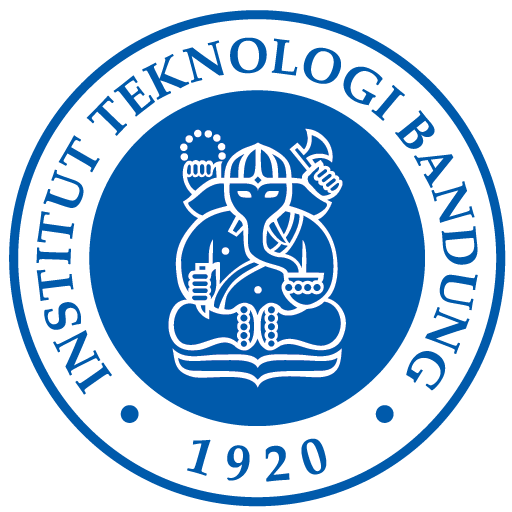
**TUGAS KECIL 2** **IF2211** **STRATEGI ALGORITMA**

**PENYELESAIAN PENYUSUNAN RENCANA KULIAH DENGAN TOPOLOGICAL SORT (PENERAPAN DECREASE AND CONQUER)**

****

**Disusun oleh:**

**Nama: Muhammad Furqon**

**NIM: 13519184**

**PROGRAM STUDI TEKNIK INFORMATIKA**

**SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA**

**INSTITUT TEKNOLOGI BANDUNG**

**2021**

**Daftar Isi**

[**1.** **Algoritma Topological Sort** 3](#_Toc65423260)

[**2.** **SourceCode(Python)** 4](#_Toc65423261)

[**3.** **Tangkapan Layar Input Output** 7](#_Toc65423262)

[**4.** **Tautan** 11](#_Toc65423263)

# **Algoritma Topological Sort**

Algoritma Topological Sort dengan pendekatan decrease and conquer

1. Bentuk graf dengan bentuk adjacency list antar simpul. Buat daftar atau list simpul yang merupakan mata kuliah yang belum diambil.
2. Proses simpul dalam daftar simpul yang belum diproses, cek apakah simpul memiliki jumlah *prerequisite*(derajat masuk)sama dengan nol, yaitu simpul tidak memiliki simpul lain yang menunjuk simpul tersebut. Jika memenuhi syarat, pilih simpul-simpul tersebut.
3. Simpul yang telah dipilih akan dihapus sisi-sisinya. Kurangi derajat masuk dari simpul-simpul yang ditunjuk simpul terpilih tersebut sebanyak satu.
4. Simpul yang terpilih dihilangkan dari daftar atau list simpul mata kuliah yang belum diambil. List simpul yang belum diproses adalah list simpul setelah proses ini.
5. Ulangi langkah kedua sampai langkah keempat hingga list simpul mata kuliah yang belum diambil sudah kosong atau telah mencapai delapan semester.

Pendekatan algoritma Topological sort ini adalah decrease and conquer dengan decrease sebanyak constant. Graf yang telah dibuat pada setiap iterasinya akan dikurangi menjadi sebuah upagraf yang lebih kecil. Jika n adalah banyak elemen pada graf dan s adalah simpul yang terpilih maka graf akan berkurang menjadi upagraf sebanyak n-s. Setelah pengurangan iterasi berikutnya akan diproses lagi sampai upagraf kosong atau semester telah mencapai semester 8.

Contoh:

Terdapat mata kuliah dengan nama mata kuliah dan *prerequisite*.

IF1,IF2.

IF2,IF3,IF4.

IF3.

IF4,IF3.

IF5,IF2,IF3,IF4.

Loop dalam pengerjaan disimbolkan dengan semester

IF1 : 1  
IF2 : 2  
IF3 : 0  
IF4 : 1  
IF5 : 3

Semester1: Pilih simpul-simpul dengan *prerequisite* sebanyak nol. IF3 dipilih, IF2,IF4, dan IF5 berkurang *prerequisite* sebanyak 1.

Semester2: Pilih simpul-simpul dengan *prerequisite* sebanyak nol. IF4 dipilih, IF2 dan IF5 berkurang *prerequisite* sebanyak 1.

Semester3: Pilih simpul-simpul dengan *prerequisite* sebanyak nol. IF2 dipilih, IF1 dan IF5 berkurang *prerequisite* sebanyak 1.

Semester4: Pilih simpul-simpul dengan *prerequisite* sebanyak nol. IF1,IF5 dipilih.

Program berakhir karena semua mata kuliah telah diambil.

# **SourceCode(Python)**

Program MatkulDecider terbagi ke dalam beberapa bagian. Dalam penerapannya digunakan kelas Simpul dan representasi graf dalam bentuk adjacency list dalam dictionary. Fungsi terdiri dari inisialisasiFile untuk file, buatGraph untuk membuat graf, topologicalSearch untuk melakukan topological sort pada simpul, dan kurangiPrereq untuk menghapus simpul dan sisinya.

MatkulDecider.py

#Muhammad Furqon (13519184)

#Decrease and conquer (Topological Sort)

#13519184.py

#MatkulDecider

import os

#class untuk simpul terdiri dari kode dan prereq

class Simpul:

    #prereq:derajat masuk ; kode:kode matkul

    def \_\_init\_\_(self,kode,prereq):

        self.kode=kode

        self.prereq=int(prereq)

#Dari list kurangi graph yang sudah diproses

def kurangiPrereq(List, Simpul, Graph):

    sisi=Graph[Simpul.kode]

    #hilangkan sisi ke simpul lain dan kurangi prereq

    for x in sisi:

        for elemen in List:

            if(elemen.kode==x):

                elemen.prereq-=1

    #hilangkan dari dictionary

    Graph.pop(Simpul.kode)

#Print isi list objek, debugging purpose

def printListObj(List):

    first=True

    for obj in List:

        print("Kode matkul: "+ obj.kode)

        print("Jumlah prereq: "+ str(obj.prereq))

        first=False

    print("-"\*10)

#Menggunakan topological sort

def topologicalSort(ListObj,ListResult,Graph):

    for matkul in ListObj:

        #mengecek adakah matkul yang prereq 0

        if matkul.prereq==0:

            #list hasil dari satu loop

            indeks=ListObj.index(matkul)

            ListResult.append(ListObj[indeks])

    for matkul\_hasil in ListResult:

        #panggil fungsi mengurangi prereq yang membutuhkan matkul tersebut

        kurangiPrereq(ListObj,matkul\_hasil,Graph)

        #buang dari list objek

        indeks=ListObj.index(matkul\_hasil)

        ListObj.pop(indeks)

def inisialisasiFile(list1,dictionary):

    list\_main=[]

    # List setiap barisnya

    os.chdir("..")

    cur\_dir=os.getcwd()

    os.chdir(cur\_dir+"\\test")

    nama\_file = input("Masukkan nama file masukan: ")

    cur\_dir=os.getcwd()

    path=cur\_dir+ "\\" + nama\_file

    file = open(path,"r")

    #membaca dari file lalu mengubah menjadi bentuk array

    for line in file:

        line=line.strip()

        line=line.replace(" ","")

        line=line.strip(".")

        line.split(",")

        list\_main.append(line)

    for x in list\_main:

        x= list(x.split(","))

        #bagian head(depan) nama, tail berisi prereq

        head=x[0]

        tail=x[1:]

        temp\_obj = Simpul(head,len(tail))

        list1.append(temp\_obj)

        #dimasukkan ke dalam dictionary temporary yang akan digunakan untuk graph

        dictionary.update({head:tail})

def buatGraph(dictionary):

    #format key=simpul lalu value=sisi

    #contoh {'IF1': [], 'IF2': ['IF1', 'IF5'], 'IF4': ['IF2', 'IF5'], 'IF5': []}

    temp\_dict={}

    for key in dictionary:

        temp\_arr=[]

        for key2 in dictionary:

            temp=dictionary[key2]

            if(key in temp):

                temp\_arr.append(key2)

        temp\_dict[key]=temp\_arr

    return temp\_dict

#MAIN PROGRAM

#list objek Simpul

list\_objek=[]

#list menyimpan hasil pada semester tersebut

list\_hasil=[]

#list hasil total, debugging purpose

list\_hasil\_all=[]

#list nama semester

list\_semester=["I","II","III","IV","V","VI","VII","VIII"]

#temporary dictionary untuk membuat graph

dict\_awal={}

#graph yang digunakan dalam memilih matkul

graph={}

#inisialisasi dari file

inisialisasiFile(list\_objek,dict\_awal)

#buat graph

graph = buatGraph(dict\_awal)

semester=0

while(len(list\_objek)!=0 and semester<=7):

    #Melakukan topological sort

    topologicalSort(list\_objek,list\_hasil,graph)

    print("Semester "+ list\_semester[semester]+" :",end=" ")

    first=True

    for matkul\_diambil in list\_hasil:

        if(first):

            print(matkul\_diambil.kode,end="")

            first=False

        else:

            print(", "+matkul\_diambil.kode,end="")

    print()

    list\_hasil\_all.append(list\_hasil)

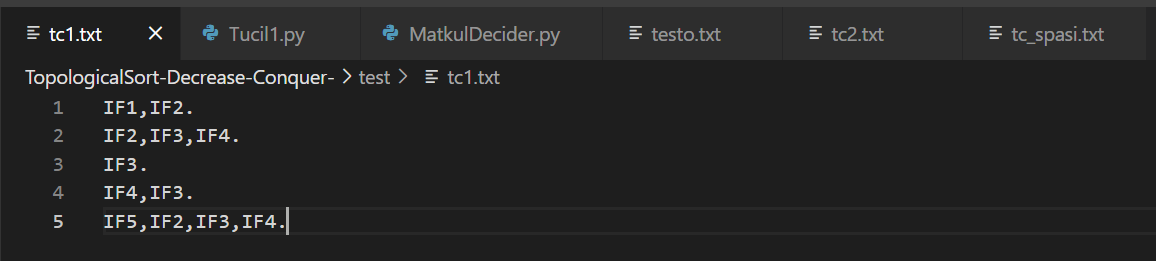
    list\_hasil=[]

    semester+=1

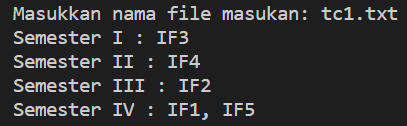
input("Tekan enter untuk mengakhiri program")   

# **Tangkapan Layar Input Output**

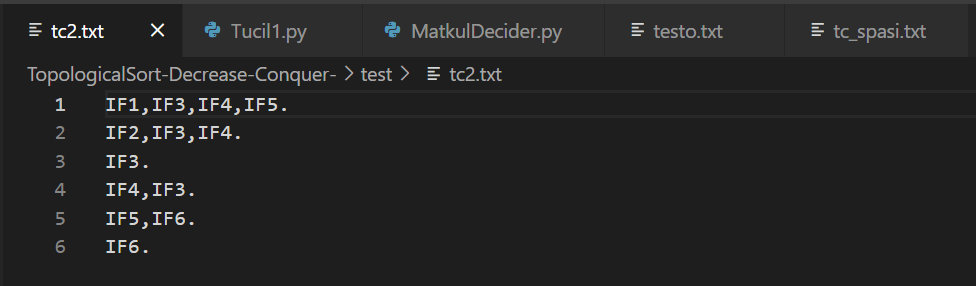
* 1. tc1.txt

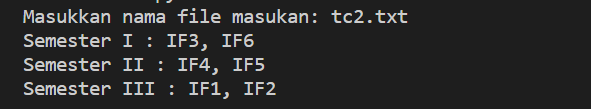


Gambar

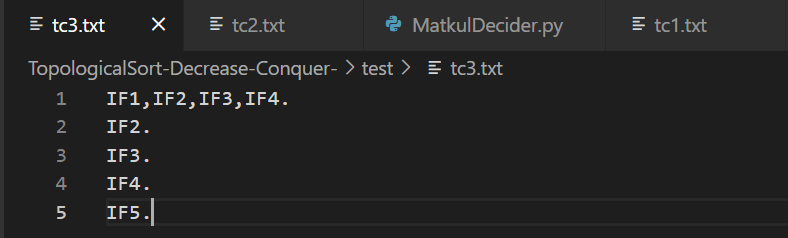


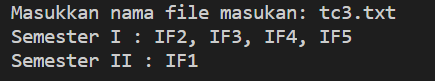
* 1. tc2.txt



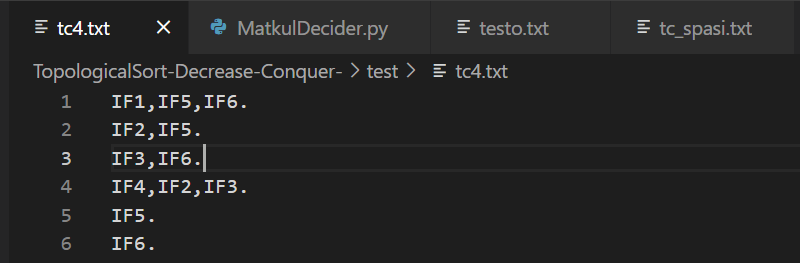


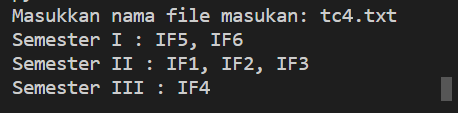
* 1. tc3.txt



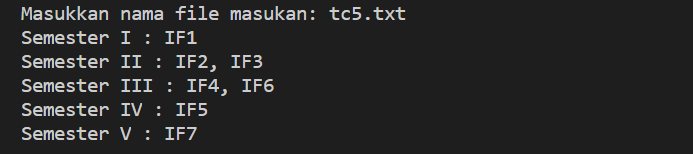
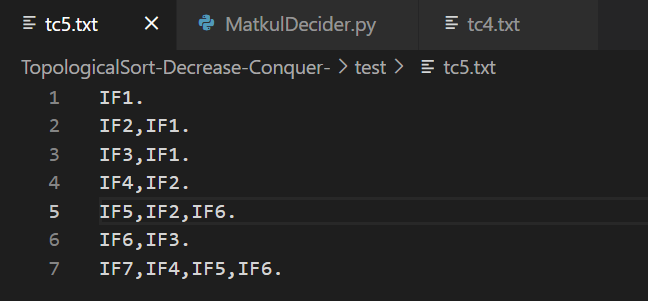


* 1. tc4.txt

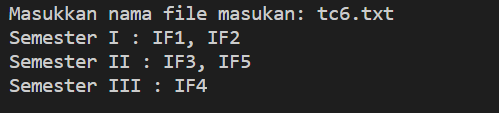
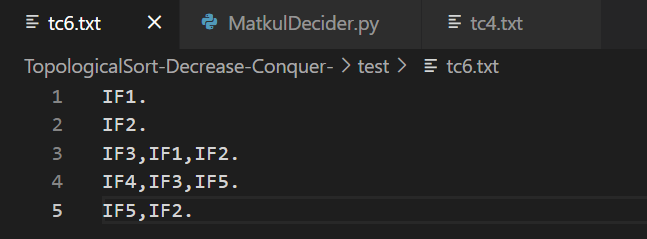




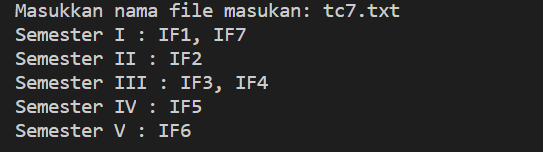
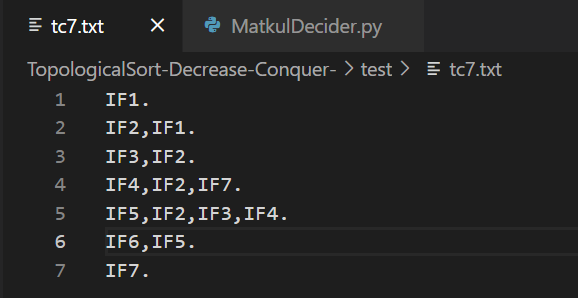
* 1. tc5.txt



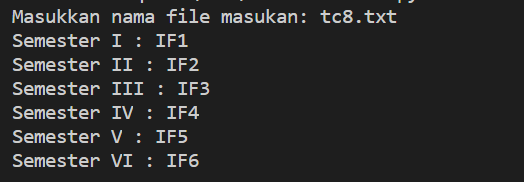
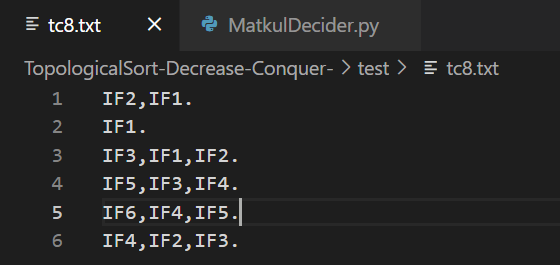
* 1. tc6.txt



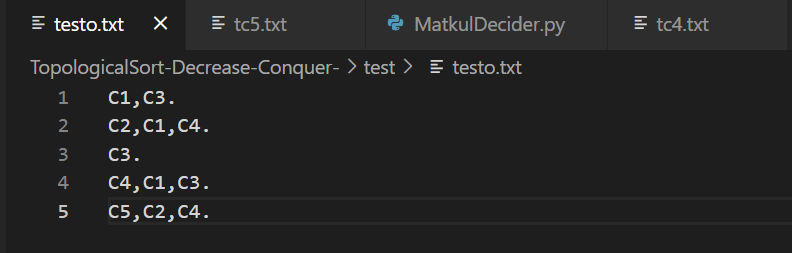
* 1. tc7.txt

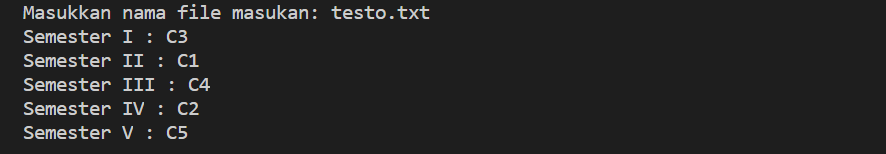


* 1. tc8.txt

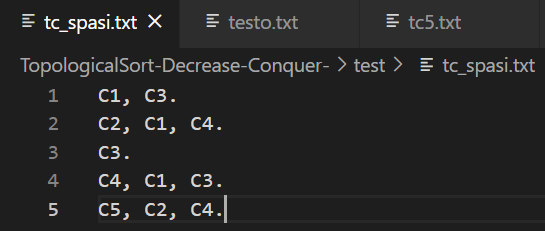


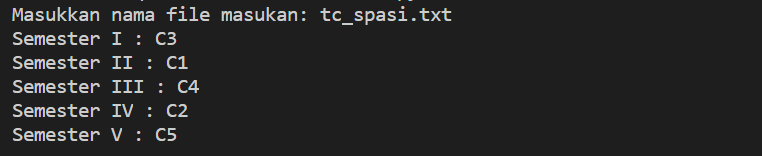
* 1. testo.txt





* 1. tc\_spasi.txt





# **Tautan**

Alamat kode, github repo

<https://github.com/Purecon/Tucil2_13519184>

Link Pengumpulan

<http://bit.ly/KumpulTucil2Stima>

|  |  |  |
| --- | --- | --- |
| Poin | Ya | Tidak |
| 1. Program berhasil dikompilasi | ✓ |  |
| 2. Program berhasil *running* | ✓ |  |
| 3. Program dapat menerima berkas input dan menuliskan output. | ✓ |  |
| 4. Luaran sudah benar untuk semua kasus input. | ✓ |  |