**Прізвище:** Дуда

**Ім'я:** Костянтин

**Група:** КН-407

**Варіант:** 11

**Кафедра:** САПР

**Дисципліна:** Дискретні моделі в системному проектуванні

**Перевірив:** Кривий Р. З.

**Дата:**

**ЗВІТ**

до лабораторної роботи №5

на тему "Ізоморфізм графів"

**Мета роботи:** Метою лабораторної роботи є вивчення і дослідження основних підходів до встановлення ізоморфізму графів.

**Короткі теоретичні відомості:**

Теорія графів дає простий, доступний і потужній інструмент побудови моделей і рішення задач впорядкування взаємозвязаних обєктів. Нині є багато проблем де необхідно дослідити деякі складні системи з допомогою впорядкування їх елементів. До таких проблем відносяться і задачі ідентифікації в електричних схемах, в авіації, в органічній хімії і т.д. Вирішення таких проблем досягається з допомогою встановлення ізоморфізму графів. Два графа G=(X,U,P) і G'=(X',U',P') називаються ізоморфними, якщо між їх вершинами, а також між їхніми ребрами можна встановити взаємно однозначне співвідношення X <-> X', U <-> U', що зберігає інцидентність, тобто таке, що для всякої пари (x,y)єX ребра u є U, що з'єднує їх, обов'язково існує пара (x',y') є X' і ребро u' є U', що з'єднує їх, і навпаки. Тут P - предикат, інцидентор графа G. Зауважимо, що відношення ізоморфізму графів рефлексивне, симетричне і транзитивне, тобто представляє собою еквівалентність. На даний час існує досить детальна класифікація розроблених методів рішення такого типу задач /1/. Розглядаючи комбінаторно-логічну природу вказаної задачі можна всі роботи в цьому напрямку розділити на дві групи: рішення теоретичної задачі встановлення ізоморфізму простих графів; розробка наближених методів, які найбільш повно враховують обмеження і специфіку задачі з застосуванням характерних ознак об’єкту дослідження. До першої групи відносяться алгоритми: повного перебору і почергового “підвішування” графів за вершини. Наближені алгоритми використовують або допустимі рішення точних методів, або побудованні на використанні евристичних прийомів. На даний час створена певна кількість алгоритмів встановлення ізоморфізму графів, які за рахунок різних евристичних прийомів знижують складність задачі від факториальної до степеневої функції.

**Код програми:**

<https://github.com/Kruvyi/Lab5_DS.git>

package com.kostya.isomorphism\_lab5

import com.kostya.{Edge, Graph}

class Isomorphism(isomorphicGraph: Graph) {

var general: Boolean = false

def changeVerticeNames(graph: Graph): Unit = {

val initialVertices = graph.getVertices

val isFirst = true

if (graph.edges.length != isomorphicGraph.edges.length) return

else {

for (edge <- graph.edges) {

edge.vertex1 = getVerticeName(edge, isFirst)

edge.vertex2 = getVerticeName(edge, !isFirst)

}

}

def getVerticeName(edge: Edge, isFirst: Boolean): String = {

if (isFirst)

isomorphicGraph.getVertices.toSeq(initialVertices.toSeq.indexOf(edge.vertex1))

else isomorphicGraph.getVertices.toSeq(initialVertices.toSeq.indexOf(edge.vertex2))

}

}

def antiFlex(graph: Graph, m: Int): Unit = {

if (m == 0) {

if (isomorphicGraph.equals(graph)) {

general = true

}

} else {

for (i <- 0 to m) {

this.antiFlex(graph, m - 1)

if (i < m) {

graph.swap(graph.getVertices.toSeq(i), graph.getVertices.toSeq(m))

reverse(graph, m - 1)

}

}

}

}

def reverse(graph: Graph, m: Int): Unit = {

var i = 0

var j = m

while (i < j) {

graph.swap(graph.getVertices.toSeq(i), graph.getVertices.toSeq(j))

i += 1; j -= 1

}

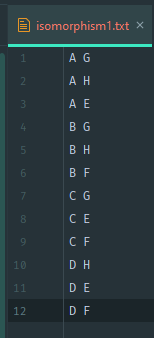
}

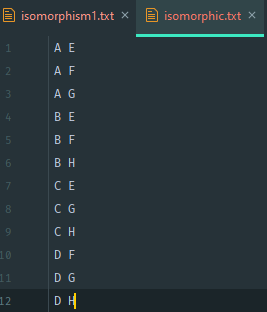
}

**Інструкція користувача:**

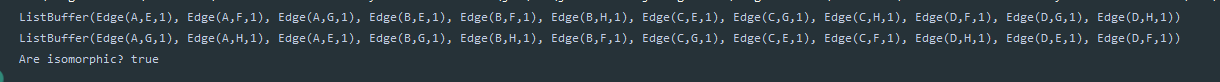
Для роботи програмі потрібно текстовий файл зі списком ребер двох графів. Кожне ребро записується з нового рядка, через пробіл записується перша вершина ребра, друга вершина. Коли вхідний файл буде готовий потрібно запустити програму, після того як програма завершить свою роботу на екран буде виведено, чи ізоморфні задані графи, чи ні.

**Вхідні файли:**





**Скріншот результату:**



**Висновок:** в цій лабораторній роботі було досліджено основні підходи до встановлення ізоморфізму графів, а також написано реалізацію алгоритму на Scala.