

Penyusunan Rencana Kuliah dengan Topological Sort (Penerapan Decrease and Conquer)

**Laporan Tugas Kecil II
Mata kuliah IF2211 Strategi Algoritma**



Disusun Oleh :

Juan Louis Rombetasik (13519075)

**Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung
Semester 2 Tahun 2020/2021**

ALGORTIMA TOPO-SORT

Merekursif List yang berisi Object Graph dengan dan men pop indeks 0 dari list;

Memasukkan ke himpunan solusi mereka yang tidak memiliki prerequisite

Memasukkan ke himpunan solusi mereka yang memiliki prerequisite yang sudah di dalam himpunan solusi

Jika tidak dimasukkan kedalam himpunan solusi maka isi list pada saat itu akan dimasukkan ke dalam List Graph ke posisi paling akhir

Menoutput (log) himpunan list solusi (satu mata kuliah per semester) jika List sudah kosong (semua Mata kuliah sudah masuk kedalam himpunan solusi)

https://github.com/mizuday/Tucil2_13519075

KODE PROGRAM

```
class Main {
    public static void main(String[] args) {
        while (true) {
            System.out.println();
            Scanner sc= new Scanner(System.in);
            System.out.print("nama file (tanpa txt): ");
            String namafile = sc.nextLine();

            List<Graph<String>> graph = Graph.fileToGraph("../test/" + namafile + ".txt");

            List<String> solusi = new ArrayList<>();
            Graph.solver(graph, solusi);
            System.out.print("solve lagi? y/n >> ");
            String in = sc.nextLine().toLowerCase();
            if (in.equals("n") || in.equals("no")){
                break;
            }
        }
    }
}

class Graph<T> {
    private T prec;
    private T succ;

    public Graph(T preccc, T succc){
        prec = preccc;
        succ = succc;
    }
}
```

```

}

public T getPrec(){
    return prec;
}

public T getSucc(){
    return succ;
}

public void cetakIsi(){
    System.out.println "{" + prec + ", " + succ + " }";
}

public static boolean punyaPrec(List<Graph<String>> lg, Graph<String> g){
    int i = 0;
    while (i < lg.size()) {
        if (lg.get(i).getSucc().equals(g.getPrec()))
            return true;
        else
            i += 1;
    }
    return false;
}

public static boolean dalamList(List<Graph<String>> lg, String str){
    for (Graph<String> stringGraph : lg) {
        if (stringGraph.getSucc().equals(str) || stringGraph.getPrec().equals(str))
            return true;
    }
    return false;
}

public static void solver(List<Graph<String>> graph, List<String> solusi){
    if (!graph.isEmpty()) {
        try {
            if (punyaPrec(graph, graph.get(0))) {
                graph.add(graph.get(0));
            }
            else {
                if (!solusi.contains(graph.get(0).getPrec()))
                    solusi.add(graph.get(0).getPrec());

                if (!dalamList(graph, graph.get(0).getSucc())) {
                    solusi.add(graph.get(0).getSucc());
                }
            }
        }
        catch (Exception e) {}
    }
}

```

```

        }
    }
    graph.remove(0);
    solver(graph, solusi);
} catch (NullPointerException e) {
    System.out.println("NullPointerException Caught");
}

}

else {
    int counter = 0;
    for (String string : solusi){
        System.out.println("Semester "+ (counter+1) + ": " +string);
        counter++;
    }
}

}

}

public static List<Graph<String>> fileToGraph(String namafile){
    List<Graph<String>> graph = new ArrayList<>();
    List<String[]> data = new ArrayList<>();
    try {
        BufferedReader reader = new BufferedReader(new FileReader(namafile));
        String line = reader.readLine();
        while (line != null) {
            line = line.replace(".", "");
            String[] temp = line.split(", ");
            data.add(temp);
            line = reader.readLine();
        }
        reader.close();

        boolean akhir;
        for (int i = 0; i < data.size(); i++) {
            akhir = true;
            for (int j = 0; j < data.size(); j++) {
                if (j == i) {
                }
                else {
                    for (int k = 0; k < data.get(j).length; k++) {
                        if (data.get(i)[0].equals(data.get(j)[k])) {
                            graph.add(new Graph<>(data.get(i)[0], data.get(j)[0]));
                            akhir = false;
                        }
                    }
                }
            }
        }
    }
}

```

```

    }
    }
    if (akhir)
        graph.add(new Graph<>(data.get(i)[0], ""));
    }

} catch (IOException e) {
    System.out.println("Nama file salah");
}

return graph;
}
}

```

SKRINSHUT OUTPUT PROGRAM

```

nama file (tanpa txt): test
Nama file salah
solve lagi? y/n >> y

nama file (tanpa txt): contoh1
Semester 1: C1
Semester 2: C2
Semester 3: C3
Semester 4: C4
Semester 5: C5
solve lagi? y/n >> y

nama file (tanpa txt): matkul
Semester 1: C3
Semester 2: C1
Semester 3: C4
Semester 4: C2
Semester 5: C5
solve lagi? y/n >> y

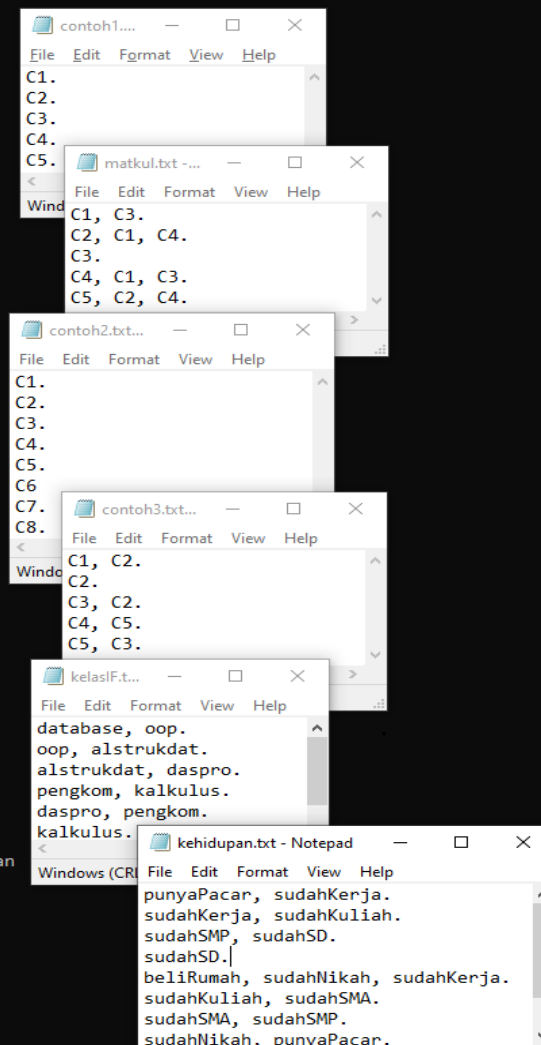
nama file (tanpa txt): contoh2
Semester 1: C1
Semester 2: C2
Semester 3: C3
Semester 4: C4
Semester 5: C5
Semester 6: C6
Semester 7: C7
Semester 8: C8
solve lagi? y/n >> y

nama file (tanpa txt): contoh3
Semester 1: C2
Semester 2: C3
Semester 3: C5
Semester 4: C1
Semester 5: C4
solve lagi? y/n >> y

nama file (tanpa txt): kelasIF
Semester 1: kalkulus
Semester 2: pengkom
Semester 3: daspro
Semester 4: alstrukdat
Semester 5: oop
Semester 6: database
solve lagi? y/n >> kehidupan

nama file (tanpa txt): kehidupan
Semester 1: sudahSD
Semester 2: sudahSMP
Semester 3: sudahSMA
Semester 4: sudahKuliah
Semester 5: sudahKerja
Semester 6: punyaPacar
Semester 7: sudahNikah
Semester 8: beliRumah
solve lagi? y/n >>

```



```
D:\Kuliah\Tucil2_13519075\src>java com.mizuday.Main
```

```
nama file (tanpa txt): kelasHarvard
```

```
Semester 1: basic_programming
```

```
Semester 2: algorithms
```

```
Semester 3: data_structures
```

```
Semester 4: introduction_to_game_development
```

```
Semester 5: object_oriented_programming
```

```
Semester 6: advanced_game_development
```

```
solve lagi? y/n >> y
```

```
nama file (tanpa txt): kelasMIT
```

```
Semester 1: 60001 Introduction to Computer Science Programming in Python
```

```
Semester 2: 6034 Artificial Intelligence
```

```
Semester 3: 6009 Fundamentals of Programming
```

```
Semester 4: 6833 The Human Intelligence Enterprise
```

```
Semester 5: 6006 Introduction to Algorithms
```

```
Semester 6: 6836 Multicore Programming
```

```
solve lagi? y/n >>
```

kelasHarvard.txt - Notepad

File Edit Format View Help

algorithms, basic_programming.

data_structures, basic_programming.

introduction_to_game_development, data_structures, algorithms.

advanced_game_development, introduction_to_game_development, object_oriented_

basic_programming.

object_oriented_programming, data_structures, algorithms.

kelasMIT.txt - Notepad

File Edit Format View Help

6034 Artificial Intelligence, 60001 Introduction to Computer Science Programming

6006 Introduction to Algorithms, 6009 Fundamentals of Programming.

6009 Fundamentals of Programming, 60001 Introduction to Computer Science Program

60001 Introduction to Computer Science Programming in Python.

6833 The Human Intelligence Enterprise, 6034 Artificial Intelligence.

6836 Multicore Programming, 6006 Introduction to Algorithms.

<

Ln 3, Col 96

100%

Windows (CRLI

Poin	Ya	Tidak
1. Program berhasil dikompilasi	✓	
2. Program berhasil <i>running</i>	✓	
3. Program dapat menerima berkas input dan menuliskan output.	✓	
4. Luaran sudah benar untuk semua kasus input.	✓	