Министерство образования Республики Беларусь

Учреждение образования

«Брестский государственный технический университет»

Кафедра ИИТ

Лабораторная работа № 6

По дисциплине ЯПИС

За шестой семестр

## Тема: «Метод поиска в глубину на графе. Рекурсивная и не рекурсивная реализации алгоритма DFS»

Вариант 7

Выполнил:

Студент 3 курса

Группы ИИ-16 (2)

Пешко А.С.

Проверил:

Слинко Е.В.

Брест, 2021

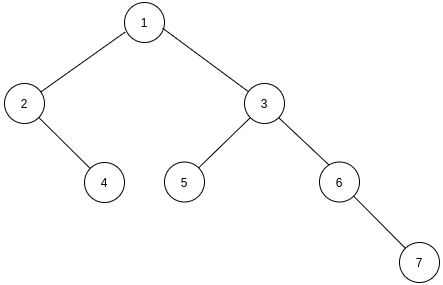
**Цель:** Сформировать представление о методе и алгоритме поиска в глубину на графе и его применении при решении задач (задачи класса сложности Р).

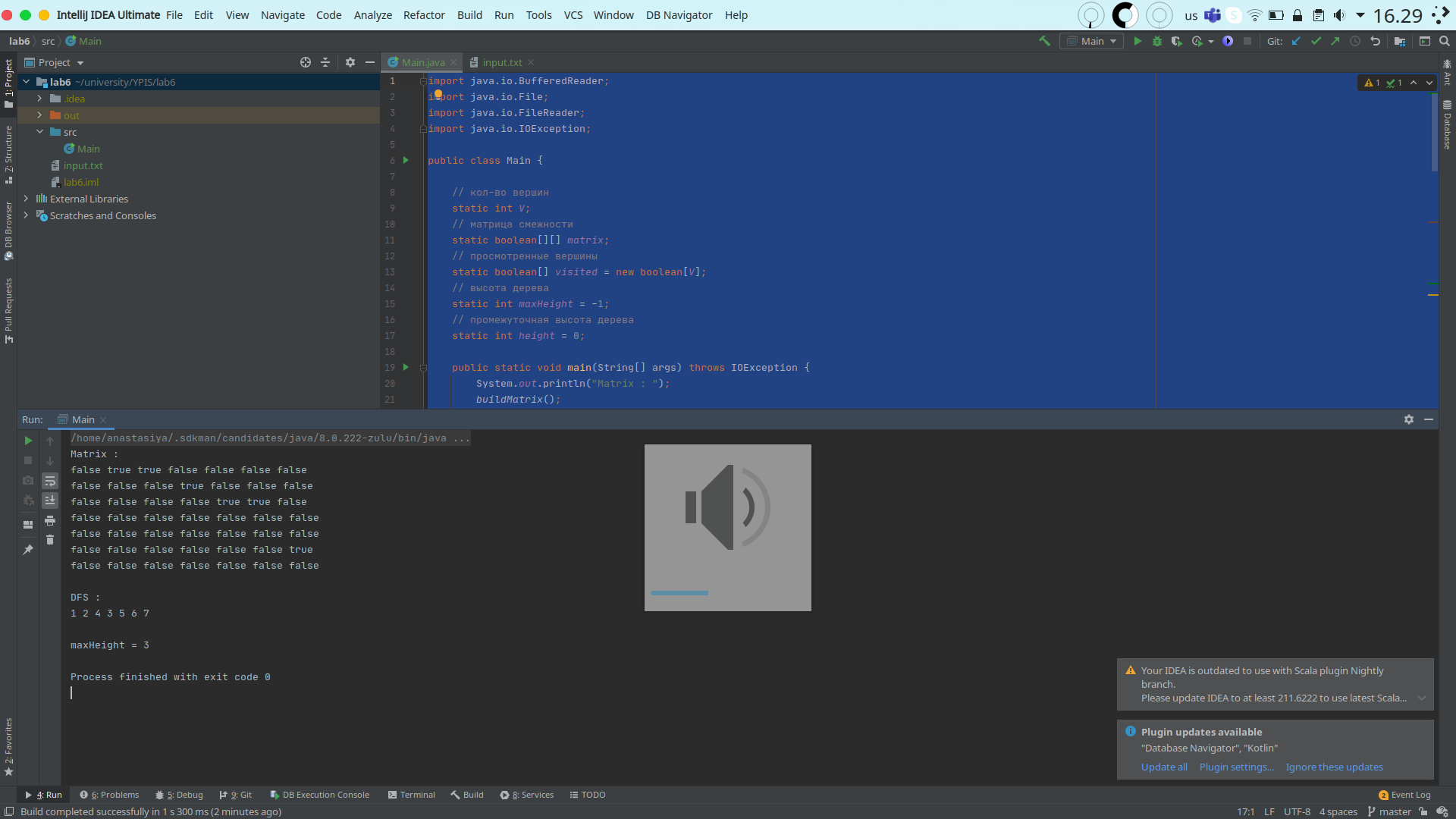
**Ход работы**

**Вариант 7.** Напишите программу, реализующую нахождение дерева поиска DFS, которое представлено в виде родительских связей (массив, индексированный номерами вершин, содержащий элементы – родительскую вершину). Определите высоту полученного дерева (высота дерева определяется как длина пути от корня к самому удаленному листу). Используйте представление графа в виде матрицы смежности.

**Код программы:**

import java.io.BufferedReader;  
import java.io.File;  
import java.io.FileReader;  
import java.io.IOException;  
  
public class Main {  
  
 // кол-во вершин  
 static int V;  
 // матрица смежности  
 static boolean[][] matrix;  
 // просмотренные вершины  
 static boolean[] visited = new boolean[V];  
 // высота дерева  
 static int maxHeight = -1;  
 // промежуточная высота дерева  
 static int height = 0;  
  
 public static void main(String[] args) throws IOException {  
 System.out.println("Matrix : ");  
 buildMatrix();  
  
 visited = new boolean[V];  
  
 System.out.println("\nDFS :");  
 DFS(0);  
  
 System.out.println("\n\nmaxHeight = " + (maxHeight - 1));  
 }  
  
 static void DFS(int start) {  
 int r;  
 System.out.print(start + 1 + " ");  
 visited[start] = true;  
 for (r = 0; r < V; r++) {  
 if (matrix[start][r] && !visited[r]) {  
 height++;  
 DFS(r);  
 }  
 }  
 if (height > maxHeight){  
 maxHeight = height;  
 height = 0;  
 }  
 }  
  
 private static void buildMatrix() throws IOException {  
  
 File file = new File("input.txt");  
 FileReader fr = new FileReader(file);  
 BufferedReader reader = new BufferedReader(fr);  
  
 int count = 0;  
 String line = "";  
  
 while ((line = reader.readLine()) != null) {  
 if (count == 0) {  
 V = Integer.parseInt(line);  
 matrix = new boolean[V][V];  
 } else {  
 String[] v1andv2 = line.split(" ");  
 int v1 = Integer.parseInt(v1andv2[0]);  
 int v2 = Integer.parseInt(v1andv2[1]);  
  
 // устанавливаем связь между вершинами v1 и v2  
 matrix[v1 - 1][v2 - 1] = true;  
 }  
 count++;  
 }  
 for (int i = 0; i < V; i++) {  
 for (int j = 0; j < V; j++) {  
 System.out.print(matrix[i][j] + " ");  
 }  
 System.out.println();  
 }  
 }  
}

Результат работы программы: Заданный граф:



**Вывод:** Сформировали представление о методе и алгоритме поиска в глубину на графе и его применении при решении задач.