LAPORAN TUGAS KECIL 2

STRATEGI ALGORITMA - IF2211

"Penyusunan Rencana Kuliah dengan *Topological Sort*

(Penerapan *Decrease and Conquer)*"

Dibuat Oleh :

Mohammad Sheva Almeyda Sofjan - 13519018

K-01

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG

2021

DESKRIPSI PERSOALAN

Akan dibuat aplikasi sederhana yang mampu menyusun rencana pengambilan mata kuliah per semester menggunakan algoritma *Topological Sort*, penerapan *Decrease and Conquer*.

Mula-mula aplikasi akan menerima masukan/*input* berupa daftar kode kuliah beserta kode mata kuliah prasyaratnya dalam bentuk file teks, dengan format berikut :

```
<kode_kuliah_1>,<kode  kuliah prasyarat -1>,  <kode  kuliah  prasyarat -
2>,  <kode  kuliah prasyarat -3>.
<kode_kuliah_2>,<kode kuliah prasyarat -1>, <kode kuliah prasyarat -2>.
<kode_kuliah_3>,<kode  kuliah  prasyarat -1>,  <kode  kuliah  prasyarat
-2>,  <kode  kuliah prasyarat -3>, <kode kuliah prasyarat -4>.
<kode_kuliah_4>.
```

Sebuah mata kuliah mungkin memiliki nol atau lebih prasyarat kuliah. Mata Kuliah bisa diambil pada suatu semester jika semua prasyaratnya sudah pernah diambil di semester sebelumnya (tidak harus 1 semester sebelumnya). Asumsi semua mata kuliah bisa diambil di sembarang semester, baik semester ganjil maupun semester genap. Tidak ada batasan jumlah mata kuliah yang dapat di ambil per semester nya.

Sebagai contoh, terdapat file masukan berikut :

```
c1.
c2, c1, c4.
c3.
c4, c1, c3.
c5, c2, c4.
```

Lalu aplikasi akan memproses file di atas dan mengeluarkan keluaran/*output* berikut :

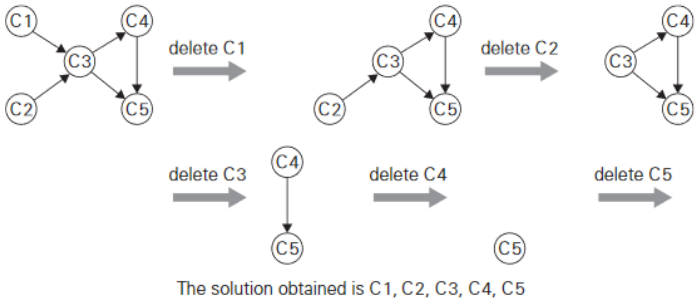
```
Semester 1 : c1, c3
Semester 2 : c4
Semester 3 : c2
Semester 4 : c5
```

ALGORITMA TOPOLOGICAL SORT

Topological Sort merupakan algoritma untuk menentukan urutan linier semua simpul dari sebuah graf asiklik berarah (*directed acyclic graph / DAG)*, di mana tiap sisi *uv* (terdapat busur dari *u* ke *v*), simpul *u* berada sebelum *v* pada hasil pengurutan.

Pada persoalan ini, simpul *u* dapat dianalogikan sebagai mata kuliah prasyarat dari mata kuliah yang dilambangkan dengan simpul *v*, sehingga pada pengurutan nantinya, mata kuliah *u* diambil pada semester sebelum semester diambilnya mata kuliah *v*.

Aplikasi yang dibuat memanfaatkan algoritma *topological sort* varian *source-removal algorithm*, yaitu pada tiap iterasi, cari simpul dengan derajat masuk 0, masukkan ke *list* hasil pengurutan, lalu hapus simpul tersebut beserta busur yang keluar dari simpul tersebut.



Ilustrasi topological sort : source-removal algorithm[1]

Hubungannya dengan *decrease and conquer* (varian *decrease by a constant*) adalah pada bagian pengulangan operasi pemilihan dan penghapusan simpul dengan derajat masuk 0 sehingga banyaknya simpul yang harus diurus semakin berkurang (tereduksi) pada tiap iterasinya hingga habis dan *list* hasil pengurutan mengandung semua simpul. Sebagai contoh, pada ilustrasi di atas mula-mula persoalan berukuran 5 (banyak simpul), lalu pada tiap iterasinya banyak simpul berkurang 1, menyebabkan sub-persoalan yang diurus semakin berkurang sebesar 1.

Gambaran langkah-langkah algoritma dalam menemukan solusi dari input persoalan pada aplikasi ini adalah sebagai berikut :

1. Program menjalankan file main, dan mulai mengeksekusi fungsi *reader()* yang terdapat pada file util untuk membaca input program, melakukan validasi terhadap input dan mengembalikan 2 *list* yaitu *nested list* yang berisi input pengguna (*list* isi) dan *list* yang berisi kode kuliah unik pada input (*list* codelist). *List* codelist berkorespondensi dengan indeks suatu simpul pada list/matriks yang akan dibuat berikutnya.

2. Membuat *adjacency matrix* (matriks ketetanggaan) dari graf pada input. Sebut matriks itu sebagai matriks *adjmat*. Jika adjmat[i][j] dengan i dan j merupakan indeks baris dan baris kolom dari adjmat, maka adjmat[i][j] = 0 menandakan tidak terdapat busur dari simpul i ke j , dan apabila adjmat[i][j] = 1 terdapat busur dari simpul i ke j.

3. Membuat list yang berisi derajat masuk tiap simpul pada graf, sebut list itu sebagai *listIn.*

4. Melakukan *topological sorting* terhadap graf menggunakan parameter *adjmat* dan *listIn* yang sudah dibuat sebelumnya, dengan rincian :

   i. Inisialisasi 2 list kosong, *nested* list order (hasil pengurutan) dan list zeroIn (berisi simpul dengan derajat masuk nol)

   ii. Kasus awal, masukkan simpul dengan derajat masuk nol ke dalam list zeroIn

   iii. Selama terdapat simpul dengan derajat masuk nol, maka :
   a. Masukkan list zeroIn ke list order
   b. Iterasi pada list zeroIn :
      - Ambil simpul pertama (sebut *currnode*) pada list zeroIn
      - Buat list simpul-simpul berikutnya (sebut *nextnodes)*, dimana terdapat busur dari *currnode* ke simpul pada *nextnodes*
      - Iterasi pada *nextnodes* : Hapus busur dari *currnode* ke simpul-simpul pada *nextnodes*, jika hasil penghapusan membuat derajat masuk suatu simpul pada *nextnodes* menjadi nol, maka masukkan simpul tersebut ke list zeroIn

   c. Hapus simpul pertama pada list zeroIn, menandakan simpul tersebut sudah selesai diproses.

   iv. Jika setelah proses *i,ii,dan iii* masih terdapat simpul dengan derajat masuk bukan nol, maka graf tersebut bukan termasuk graf asiklik berarah (DAG), sehingga mengeluarkan keluaran berupa list kosong dan pernyataan bahwa graf bukan termasuk graf asiklik berarah.
   Jika derajat masuk semua simpul menjadi nol, maka graf tersebut merupakan graf asiklik berarah sehingga hasil *topological sort* menjadi valid, mengeluarkan keluaran berupa *nested* list order, yang berisi hasil pengurutan semua simpul pada graf.

5. Jika hasil *topological sort* valid, maka keluarkan keluaran berupa banyaknya mata kuliah, banyak semester yang dibutuhkan, dan kode kuliah dengan mencocokan hasil pengurutan pada list order dengan list codelist.

Sebagai contoh, apabila program diberikan masukan :

```
c1.
c2, c1, c4.
c3.
c4, c1, c3.
c5, c2, c4.
```

Maka program akan menjalankan hal-hal berikut :

1. Program membaca masukan, dan hasilnya :
   List isi = [['c1'], ['c2', 'c1', 'c4'], ['c3'], ['c4', 'c1', 'c3'], ['c5', 'c2', 'c4']]
   List codelist = ['c1', 'c2', 'c3', 'c4', 'c5']
2. Membuat *adjacency matrix* dari input masukan, hasilnya :
   Matriks adjmat :
   ```
   0 1 0 1 0
   0 0 0 0 1
   0 0 0 1 0
   0 1 0 0 1
   0 0 0 0 0
   ```
3. Membuat list berisi derajat masuk tiap simpul, hasilnya :
   List listIn : [0, 2, 0, 2, 2]
4. Lakukan *topological sorting* pada input, untuk setiap iterasi akan dihasilkan list order dan zeroIn sebagai berikut : (0 merupakan c1, 1 merupakan c2, dan seterusnya)
   i. order : []
      zeroIn : [0, 2]
   ii. order : [[0, 2]]
      zeroIn : [3]
   iii. order : [[0, 2], [3]]
      zeroIn :[1]
   iv. order: [[0, 2], [3], [1]]
      zeroIn : [4]
   v. order : [[0, 2], [3], [1], [4]]
      zeroIn : []

5. Keluarkan keluaran, sehingga seperti berikut :

```
Banyak Mata Kuliah : 5
Banyak Semester yang dibutuhkan : 4

Output :
Semester 1 : c1, c3
Semester 2 : c4
Semester 3 : c2
Semester 4 : c5
```

SOURCE CODE APLIKASI

Aplikasi dibuat menggunakan bahasa Python (Python 3). Berikut adalah *source code* dari aplikasi yang telah dibuat.

13519018_main.py

```
'''
Nama      : Mohammad Sheva Almeyda Sofjan
NIM/Kelas : 13519018/K01
Deskripsi : Tugas Kecil 2 IF2211 Strategi Algoritma
            Aplikasi penyusunan kuliah
            menggunakan Topological Sort,
            penerapan Decrease n Conquer
Deskripsi File : File Utama (Driver)
'''

import importlib # Library import module

#import sols
sols = importlib.import_module('13519018_sols') # Solusi
#import util
util = importlib.import_module('13519018_util') # Utility

from time import time # Library Waktu

print("\n=========================================================")
print("             <<Memorable MatKul Planner>>                ")
print("  Dapat menyusun rencana pengambilan mata kuliah anda    ")
print("            Cr : Mohammad Sheva (13519018)               ")
```

```
print("============================================================\n")

parse = util.reader() # Parsing dari text file , parse[0] = list isi dan
parse[1] = list codelist
inittime = time() # Time start

print("\nInput :")
util.inputPrinter(parse[0]) # Print input

adjmat = sols.makeAdj(parse[0],parse[1]) # Merepresentasikan graf dalam
matriks adjacency

listIn = sols.countIn(adjmat) # Membuat list yang berisi derajat masuk
tiap node

order = sols.topSort(adjmat,listIn) # Membuat list yang berisi index
kode mata kuliah  yang sudah terurut menggunakan topological sort

if(len(order)!=0):
    print("\nBanyak Mata Kuliah :",end=" ")
    print(len(parse[1]))
    print("Banyak Semester yang dibutuhkan :",end=" ")
    print(len(order))
    finList = util.generateCode(order,parse[1]) # Membuat list yang
berisi kode mata kuliah yang sudah terurut pada list order
    print("\nOutput :")
    util.printSems(finList) # Print output per semester

print("\nLama eksekusi : ",time()-inittime," detik")

exitter = input("Masukkan input apapun untuk keluar dari program...")
```

13519018_sols.py

```
'''
Nama      : Mohammad Sheva Almeyda Sofjan
NIM/Kelas : 13519018/K01
Deskripsi : Tugas Kecil 2 IF2211 Strategi Algoritma
            Aplikasi penyusunan kuliah
```

```
            menggunakan Topological Sort,
            penerapan Decrease n Conquer
Deskripsi File : File Algoritma Penyelesaian
'''

def makeAdj(isi,codelist):
    '''
    Membuat adjacency matrix
    isi : list yang berisi input pengguna
    codelist : list yang berisi kode mata kuliah, unik
    '''
    adjmat = [[0 for i in range(len(codelist))] for j in
range(len(codelist))]
    for sublist in isi:
        for i in range(len(sublist)):
            if(i > 0):

adjmat[codelist.index(sublist[i])][codelist.index(sublist[0])] = 1;
    return adjmat

def countIn(adjmat):
    '''
    Mencari derajat masuk tiap node, mereturn list yang berisi derajat
masuk tiap node
    adjmat : matriks adjacency
    '''
    listIn = [0 for i in range(len(adjmat))]
    for i in range(len(adjmat)):
        count = 0
        for j in range(len(adjmat[0])):
            if(adjmat[j][i]==1):
                count+=1
        listIn[i] = count
    return listIn

def topSort(adjmat,listIn):
    '''
    Topological Sort untuk menentukan urutan pengambilan mata kuliah,
implementasi source-removal algo (ref : Levitin page 141),
    mereturn list order yaitu list yang berisi urutan pengambilan mata
```

```
kuliah
    adjmat : matriks adjacency
    listIn : list berisi derajat masuk tiap node
    Decrease and conquer pada algoritma ini terletak pada pengulangan operasi pemilihan dan penghapusan node dengan derajat masuk 0
    sehingga banyaknya node yang harus diurus semakin berkurang hingga habis (memenuhi ketentuan stop pada while len(zeroIn) > 0)
    dan order mengandung semua node
    '''
    order = [] # inisialisasi list order dan zeroIn (list yang berisi node yang derajat masuknya = 0)
    zeroIn = []
    for i in range(len(listIn)): # Base case, Memasukkan node dengan derajat masuk 0 ke list zeroIn (dapat dianggap queue juga)
        if(listIn[i] == 0):
            zeroIn.append(i)
    while(len(zeroIn)>0): # Loop akan berjalan selama masih ada list dengan derajat masuk 0
        order.append(list(zeroIn)) # Memasukkan list zeroIn kedalam list order sehingga zeroIn merupakan sublist dari list order
        for num in range(len(zeroIn)): # Iterasi pada list zeroIn, sehingga dapat menghasilkan lebih dari satu mata kuliah pada satu semester
            currnode = zeroIn[0] # Ambil elemen pertama zeroIn
            nextnodes = getNextNodes(currnode,adjmat) # Node-node berikutnya dari currnode
            for node in nextnodes: # Iterasi pada node-node berikutnya (akan disebut node) dari currnode
                adjmat[currnode][node] = 0 # Mengeliminasi/menghapus edge dari currnode ke node
                listIn[node] -= 1  # Mengurangi derajat masuk node akibat perubahan pada perintah sebelumnya
                if(listIn[node] == 0): # Jika node sudah tidak memiliki derajat masuk akibat eliminasi
                    zeroIn.append(node) # Masukkan ke list zeroIn
            zeroIn.pop(0) # Hapus elemen pertama zeroIn (elemen pertama sudah selesai diproses)

    if(sum(listIn) > 0): # Input bukan DAG berarti masih ada edge yang tidak tereliminasi
```

```
        print("\nGraf input bukan merupakan DAG (Directed Acyclic
Graph), sehingga tidak dapat dicari urutan pengambilan mata kuliahnya.")
        return []
    else:
        return order # return urutan pengambilan

def getNextNodes(currnode,adjmat):
    '''
    Untuk mendapatkan node-node tetangga di mana terdapat busur dari
currnode ke node tetangga tersebut , mereturn list yang berisi informasi
tersebut
    currnode : node yang akan dicari tetangganya
    adjmat : adjacency matrix
    '''
    nextnodes = []
    for i in range(len(adjmat)):
        if(adjmat[currnode][i]==1):
            nextnodes.append(i)
    return nextnodes
```

13519018_util.py

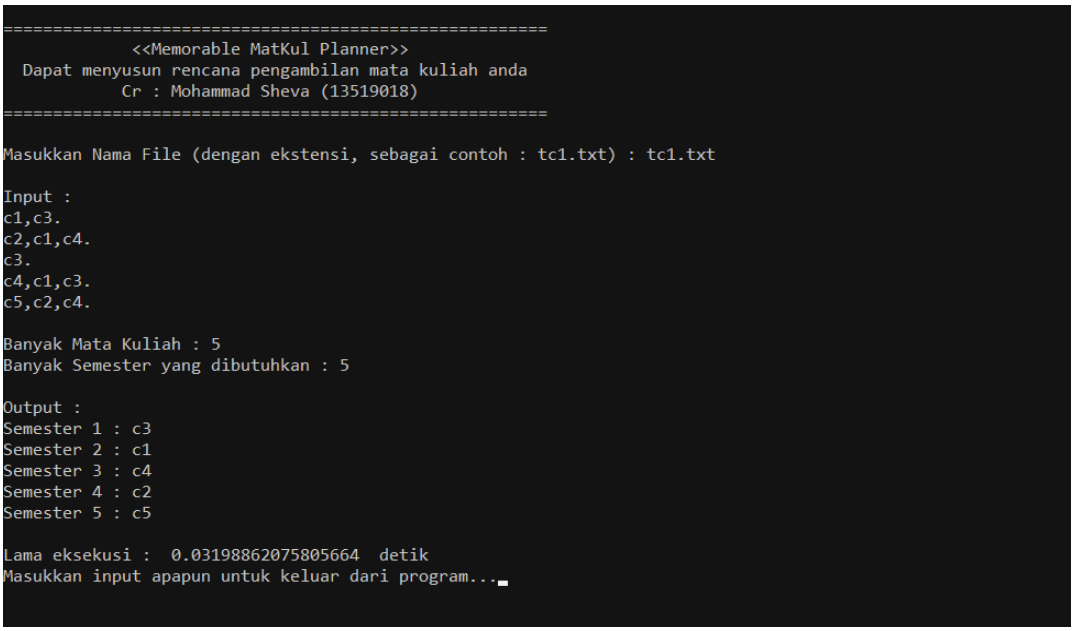
```
'''
Nama      : Mohammad Sheva Almeyda Sofjan
NIM/Kelas : 13519018/K01
Deskripsi : Tugas Kecil 2 IF2211 Strategi Algoritma
            Aplikasi penyusunan kuliah
            menggunakan Topological Sort,
            penerapan Decrease n Conquer
Deskripsi File : File Utility
'''

def reader():
    '''
    Reader file text, mereturn list isi(list berisi kode kuliah dan kode
kuliah prereq) dan list codelist(list yang berisi kode kuliah unik)
    '''
    notfound = True
    while notfound :
```

```
        fname = input("Masukkan Nama File (dengan ekstensi, sebagai
contoh : tc1.txt) : ")
        try:
            file = open('../test/'+fname,"r")
            notfound = False
        except:
             print("File tidak ditemukan. Ulangi lagi")

    isi = file.readlines() # Membaca isi file
    file.close()
    # Mengubah ke array, menghapus karakter tak guna
    for i in range(len(isi)):
        isi[i] = isi[i].replace("\n","")
        isi[i] = isi[i].replace(" ","")
        isi[i] = isi[i].replace(".","")
        isi[i] = isi[i].split(",")

    # List of Unique nodes/codes
    codelist = []

    # Append ke list of unique nodes/codes
    for sublist in isi:
        for code in sublist:
            try:
                codelist.index(code)
            except:
                codelist.append(code)

    codelist = sorted(codelist) #sort kode kuliah secara alfanumerik

    return isi,codelist

def inputPrinter(isi):
    '''
    Mencetak input(dari file teks) yang sudah dikonversi kedalam list
isi ke layar dalam format semula
    '''
    for sublist in isi:
        for i in range(len(sublist)):
            if(i < len(sublist)-1):
```

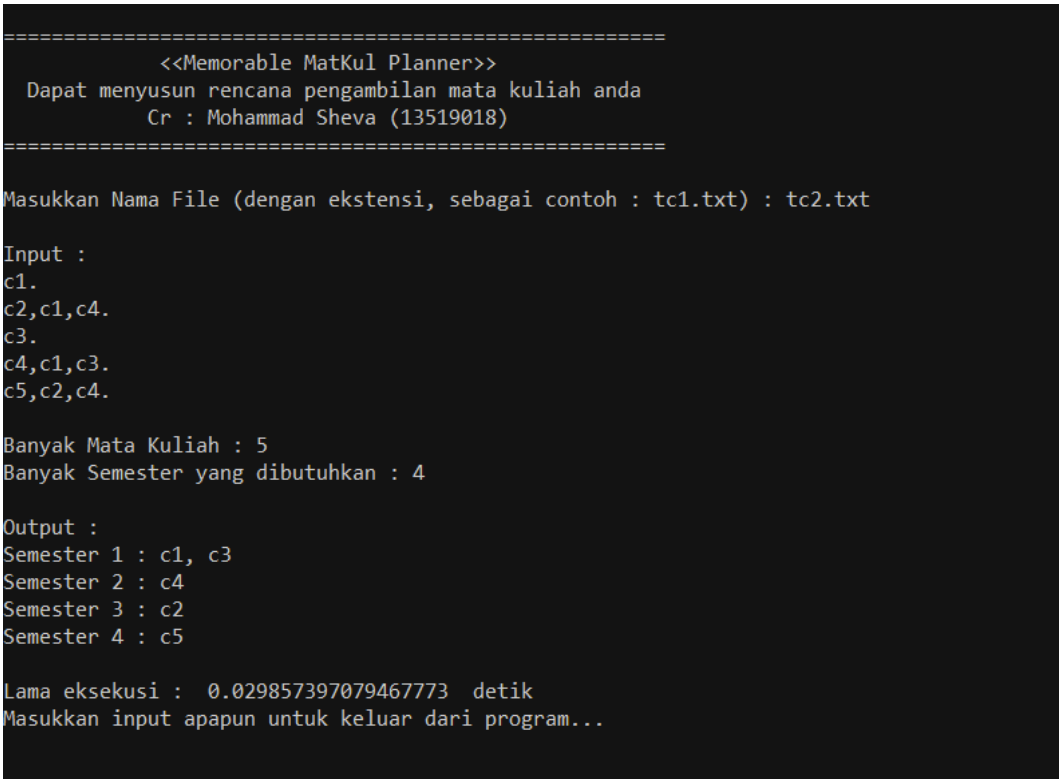
```
                print(sublist[i],end="")
                print(",",end="")
            else:
                print(sublist[i],end="")
                print(".")


def matrixPrinter(mat):
    '''
    Mencetak Matrix
    '''
    for i in range(len(mat)):
        for j in range(len(mat[0])):
            print(mat[i][j],end=" ")
        print()

def generateCode(order,codelist):
    '''
    Mencocokan hasil pengurutan dengan kode kuliah pada codelist
    '''
    finList = []
    for sublist in order:
        templist = []
        for num in sublist:
            templist.append(codelist[num])
        finList.append(list(templist))
    return finList

def printSems(finList):
    '''
    Print Output per semester
    '''
    i = 1
    for sublist in finList:
        print("Semester",end=" ")
        print(i,end=" : ")
        for j in range(len(sublist)-1):
            print(sublist[j],end=", ")
        print(sublist[len(sublist)-1])
        i+=1
```

SCREENSHOT HASIL TESTING

- Test Case 1



- Test Case 2 (modifikasi test case 1)



- Test Case 3 (*edgeless graph*)

```
=========================================================
            <<Memorable MatKul Planner>>
  Dapat menyusun rencana pengambilan mata kuliah anda
           Cr : Mohammad Sheva (13519018)
=========================================================

Masukkan Nama File (dengan ekstensi, sebagai contoh : tc1.txt) : tc3.txt

Input :
c1.
c2.
c3.
c4.
c5.

Banyak Mata Kuliah : 5
Banyak Semester yang dibutuhkan : 1

Output :
Semester 1 : c1, c2, c3, c4, c5

Lama eksekusi :  0.020122528076171875  detik
Masukkan input apapun untuk keluar dari program...
```

- Test Case 4

```
=========================================================
            <<Memorable MatKul Planner>>
  Dapat menyusun rencana pengambilan mata kuliah anda
           Cr : Mohammad Sheva (13519018)
=========================================================

Masukkan Nama File (dengan ekstensi, sebagai contoh : tc1.txt) : tc4.txt

Input :
Semester1_1.
Semester1_2.
Semester1_3.
Semester2_1,Semester1_1.
Semester2_2,Semester1_2.
Semester3_1,Semester2_1,Semester1_1.
Semester4_1,Semester3_1.
Semester5_1,Semester4_1.
Semester6_1,Semester5_1.
Semester6_2,Semester1_2,Semester2_2,Semester3_1,Semester4_1,Semester5_1.
Semester7_1,Semester6_1.
Semester8_1,Semester1_1,Semester7_1.

Banyak Mata Kuliah : 12
Banyak Semester yang dibutuhkan : 8

Output :
Semester 1 : Semester1_1, Semester1_2, Semester1_3
Semester 2 : Semester2_1, Semester2_2
Semester 3 : Semester3_1
Semester 4 : Semester4_1
Semester 5 : Semester5_1
Semester 6 : Semester6_1, Semester6_2
Semester 7 : Semester7_1
Semester 8 : Semester8_1

Lama eksekusi :  0.03987860679626465  detik
Masukkan input apapun untuk keluar dari program...
```

- Test Case 5 (graf siklik)

```
=========================================================
            <<Memorable MatKul Planner>>
  Dapat menyusun rencana pengambilan mata kuliah anda
           Cr : Mohammad Sheva (13519018)
=========================================================

Masukkan Nama File (dengan ekstensi, sebagai contoh : tc1.txt) : tc5.txt

Input :
c1.
c2,c1,c3.
c3,c4.
c4,c2.

Graf input bukan merupakan DAG (Directed Acyclic Graph), sehingga tidak dapat dicari urutan pengambilan mata kuliahnya.

Lama eksekusi :  0.010166645050048828  detik
Masukkan input apapun untuk keluar dari program...
```

- Test Case 6

```
=========================================================
           <<Memorable MatKul Planner>>
  Dapat menyusun rencana pengambilan mata kuliah anda
           Cr : Mohammad Sheva (13519018)
=========================================================

Masukkan Nama File (dengan ekstensi, sebagai contoh : tc1.txt) : tc6.txt

Input :
MA1101.
FI1101.
KU1101.
KU1102.
KU1011.
KU1024.
MA1201,MA1101.
FI1201,FI1101.
IF1210,KU1102.
IF2121,MA1101,MA1201.
IF2110,IF1210.
IF2120,MA1101,MA1201.
IF2124,IF1210.
IF2123,MA1101,MA1201.
IF2130,IF1210.
IF2210,IF2110.
IF2211,IF2120,IF2110.
IF2220,MA1101,MA1201,IF2120.
IF2230,IF2130.
IF2240,IF2121.
IF2250,IF2110.
IF3170,IF2121,IF2124,IF2220,IF2211.
IF3110,IF2110,IF2210.
IF3130,IF2230.
IF3141,IF2240,IF2250.
IF3150,IF2250.
IF3140,IF2240.
IF3151,IF2250.
IF3210,IF2130,IF2110.
IF3270,IF3170,IF2110.
IF3230,IF3130.
IF3250,IF3150,IF2250.
IF3260,IF2130,IF2110,IF2123.
IF3280,IF3151.
IF4020,IF2120,IF2110.
IF4040,IF2240.
IF4041,IF3270,IF2240.
IF4091,IF2121,IF2110,IF2120,IF2124,IF2123,IF2130,IF2210,IF2211,IF2220,IF2230,IF2240,IF2250,IF3170,IF3110,IF3130,IF3141,IF3150,IF3140,IF3151,IF3210,IF3270,IF3230,IF3250,IF3260,IF3280.
IF4092,IF4091.
```

```
Banyak Mata Kuliah : 39
Banyak Semester yang dibutuhkan : 8

Output :
Semester 1 : FI1101, KU1011, KU1024, KU1101, KU1102, MA1101
Semester 2 : FI1201, IF1210, MA1201
Semester 3 : IF2110, IF2124, IF2130, IF2120, IF2121, IF2123
Semester 4 : IF2210, IF2250, IF2230, IF3210, IF2211, IF2220, IF4020, IF2240, IF3260
Semester 5 : IF3110, IF3150, IF3151, IF3130, IF3170, IF3140, IF3141, IF4040
Semester 6 : IF3250, IF3280, IF3230, IF3270
Semester 7 : IF4041, IF4091
Semester 8 : IF4092

Lama eksekusi :  0.0450747013092041  detik
Masukkan input apapun untuk keluar dari program...
```

- Test Case 7

```
=========================================================
             <<Memorable MatKul Planner>>
  Dapat menyusun rencana pengambilan mata kuliah anda
           Cr : Mohammad Sheva (13519018)
=========================================================

Masukkan Nama File (dengan ekstensi, sebagai contoh : tc1.txt) : tc7.txt

Input :
PengKom.
DasPro,PengKom.
Alstrukdat,DasPro.
PBO,Alstrukdat.
Stima,Alstrukdat.

Banyak Mata Kuliah : 5
Banyak Semester yang dibutuhkan : 4

Output :
Semester 1 : PengKom
Semester 2 : DasPro
Semester 3 : Alstrukdat
Semester 4 : PBO, Stima

Lama eksekusi :  0.0  detik
Masukkan input apapun untuk keluar dari program...
```

- Test Case 8 (graf siklik)

```
=======================================================
            <<Memorable MatKul Planner>>
  Dapat menyusun rencana pengambilan mata kuliah anda
           Cr : Mohammad Sheva (13519018)
=======================================================

Masukkan Nama File (dengan ekstensi, sebagai contoh : tc1.txt) : tc8.txt

Input :
IF7011,IF7029,IF5018,IF7029,IF5018,IF6022,IF6013,IF6028,IF7020,IF6019,IF5018,IF7023,IF6022,IF7014,IF7029,IF7026.
IF5012.
IF6013,IF5021,IF6016,IF5021,IF5015,IF5012,IF6016,IF7011,IF6028,IF7020,IF6019,IF6028,IF6022,IF5021,IF7029,IF6028.
IF7014,IF7020,IF5030,IF7017,IF7029.
IF5015.
IF6016,IF5018,IF7011,IF5012,IF6025.
IF7017,IF7011,IF5024,IF7011,IF5012,IF7023,IF5027,IF5012,IF7020,IF5015,IF6028,IF6016,IF7014,IF5030,IF7020,IF6025,IF5018.
IF5018,IF5024,IF6025,IF5027,IF6013,IF5015,IF7026,IF6022,IF5024,IF7020,IF5027,IF5030,IF5021,IF7017,IF6019,IF7026,IF5021.
IF6019.
IF7020.
IF5021.
IF6022,IF6019,IF5024,IF7026,IF5021,IF7023,IF7014.
IF7023,IF6019,IF5021,IF5021,IF6019,IF6028,IF7029,IF6016,IF7017,IF5018,IF6028,IF6025,IF7014,IF5015.
IF5024,IF6019,IF5030.
IF6025.
IF7026.
IF5027,IF6025,IF5015,IF5015,IF5030,IF5024,IF5012,IF7020,IF7017.
IF6028,IF6019,IF7014,IF6019,IF6025,IF6013,IF6019,IF6025,IF5018,IF7011,IF5018,IF6016,IF7014,IF5030,IF6025,IF6013,IF7026,IF7020
IF7029,IF5030,IF5021,IF5012,IF6028,IF7023,IF5018.
IF5030,IF7017,IF6013,IF6019,IF5021,IF6016,IF5012,IF7014,IF6028,IF5015,IF7023,IF7011,IF7014.

Graf input bukan merupakan DAG (Directed Acyclic Graph), sehingga tidak dapat dicari urutan pengambilan mata kuliahnya.

Lama eksekusi :  0.03956913948059082  detik
Masukkan input apapun untuk keluar dari program...
```

- Test Case 9

```
=====================================================
            <<Memorable MatKul Planner>>
  Dapat menyusun rencana pengambilan mata kuliah anda
           Cr : Mohammad Sheva (13519018)
=====================================================

Masukkan Nama File (dengan ekstensi, sebagai contoh : tc1.txt) : tc9.txt

Input :
C42.
C3.
C35.
C38.
C36.
C37.
C43,C42.
C4,C3.
C5,C38.
C39,C38.
C34,C42.
C2,C38.
C8,C38,C5.
C6,C38,C5.
C7,C42,C43.
C10,C5.
C9,C42,C43,C5.
C11,C42,C43,C5.
C12,C6.
C13,C8.
C14,C9.
C15,C11.
C16,C7.
C17,C39,C5,C6.
C24,C7,C9,C14,C10.
C18,C6,C12.
C19,C15,C11.
C21,C16,C17.
C22,C17.
C20,C16.
C23,C17.
C25,C11,C6,C18.
C29,C24,C6.
C26,C19.
C27,C22,C17.
C28,C11,C6,C9,C24.
C30,C23.
C31,C30.
C32,C30.
C41,C35,C34,C36,C38.
C33,C31.
```

```
Banyak Mata Kuliah : 43
Banyak Semester yang dibutuhkan : 8

Output :
Semester 1 : C3, C35, C36, C37, C38, C42
Semester 2 : C4, C2, C39, C5, C34, C43
Semester 3 : C10, C6, C8, C41, C11, C7, C9
Semester 4 : C12, C17, C13, C1, C40, C15, C16, C14
Semester 5 : C18, C22, C23, C19, C20, C21, C24
Semester 6 : C25, C27, C30, C26, C28, C29
Semester 7 : C31, C32
Semester 8 : C33

Lama eksekusi :  0.1302506923675537  detik
Masukkan input apapun untuk keluar dari program...
```

- Test Case 10

```
Masukkan Nama File (dengan ekstensi, sebagai contoh : tc1.txt) : tc10.txt

Input :
We,are,no,strangers,to,love.
You,know,the,rules,and,so,do,I.
A,full,commitment,what,I,am,thinking,of.
I,just,wanna,tell,how,feeling.
Gotta,make,you,understand.
Never,gonna,give,you,up,let,down,run,around,and,desert,cry,say,goodbye,tell,lie,hurt.
are.
no.
strangers.
to.
love.
know.
the.
rules.
and.
so.
do.
full.
commitment.
what.
am.
thinking.
of.
just.
wanna.
tell.
how.
feeling.
make.
understand.
gonna.
give.
up.
let.
down.
run.
around.
desert.
cry.
say.
goodbye.
tell.
lie.
hurt.
```

```
Banyak Mata Kuliah : 44
Banyak Semester yang dibutuhkan : 3

Output :
Semester 1 : am, and, are, around, commitment, cry, desert, do, down, feeling, full, give, gonna, goodbye, how, hurt, just, know, let, lie, love, make, no, of, rules, run, say, so, strangers, tell, the, thinking
, to, understand, up, wanna, what, you
Semester 2 : We, I, Gotta, Never
Semester 3 : A, You

Lama eksekusi :  0.11982488632202148  detik
Masukkan input apapun untuk keluar dari program...
```

TAUTAN SOURCE CODE

Github : https://github.com/moshval/2CilStima

TABEL PENILAIAN

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	✓	

REFERENSI

[1] Levitin, A. *Introduction to the Design & Analysis of Algorithms*. Boston Pearson (2012)