ЗВІТ

про виконання лабораторної роботи № 11

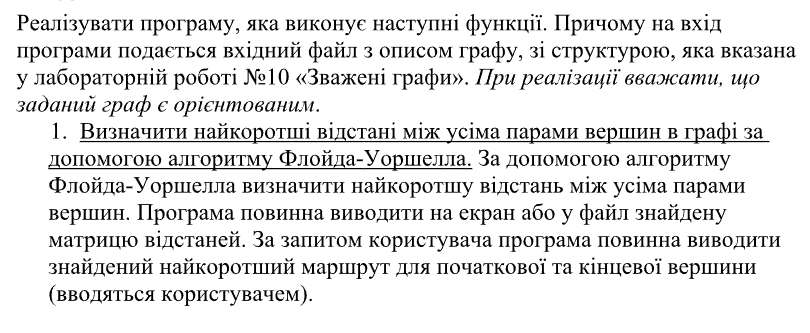
*Алгоритм Флойда-Уоршелла*

з дисципліни «Програмування дискретних структур»

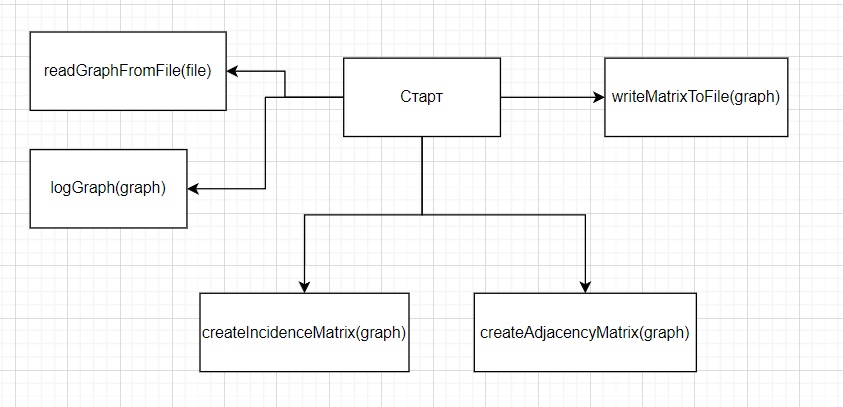
студента групи КН-2226

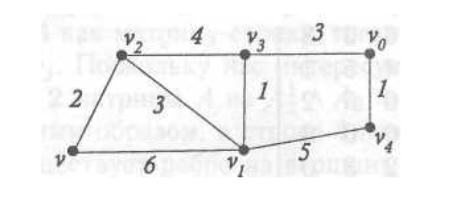
*Веждел Андрій Іванович*

**Умова завдання:**



**Блок схема:**

****

**Граф:  
**

**Текст коду:**

const fs = require("fs");

function readGraphFromFile(filename) {

  try {

    const data = fs.readFileSync(filename, "utf8");

    const lines = data.split("\n");

    const [n, m] = lines[0].split(" ").map(Number);

    const list = {};

    for (let i = 1; i < lines.length; i++) {

      let [v, u, w] = lines[i].split(" ");

      if (!u || u === undefined) {

        u = [];

      }

      if (!list[v]) {

        list[v] = [];

      }

      list[v].push({ end: u, w: Number(w) });

    }

    return { n, m, list };

  } catch (error) {

    console.error("Помилка при читанні файлу:", error.message);

    return null;

  }

}

function logGraph(graph) {

  const list = graph.list;

  for (const key in list) {

    const neighbors = list[key];

    neighbors.forEach((neighbor) => {

      console.log(` ${key} --> ${neighbor.end} (вага: ${neighbor.w})`);

    });

  }

}

function floydWarshall(graph) {

  let { n, list } = graph;

  const floydMatrix = Array.from({ length: n }, () => Array(n).fill(Infinity));

  for (let i = 0; i < n; i++) {

    floydMatrix[i][i] = 0;

  }

  let count\_row = 0;

  for (let row\_key in list) {

    let count\_column = 0;

    for (let column\_key in list) {

      if (list[row\_key].some((item) => item.end === column\_key)) {

          floydMatrix[count\_row][count\_column] = list[row\_key].find((item) => item.end === column\_key).w;

      }

      count\_column++;

    }

    count\_row++;

  }

  for (let k = 0; k < n; k++) {

    for (let i = 0; i < n; i++) {

      for (let j = 0; j < n; j++) {

        if (floydMatrix[i][k] + floydMatrix[k][j] < floydMatrix[i][j]) {

          floydMatrix[i][j] = floydMatrix[i][k] + floydMatrix[k][j];

        }

      }

    }

  }

  return floydMatrix;

}

function writeMatrixToFile(matrix, filename) {

    const data = matrix.map(row => row.map(cell => cell.toString().padEnd(3, ' ')).join(" ")).join("\n");

    fs.writeFileSync(filename, data, "utf8");

    console.log(`Матрицю записано у файл: ${filename}`);

}

function findShort(column,row){

    return floydMatrixResult[column][row];

}

const graph = readGraphFromFile("./lb11/lb11.txt");

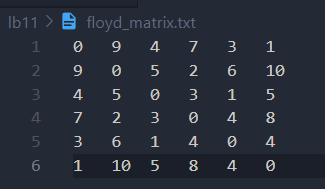
let floydMatrixResult = floydWarshall(graph);

logGraph(graph);

console.log(findShort(0,4));

writeMatrixToFile(floydMatrixResult, "./lb11/floyd\_matrix.txt");

**Результат виконання:**

****

**Посилання на гіт-репозиторій:**

[disc-systems/lb11/lb11.js at main · 1duxa/disc-systems (github.com)](https://github.com/1duxa/disc-systems/blob/main/lb2.js)

**Результат юніт-тесту:**

const lb6 = require("./lb6");

test("Матриця інцидентності", () => {

    const incidenceMatrix = [[-1,-1,0,0,0,1],[1,0,-1,0,0,0],[0,1,0,-1,0,0],[0,0,1,1,-1,0],[0,0,0,0,1,-1]];

    const graph = {

        n: 5,

        m: 6,

        list: {

            "1": ["2", "3"],

            "2": ["4"],

            "3": ["4"],

            "4": ["5"],

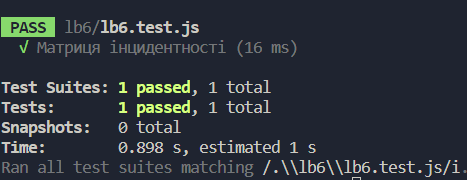
            "5": ["1"],

        },

    };

    expect(lb6.createIncidenceMatrix(graph)).toEqual(incidenceMatrix);

});

****

**Висновок:**

Розглянув та вивчити різні типи представлення зважених графів в пам’яті

комп`ютерів.