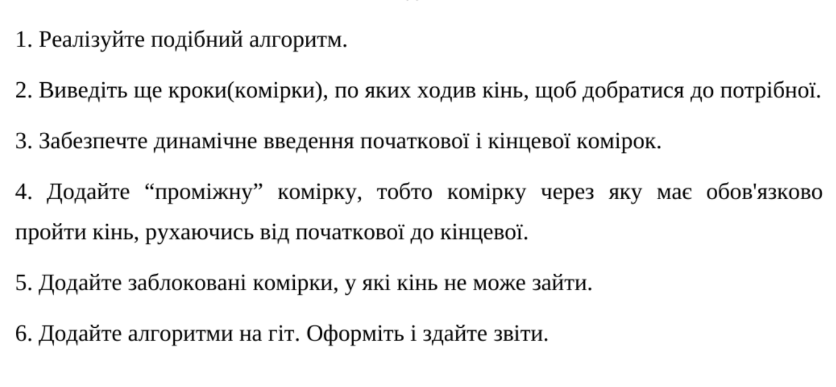
**ЗВІТ**

про виконання лабораторної роботи 6-7

«**Задачі-ігри**»  
з дисципліни  
«Методика розв'язування олімпіадних задач» студента(ки) групи ІН-2327Б  
Чухрая Олександра Васильовича

**Умова завдання:**

****

**Виконання:**

using System;

using System.Collections.Generic;

class KnightMoves

{

static readonly int[,] moves = {

{2, 1}, {2, -1}, {-2, 1}, {-2, -1},

{1, 2}, {1, -2}, {-1, 2}, {-1, -2}

};

static List<(int, int)> FindShortestPath((int, int) start, (int, int) mid, (int, int) end, HashSet<(int, int)> blocked)

{

List<(int, int)> pathToMid = BFS(start, mid, blocked);

List<(int, int)> pathToEnd = BFS(mid, end, blocked);

if (pathToMid == null || pathToEnd == null)

return null;

pathToMid.AddRange(pathToEnd.GetRange(1, pathToEnd.Count - 1));

return pathToMid;

}

static List<(int, int)> BFS((int, int) start, (int, int) target, HashSet<(int, int)> blocked)

{

Queue<List<(int, int)>> queue = new Queue<List<(int, int)>>();

HashSet<(int, int)> visited = new HashSet<(int, int)>();

queue.Enqueue(new List<(int, int)> { start });

visited.Add(start);

while (queue.Count > 0)

{

var path = queue.Dequeue();

var lastPos = path[path.Count - 1]; // Using Count - 1 instead of ^1

int x = lastPos.Item1, y = lastPos.Item2;

if (x == target.Item1 && y == target.Item2)

return path;

for (int i = 0; i < 8; i++) // Iterating through possible moves

{

int nx = x + moves[i, 0], ny = y + moves[i, 1];

if (nx >= 1 && nx <= 8 && ny >= 1 && ny <= 8 &&

!visited.Contains((nx, ny)) && !blocked.Contains((nx, ny)))

{

List<(int, int)> newPath = new List<(int, int)>(path) { (nx, ny) };

queue.Enqueue(newPath);

visited.Add((nx, ny));

}

}

}

return null;

}

static void Main()

{

Console.Write("Enter the starting position (x y): ");

var start = ReadPosition();

Console.Write("Enter the intermediate position (x y): ");

var mid = ReadPosition();

Console.Write("Enter the target position (x y): ");

var end = ReadPosition();

Console.Write("Enter the number of blocked positions: ");

int blockCount = int.Parse(Console.ReadLine());

HashSet<(int, int)> blocked = new HashSet<(int, int)>();

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

{

Console.Write($"Enter blocked position {i + 1} (x y): ");

blocked.Add(ReadPosition());

}

var path = FindShortestPath(start, mid, end, blocked);

if (path != null)

{

Console.WriteLine($"Minimum number of moves: {path.Count - 1}");

Console.WriteLine("Path:");

foreach (var pos in path)

Console.WriteLine($"({pos.Item1}, {pos.Item2})");

}

else

{

Console.WriteLine("No path found.");

}

// Prevent console from closing immediately

Console.WriteLine("Press any key to exit...");

Console.ReadKey();

}

static (int, int) ReadPosition()

{

var input = Console.ReadLine().Split();

return (int.Parse(input[0]), int.Parse(input[1]));

}

}

Результат:

