. Дано дві змінні дійсного типу: A, B. Перерозподілити значення даних змінних так, щоб в A було записано менше зі значень, а в B – більше. Вивести нові значення змінних A і B.

Завдання 2. Дано координати точки на площині (x, y). Визначити, чи потрапляє точка в фігуру заданого кольору (або групу фігур) і вивести відповідне повідомлення.



Завдання 3. Обчислити площу і периметр плоскої фігури.

Завдання 4. Для вибору користувачем одного з трьох зазначених вище завдань розробити алгоритм організації меню в командному вікні з використанням інструкції вибору.

ВИКОНАННЯ РОБОТИ

Завдання 1.

Вирішення задачі If9.

A: Змінна для збереження значення першого числа. Дійсного типу.

B: Змінна для збереження значення другого числа. Дійсного типу.

temp: Тимчасова змінна, що використовується для збереження значення A перед обміном значень A і B. Вона допомагає виконати операцію обміну без втрати даних.

Алгоритм вирішення показано на рис.1

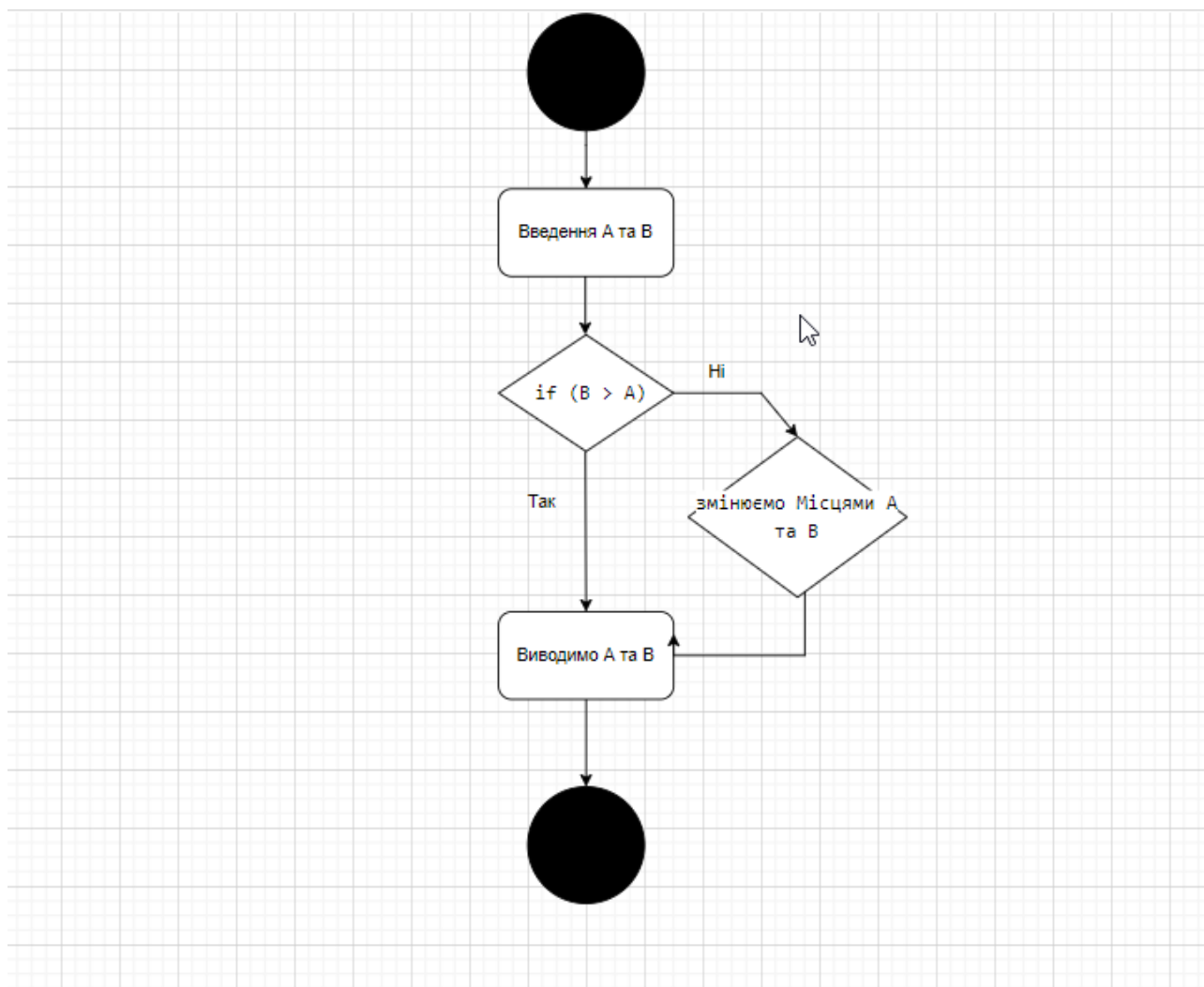


Рисунок 1 – Діаграма активності для алгоритму вирішення задачі If9.

Лістинг коду вирішення задачі. А (стор.6-9).

Екран роботи програми показаний на рис. Б.1.

Завдання 2.

Вирішення задачі Геометричні фігури.

Вхідні дані

point (структура Point): Координати введеної точки.

point.x (тип int): Координата x точки.

point.y (тип int): Координата y точки.

color (тип int): Колір фігури, введений користувачем.

Вихідні дані:

figures[].color (тип int): Колір фігури.

figures[].center (структура Point): Координати центра фігури.

figures[].center.x (тип int): Координата x центра фігури.

figures[].center.y (тип int): Координата y центра фігури.

figures[].radius (тип int): Радіус фігури.

isInside (тип bool): Логічне значення, чи потрапляє точка в обрану фігуру.

perimeter (тип int): Розрахований периметр фігури.

area (тип int): Розрахована площа фігури.

Алгоритм вирішення показано на рис.2

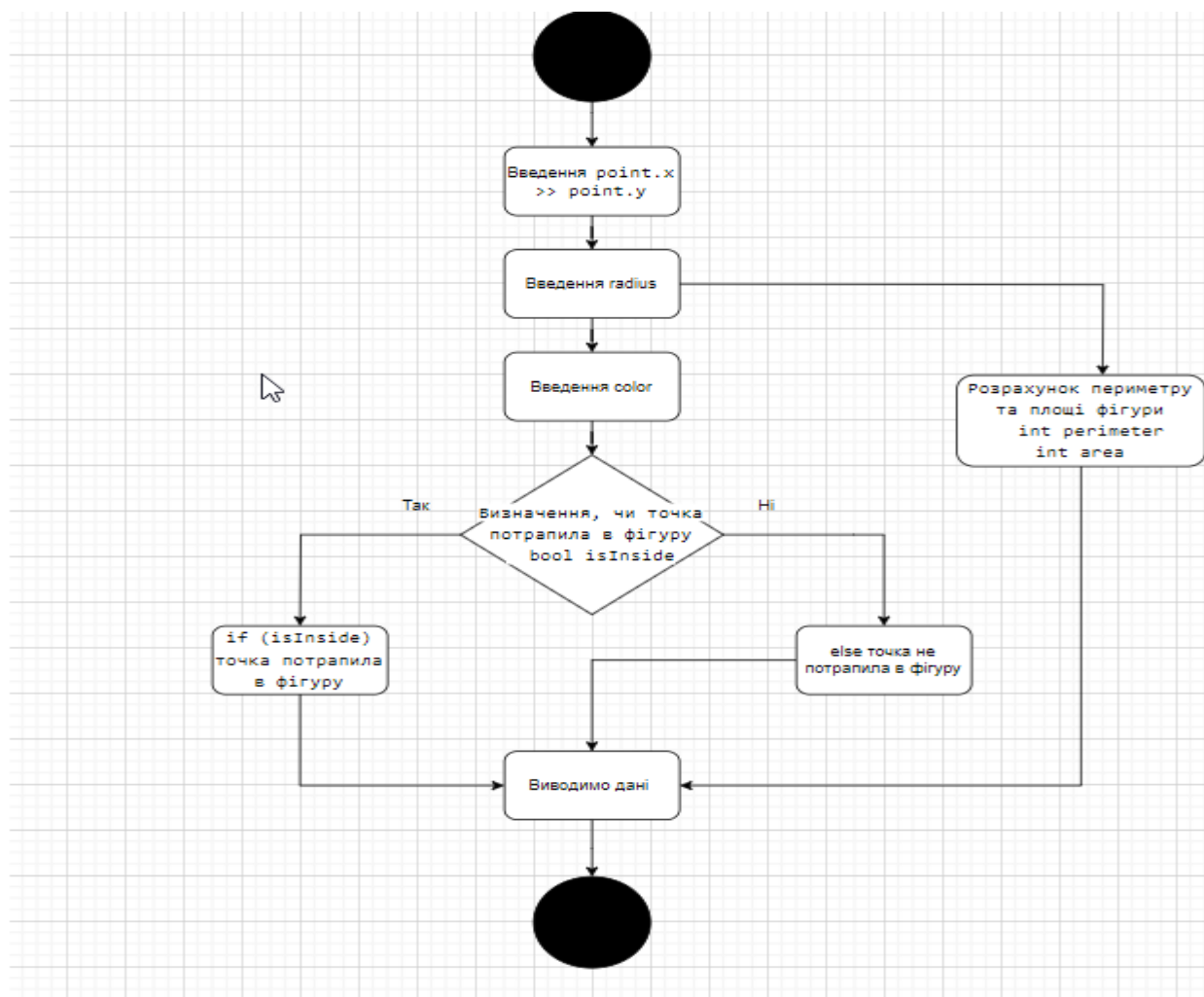


Рисунок 2 – Діаграма активності для алгоритму вирішення задачі Геометричні фігури.

Лістинг коду вирішення задачі. А (стор.6-9).

Екран роботи програми показаний на рис. Б.2.

ВИСНОВКИ

У ході даної роботи було детально вивчено процес створення програм на мові програмування C++, в яких використовуються умовні оператори та розгалуження. Головна мета полягала у розумінні методів написання коду з використанням цих інструкцій, а також в структуруванні програми за допомогою функцій. Був ознайомлений теоретичний матеріал, алгоритми були

відображені з використанням UML діаграм активності, й після цього було практично реалізовано ці алгоритми.

ДОДАТОК А

Лістинг коду програми

```
#include <iostream>
#include <cmath>

using namespace std;

// Прототипи функцій
void task_if1(); // Прототип функції для першого завдання
void task_geom1(); // Прототип функції для другого завдання

int main() {
    int menu;
    cout << "Task number:";
    cin >> menu;
    switch (menu) {
        case 1:
            task_if1(); // Виклик функції для першого завдання
            break;
        case 2:
            task_geom1(); // Виклик функції для другого завдання
            break;
        default:
            cout << "Wrong task! (Only 1, 2)" << endl; // Помилкове введення
номера завдання
            break;
    }

    return 0; // Повернення значення з main()
}

void task_if1() {
    double A, B;

    // Зчитування значень змінних A і B
    cout << "Введіть значення для A: ";
    cin >> A;
    cout << "Введіть значення для B: ";
    cin >> B;

    if (B > A) {
        // Якщо B вже більше за A, виводимо їх без змін
        cout << "A: " << A << endl;
        cout << "B: " << B << endl;
    } else {
        // Інакше міняємо значення A та B місцями
    }
}
```

```

        double temp = A;
        A = B;
        B = temp;

        cout << "Після перерозподілу:" << endl;
        cout << "A: " << A << endl;
        cout << "B: " << B << endl;
    }
}

struct Point {
    int x;
    int y;
};

struct Figure {
    int color;
    Point center;
    int radius;
};

// Перевірка чи точка знаходиться всередині кола фігури
bool isInsideCircle(Point point, Figure figure) {
    return pow(point.x - figure.center.x, 2) + pow(point.y - figure.center.y, 2)
    <= pow(figure.radius, 2);
}

void task_geom1() {
    // Введення даних про точку і колір фігури
    Point point;
    cout << "Введіть координати точки: ";
    cin >> point.x >> point.y;
    int color;
    cout << "Введіть колір фігури: ";
    cin >> color;

    // Створення масиву фігур (у цьому випадку масив має лише один елемент)
    Figure figures[] = {
        {12, {5, 5}, 5} // фігура з певним кольором, центром та радіусом
    };

    // Пошук фігури з введеним кольором
    int i;
    for (i = 0; i < 1; i++) { // Пошук фігури в масиві (в даному випадку одна фігура)
        if (figures[i].color == color) {
            break; // Якщо знайдено фігуру з введеним кольором, вийти з циклу
        }
    }
}

```



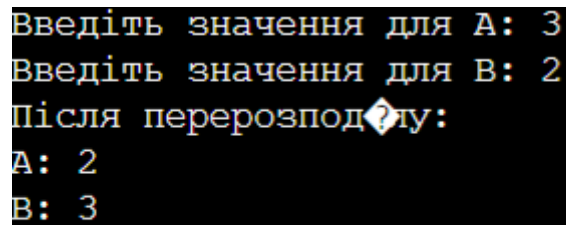
```
// Визначення, чи точка потрапила всередину фігури
bool isInside = isInsideCircle(point, figures[i]);

// Виведення результату
if (isInside) {
    cout << "Точка потрапила в фігуру." << endl;
} else {
    cout << "Точка не потрапила в фігуру." << endl;
}

// Розрахунок периметру та площі фігури
double perimeter = 2 * M_PI * figures[i].radius; // Обчислення периметру
кола
double area = M_PI * figures[i].radius * figures[i].radius; // Обчислення
площі кола

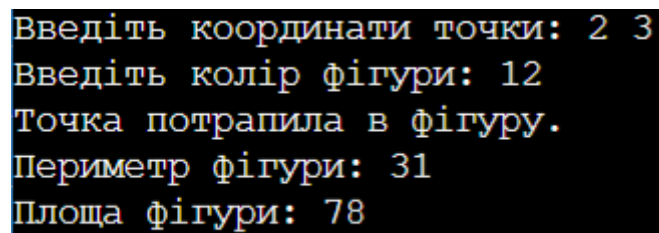
// Виведення результату
cout << "Периметр фігури: " << perimeter << endl;
cout << "Площа фігури: " << area << endl;
}
```

ДОДАТОК Б
Скрін-шоти вікна виконання програми



```
Введіть значення для А: 3  
Введіть значення для В: 2  
Після перерозподілу:  
А: 2  
В: 3
```

Рисунок Б.1 – Екран виконання програми для вирішення завдання Іf9.



```
Введіть координати точки: 2 3  
Введіть колір фігури: 12  
Точка потрапила в фігуру.  
Периметр фігури: 31  
Площа фігури: 78
```

Рисунок Б.2 – Екран виконання програми для вирішення завдання Геометричні фігури