Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung

Laporan Tugas Kecil 2 IF2211 Strategi Algoritma

Semester II Tahun 2020/2021

Penyelesaian *Topological Sort* dengan Algoritma *Decrease and Conquer*

Nama : Daru Bagus Dananjaya

NIM : 13519080

Kelas : K02

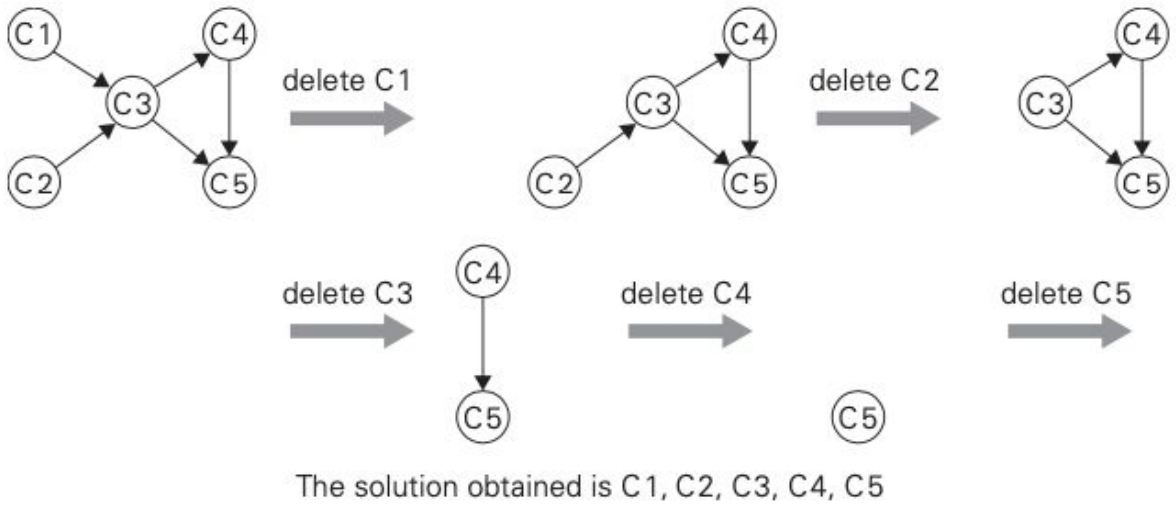
Penjelasan Topological Sort

Pada tugas kali ini, mahasiswa diminta membuat aplikasi sederhana yang dapat menyusun rencana pengambilan kuliah, dengan memanfaatkan algoritma Decrease and Conquer. Penyusunan Rencana Kuliah diimplementasikan dengan menggunakan pendekatan Topological Sorting. Aplikasi akan menerima daftar mata kuliah beserta prasyarat yang harus diambil seorang mahasiswa sebelum mengambil mata kuliah tersebut. Sebuah kode_kuliah mungkin memiliki nol atau lebih prasyarat kuliah. Kode_kuliah bisa diambil pada suatu semester jika semua prasyaratnya sudah pernah diambil di semester sebelumnya (tidak harus 1 semester sebelumnya).

Persoalan perencanaan pengambilan mata kuliah di atas dapat direpresentasikan ke dalam sebuah graf berarah dengan kode_kuliah *prerequisite* sebagai *source node* dan kode_kuliah lainnya menjadi *destination node*. Untuk dapat memiliki solusi, persoalan harus berupa *directed acyclic graph* (DAG). Terdapat dua buah pendekatan solusi untuk menyelesaikan masalah pengambilan mata kuliah dengan metode topological sort, yaitu dengan DFS Traversal dan implementasi langsung dari *decrease-(by one)-and-conquer technique*. Namun pada Tugas ini, persoalan ini hanya akan diselesaikan menggunakan implementasi dari *decrease-and-conquer*.

Pada algoritma *topological sort* dengan menggunakan pendekatan *decrease-and-conquer*, proses yang dilakukan adalah mengidentifikasi *source vertex* dari DAG, yaitu *vertex* yang tidak

memiliki *incoming edges*, kemudian menghapusnya dari graf beserta seluruh *edge* yang keluar dari *vertex* tersebut kemudian *vertex* tersebut dimasukkan ke dalam himpunan solusi. Jika terdapat beberapa *source vertex* dalam graf, maka lakukan penghapusan *vertex* satu per satu sambil dimasukkan ke dalam himpunan solusi. Namun, jika tidak terdapat *source vertex*, maka proses sorting dihentikan karena berarti permasalahan tersebut tidak dapat diselesaikan.



Gambar 1. Ilustrasi penyelesaian masalah topological sorting menggunakan pendekatan *decrease-and-conquer*.

(*Sumber : Introduction to the Design and Analysis of Algorithms, 3rd Ed. - Levitin*)

Deskripsi Langkah-Langkah

Dalam menyelesaikan persoalan *topological sort* untuk *course scheduling* menggunakan metode *decrease-and-conquer*, saya memanfaatkan *data structure* berupa array. Array digunakan untuk menyimpan *course* beserta mata kuliah *prerequisite* nya, serta untuk menyimpan *course* yang dapat diselenggarakan di setiap semesternya.

Langkah-langkah program :

1. Membaca file persoalan *topological sort* yang akan diselesaikan, lalu dilakukan parsing pada setiap *course* untuk dimasukkan ke dalam array *listOfCourse*.
2. Selama list *listOfCourse* masih belum kosong, dilakukan looping untuk penghapusan *course* yang tidak memiliki prerequisite.

a. Mencari *course* yang tidak memiliki prerequisite kemudian memasukkannya ke dalam array *noPrereq*.
b. Lakukan penghapusan *course* terkait di dalam list *course* lainnya apabila *course* tersebut menjadi prerequisite dari *course* lainnya.
c. Lakukan penghapusan *course* terkait dari array *listOfCourse.*
d. Tambahkan array *noPrereq* ke dalam array *courseSemester*. index dari array *courseSemester* merepresentasikan pada semester berapa *course* tersebut diambil.

Pada persoalan *topological sort* yang diselesaikan dengan menggunakan pendekatan *decrease-and-conquer*, kompleksitas waktunya adalah *O(V+E)* dengan *V* adalah jumlah vertex dan *E* adalah jumlah edges. Penghapusan setiap node yang menjadi *source vertex* menghasilkan *V* dan ketika kita melakukan penghapusan sisi yang berasal dari *source vertex*, dilakukan iterasi ke seluruh *edges* yang ada di graf menghasilkan *E*. Oleh karena itu, diperoleh Kompleksitas waktu *O(V+E)*

Source Code

```
import roman as bm

def readFile(testcase):
    # I.S -
    # F.S Mengembalikan listOfCourse sesuai dengan bentuk yang diinginkan
    listOfCourse = []
    line = inString.readlines()

    i = 0
    while (i < len(line)):
        cleanString = line[i].replace(" ","")
        cleanString = cleanString.replace(".","")
        cleanString = cleanString.replace("\n","")
        listOfCourse.append(cleanString)
        i+=1

    for i in range(len(listOfCourse)):
        listOfCourse[i] = listOfCourse[i].split(',')
```

```
    # Bentuk List Of Course : [["C1","C3"], ["C2","C1","C4"], ["C3"], ["C4","C1","C3"], ["C5","C2","C4"]]
    return listOfCourse

def printPersoalan(testcase) :

    line = inString.readlines()
    print("Problem :")
    i = 0
    while (i < len(line)):
        cleanString = line[i].replace("\n","")
        cleanString = cleanString.replace(".","")
        cleanString = cleanString.split(',')
        if (len(cleanString)==1):
            print("Course "+cleanString[0]+" has no prerequisite.")
        else :
            print("Course "+cleanString[0]+"'s prerequisite(s) are ", end="")
            for j in range(1,len(cleanString)):
                if (j != len(cleanString)-1) :
                    print(" "+cleanString[j]+",", end='')
                else :
                    print(" "+cleanString[j])
        i+=1

def topologicalSKRT(listOfCourse):
    courseSemester = []

    # Selama masih ada course di listOfCourse, maka loop penghapusan akan berjalan
    while (len(listOfCourse) != 0) :
        noPrereq = []

        # Mencari course yang tidak memiliki prerequisite
        for i in range(len(listOfCourse)):
            '''Jika course tidak memiliki prerequisite, maka tambahkan ke list course
            yang tidak memiliki prerequisite'''
            if (len(listOfCourse[i]) == 1):
                noPrereq.append(listOfCourse[i][0])

        # Menghapus course yang tidak memiliki prerequisite dari seluruh listOfCourse (yang ada di dalam array course lain)
        for i in range(len(noPrereq)):
            for j in range(len(listOfCourse)):
```

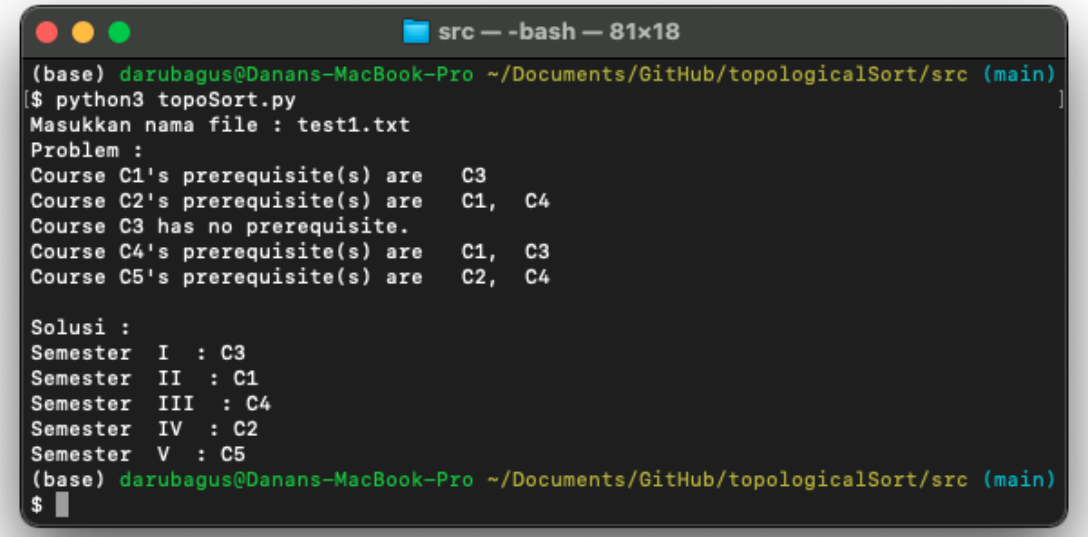
```
            '''Kalo course yang gapunya prerequisite ada di dalam list prerequisite
course lain, maka dia dihapus
            tapi kalo gaada, yaudah lanjut aja'''
            if (noPrereq[i] in listOfCourse[j]) :
                k = 0
                status = False
                while ((not status) and k < len(listOfCourse)):
                    if (noPrereq[i] == listOfCourse[j][k]) :
                        # hapus course yang tidak memiliki prerequisite dari list
prerequisite course lain
                        del listOfCourse[j][k]
                        status = True
                    else :
                        k+=1

        # Menghapus course yang tidak memiliki prerequisite dari listOfCourse
        i = 0
        while (i < len(listOfCourse)):
            # Kalo dia gapunya prerequisite, dia pasti udah diapus di langkah
sebelumnya, makanya disini kosong terus diapus deh
            if (len(listOfCourse[i]) == 0) :
                del listOfCourse[i]
            else :
                i+=1

        # Masukkan seluruh course yang tidak memiliki prerequisite ke dalam array
courseSemester
        courseSemester.append(noPrereq)

    printSemester(courseSemester)


def printSemester(courseSemester) :
    #Mencetak data course per semester
    for i in range(len(courseSemester)) :
        print("Semester ", bm.toRoman(i+1), " :", end='')
        # Mencetak data course yang ada di semester yang bersangkutan
        for j in range(len(courseSemester[i])):
            if (j != len(courseSemester[i])-1) :
                print(" "+courseSemester[i][j]+",", end='')
            else :
                print(" "+courseSemester[i][j])
```

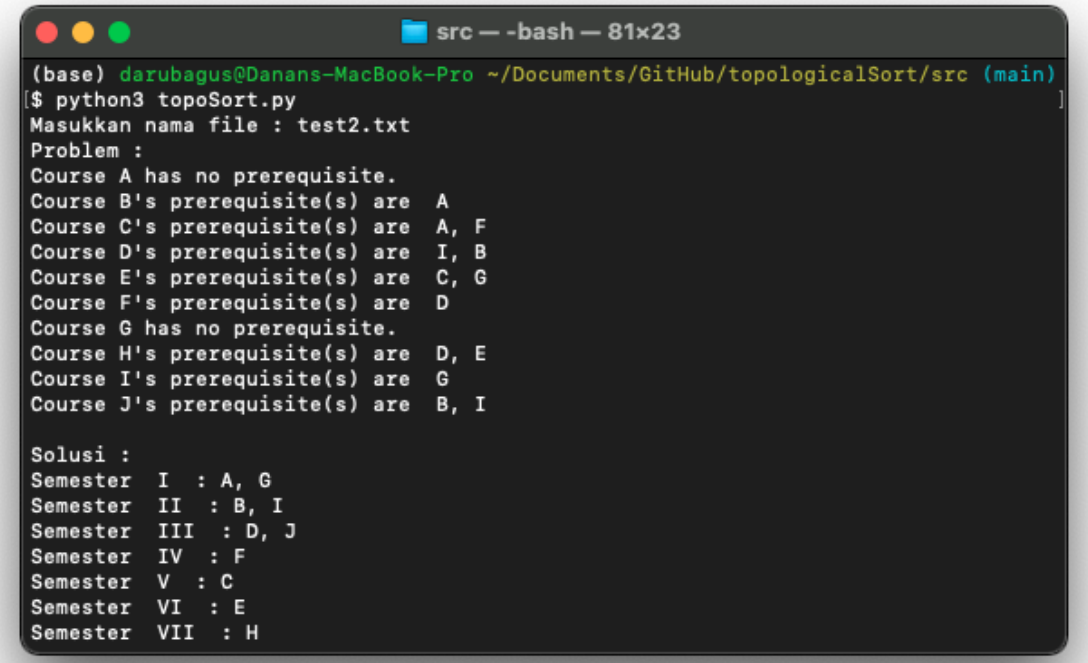
```
# Main Program
fileName = input("Masukkan nama file : ")
testCase = "../test/"+fileName

inString = open(testCase, "r")
printPersoalan(inString)
inString.close()

inString = open(testCase, "r")
listOfCourse = readFile(inString)
inString.close()
print()
print("Solusi :")
topologicalSKRT(listOfCourse)
```

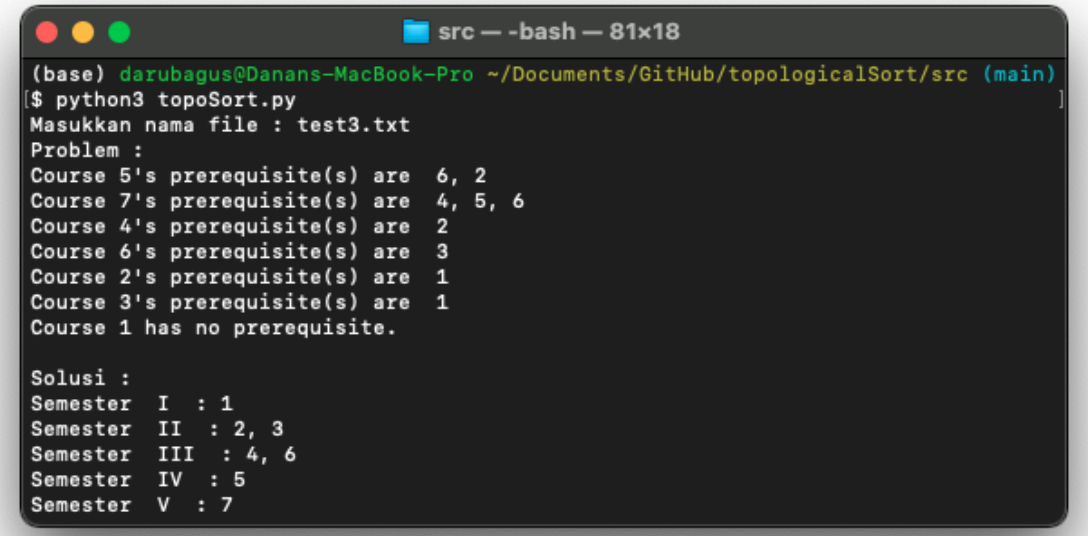
Link File

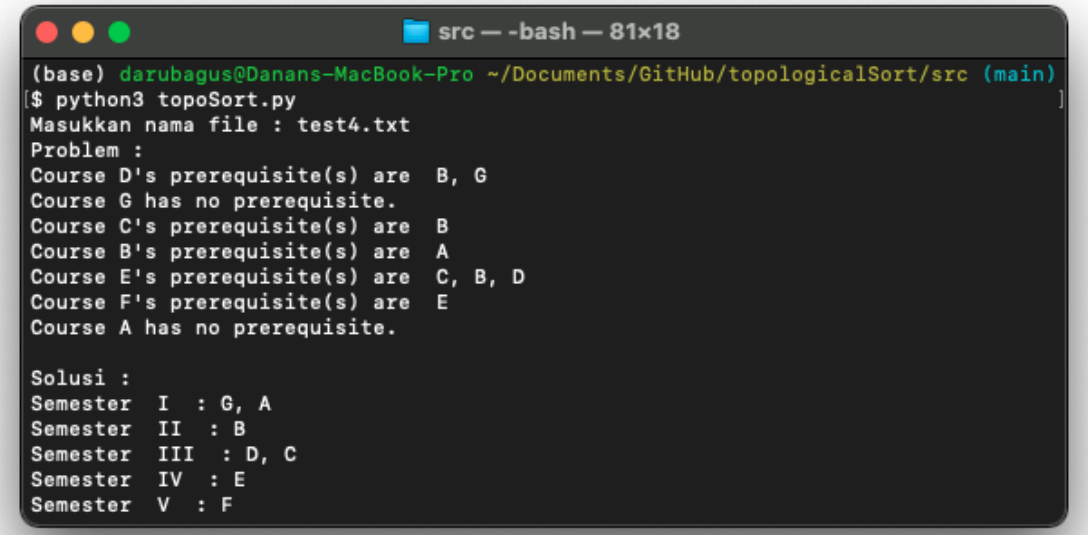
1. Github
   https://github.com/darubagus/topologicalSort

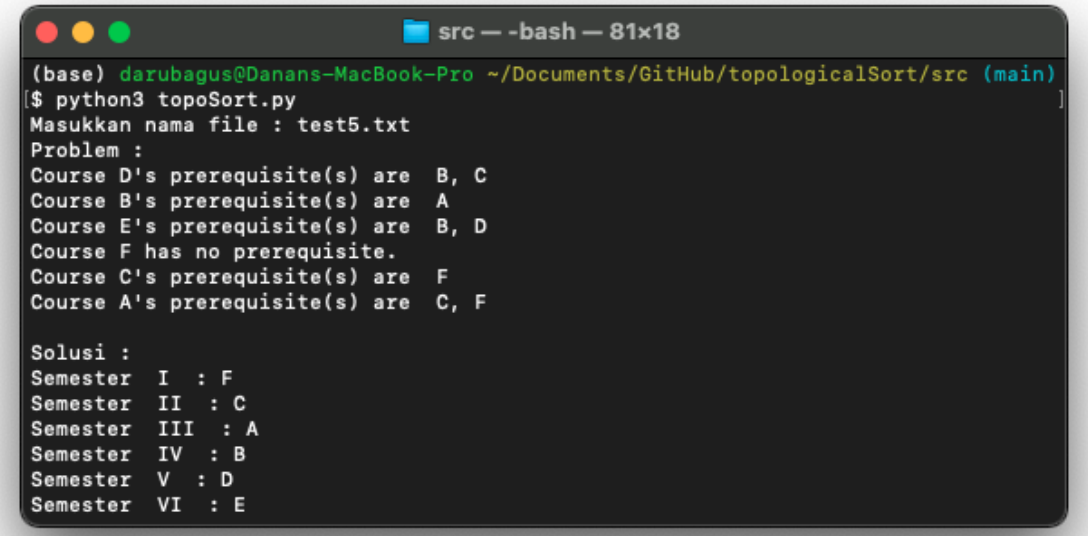
Hasil Screenshot

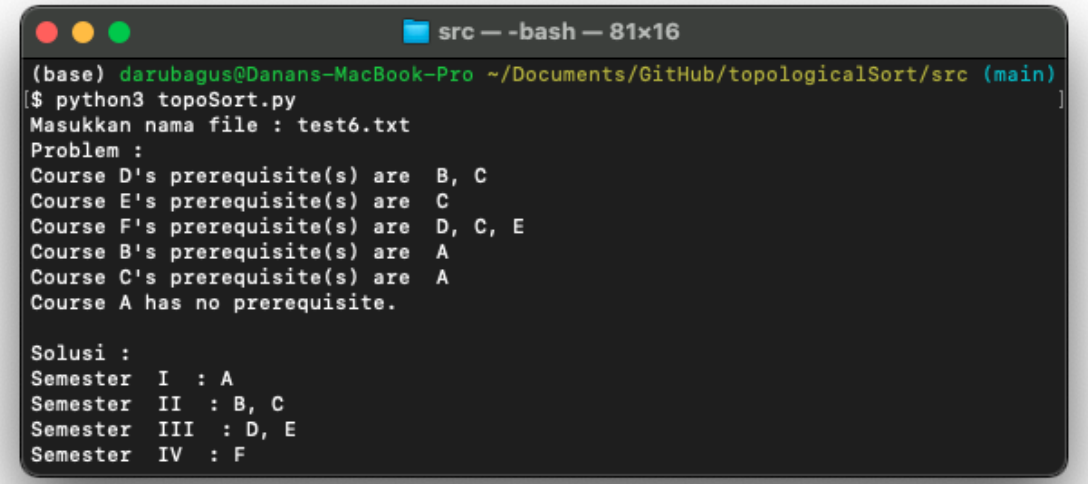


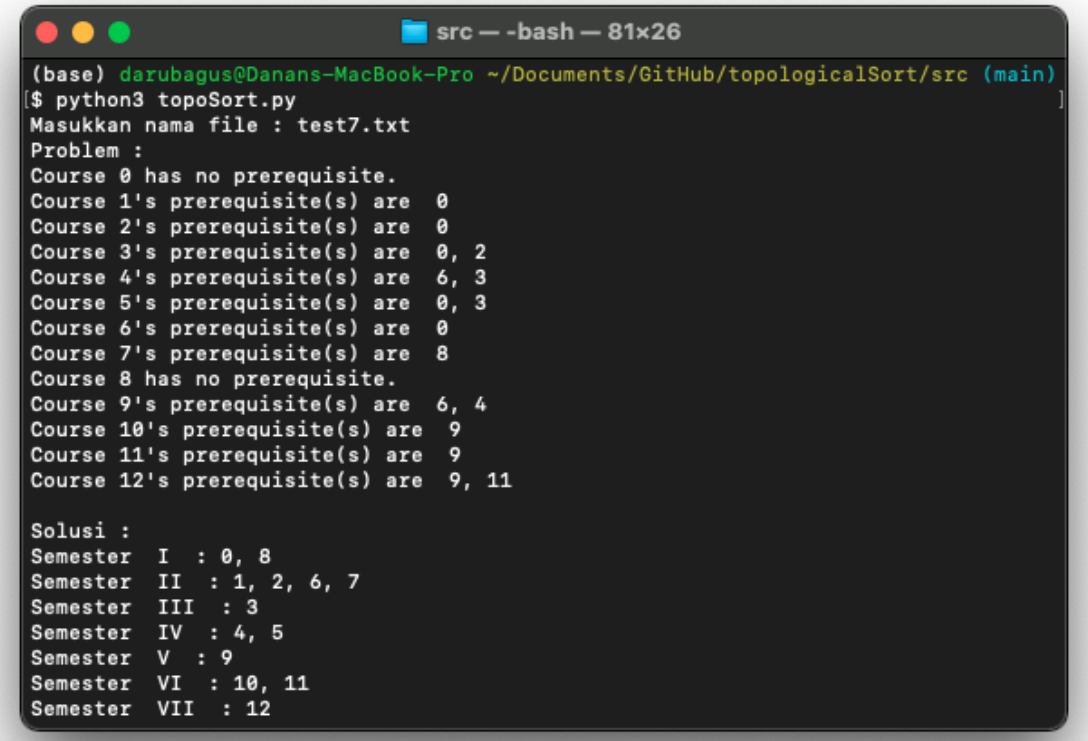


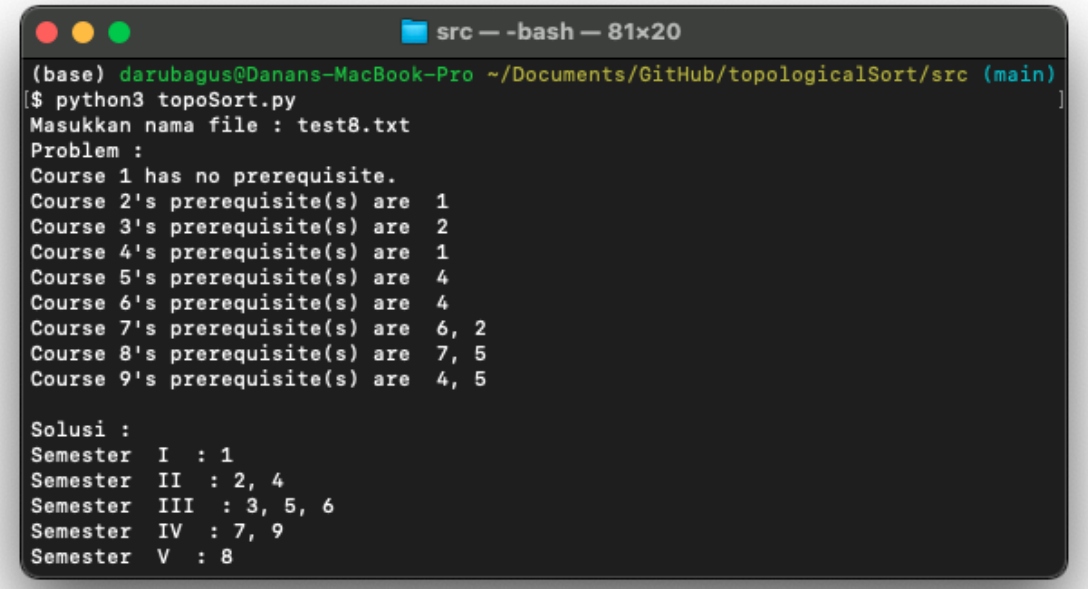

src — -bash — 81×18
(base) darubagus@Danans-MacBook-Pro ~/Documents/GitHub/topologicalSort/src (main)
$ python3 topoSort.py
Masukkan nama file : test3.txt
Problem :
Course 5's prerequisite(s) are 6, 2
Course 7's prerequisite(s) are 4, 5, 6
Course 4's prerequisite(s) are 2
Course 6's prerequisite(s) are 3
Course 2's prerequisite(s) are 1
Course 3's prerequisite(s) are 1
Course 1 has no prerequisite.
Solusi :
Semester I : 1
Semester II : 2, 3
Semester III : 4, 6
Semester IV : 5
Semester V : 7


src — -bash — 81×18
(base) darubagus@Danans-MacBook-Pro ~/Documents/GitHub/topologicalSort/src (main)
$ python3 topoSort.py
Masukkan nama file : test4.txt
Problem :
Course D's prerequisite(s) are B, G
Course G has no prerequisite.
Course C's prerequisite(s) are B
Course B's prerequisite(s) are A
Course E's prerequisite(s) are C, B, D
Course F's prerequisite(s) are E
Course A has no prerequisite.
Solusi :
Semester I : G, A
Semester II : B
Semester III : D, C
Semester IV : E
Semester V : F


src — -bash — 81×18
(base) darubagus@Danans-MacBook-Pro ~/Documents/GitHub/topologicalSort/src (main)
$ python3 topoSort.py
Masukkan nama file : test5.txt
Problem :
Course D's prerequisite(s) are B, C
Course B's prerequisite(s) are A
Course E's prerequisite(s) are B, D
Course F has no prerequisite.
Course C's prerequisite(s) are F
Course A's prerequisite(s) are C, F
Solusi :
Semester I : F
Semester II : C
Semester III : A
Semester IV : B
Semester V : D
Semester VI : E


src — -bash — 81×16
(base) darubagus@Danans-MacBook-Pro ~/Documents/GitHub/topologicalSort/src (main)
$ python3 topoSort.py
Masukkan nama file : test6.txt
Problem :
Course D's prerequisite(s) are B, C
Course E's prerequisite(s) are C
Course F's prerequisite(s) are D, C, E
Course B's prerequisite(s) are A
Course C's prerequisite(s) are A
Course A has no prerequisite.
Solusi :
Semester I : A
Semester II : B, C
Semester III : D, E
Semester IV : F


src — -bash — 81×26
(base) darubagus@Danans-MacBook-Pro ~/Documents/GitHub/topologicalSort/src (main)
$ python3 topoSort.py
Masukkan nama file : test7.txt
Problem :
Course 0 has no prerequisite.
Course 1's prerequisite(s) are 0
Course 2's prerequisite(s) are 0
Course 3's prerequisite(s) are 0, 2
Course 4's prerequisite(s) are 6, 3
Course 5's prerequisite(s) are 0, 3
Course 6's prerequisite(s) are 0
Course 7's prerequisite(s) are 8
Course 8 has no prerequisite.
Course 9's prerequisite(s) are 6, 4
Course 10's prerequisite(s) are 9
Course 11's prerequisite(s) are 9
Course 12's prerequisite(s) are 9, 11
Solusi :
Semester I : 0, 8
Semester II : 1, 2, 6, 7
Semester III : 3
Semester IV : 4, 5
Semester V : 9
Semester VI : 10, 11
Semester VII : 12


src — -bash — 81×20
(base) darubagus@Danans-MacBook-Pro ~/Documents/GitHub/topologicalSort/src (main)
$ python3 topoSort.py
Masukkan nama file : test8.txt
Problem :
Course 1 has no prerequisite.
Course 2's prerequisite(s) are 1
Course 3's prerequisite(s) are 2
Course 4's prerequisite(s) are 1
Course 5's prerequisite(s) are 4
Course 6's prerequisite(s) are 4
Course 7's prerequisite(s) are 6, 2
Course 8's prerequisite(s) are 7, 5
Course 9's prerequisite(s) are 4, 5
Solusi :
Semester I : 1
Semester II : 2, 4
Semester III : 3, 5, 6
Semester IV : 7, 9
Semester V : 8

Checklist

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	✔	