

LAPORAN
Tugas Kecil 2 IF2121 Strategi Algoritma

DISUSUN OLEH

M. Reyhanullah Budiaman 13519045

INSTITUT TEKNOLOGI BANDUNG
SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
TEKNIK INFORMATIKA
2020/2021

BAB I

Deskripsi Masalah

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*. Berikut akan dijelaskan tugas yang dikerjakan secara detail.

1. Aplikasi akan menerima daftar mata kuliah beserta prasyarat yang harus diambil seorang mahasiswa sebelum mengambil mata kuliah tersebut. Daftar mata kuliah tersebut dituliskan dalam suatu file teks dengan format:

```
<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>.
.
.
.
```

Gambar 1. Format File Teks untuk Masukan Daftar Kuliah

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). Asumsi semua kuliah bisa diambil di sembarang semester, baik semester ganjil maupun semester genap.

2. Dari file teks yang telah diterima, ditentukan kuliah apa saja yang bisa diambil di semester 1, semester 2, dan seterusnya. Sebuah kuliah tidak mungkin diambil pada semester yang sama dengan prerequisitenya. Untuk menyederhanakan persoalan, tidak ada Batasan banyaknya kuliah yang bisa diambil pada satu semester.

Pendekatan Topological Sorting

a. Dari graf (DAG) yang terbentuk, hitung semua derajat-masuk (*in-degree*) setiap simpul, yaitu banyaknya busur yang masuk pada simpul tersebut. Pada contoh kasus di Gambar 2, maka derajat-masuk tiap simpul adalah sebagai berikut.

C1 : 1

C2 : 2

C3 : 0

C4 : 2

C5 : 2

b. Pilih sembarang simpul yang memiliki derajat-masuk 0. Pada kasus Gambar 2, pilih simpul C3.

c. Ambil simpul tersebut, dan hilangkan simpul tersebut beserta semua busur yang keluar dari simpul tersebut pada graf, dan kurangi derajat simpul yang berhubungan dengan simpul tersebut dengan 1.

Setelah simpul C3 dipilih, maka derajat simpul yang lain menjadi sebagai berikut.

C1 : 0

C2 : 2

C4 : 1

C5 : 2

Ulangi langkah (b) dan (c) hingga semua simpul pada DAG terpilih. Untuk kasus pada Gambar 2, setelah simpul terakhir dipilih rencana kuliah yang dihasilkan adalah sebagai berikut.

Semester I : C3

Semester II : C1

Semester III : C4

Semester IV : C2

Semester V : C5.

Kebetulan untuk contoh ini, satu semester hanya ada 1 kuliah.

3. Sediakan data uji sendiri, yang menjamin DAG jika diubah ke dalam representasi graf.

BAB II

IMPLEMENTASI PROGRAM

Bahasa pemograman yang digunakan adalah python. Data mata kuliah direpresentasikan ke dalam bentuk DAG(Directed Acyclic Graph) atau grap berarah non-siklus. Tiap-tiap matakuliah direpresentasikan dengan node dan sisi merepresentasikan suatu matakuliah merupakan prerequisite matakuliah tertentu. Algoritma yang digunakan adalah algoritma topological sort yang berkaitan dengan pendekatan decrease and conquer.

Algoritma decrease and conquer:

1. Hitung derajat masuk setiap simpul pada graf (N).
2. Eliminasi n simpul dengan derajat = 0 dan simpan simpul tersebut pada sebuah tuple solusi. Untuk setiap simpul yang dieliminasi, lakukan pengurangan derajat masuk dari simpul yang bersisian dari simpul yang dieliminasi (next node)
3. Ulangi proses 1-2 terhadap N-n simpul yang tersisa.

Jumlah iterasi yang dilakukan, dengan memperhitungkan iterasi untuk mengurangi derajat masuk, adalah $T(n) = n^2$ untuk worst case scenarionya, dan best case scenarionya adalah $T(n) = 1$. Sehingga kompleksitas algoritmanya adalah $O(n^2)$ yang tentunya lebih baik dibandingkan dengan menggunakan algoritma brute-force dengan kompleksitas $O(n^n)$

Berikut merupakan Source code dari program yang telah dibuat dalam bahasa python.

```
VISITED, UNVISITED = 1, 0

def topological_sort(graf, prec):
    in_edge = prec
    result_in_one_line = []
    result_by_semester = []

    def sort():
        state = []
        remove = []
        semester = []
        for i in in_edge:
            if(in_edge[i] == 0 and (i not in state)):
                result_in_one_line.append(i)
                remove.append(i)
                semester.append(i)
                for j in graf.getedge():
                    if(j[0] == i):
                        in_edge[j[1]] -= 1
                        state[j[1]] = VISITED

        for k in remove:
            del in_edge[k]

        result_by_semester.append(semester)
        return

    while(has_zero(in_edge)) :
        sort()

    return result_in_one_line, result_by_semester

def has_zero(prec):
    for i in prec:
        if(prec[i] == 0):
            return True
    else: return False

def printSchedule(schedule):
    print("Rencana Kuliah : ")
    print("-----")
    if(schedule != []):
        for semester in range(len(schedule)):
            if(semester > 8):
                break
            print("Semester " + str(semester + 1) + " : ", end='')
            for course in range(len(schedule[semester])):
                print(schedule[semester][course], end='')
                if(course != len(schedule[semester])-1):
                    print(", ", end='')
            print()

    print("-----")
    return
```

```
# GRAPH
```

```
class DAG:
    nodes = []
    edge = []

    def __init__(self, prec):
        node_list = []
        edge_list = []
        for course in prec:
            if(course not in node_list):
                node_list.append(course)
            for j in prec[course]:
                edge_list.append([j, course])

        for course in prec:
            for j in prec[course]:
                if(j not in node_list):
                    node_list.append(j)

        self.nodes = node_list
        self.edge = edge_list

    def getnodes(self):
        return self.nodes

    def getedge(self):
        return self.edge

    def getderajatmasuk(self):
        in_node = {}
        for course in self.nodes:
            count = 0
            for i in self.edge:
                if(i[1] == course):
                    count += 1
            in_node[course] = count

        return in_node
```

```

# MADE BY : 16519045 / M. Reyhanullah Budiaman
#
# This is a simple program for solving topological sort
# for making course schedule

from Dag import DAG
from topological import *

def readfile(file_name):
    cd_file = open("../test/" + file_name, 'r')
    matkul = cd_file.readlines()
    arr_of_course = []
    |
    for i in range(len(matkul)):
        arr_of_course.append(matkul[i].replace("\n", "").replace(".", ""))

    return arr_of_course

def getPrecDict(course):
    '''Get all preRec from course and returning it with dictionary representation.'''
    prec = {}

    for i in range(len(course)):
        deskripsi = course[i].split(", ")
        key = deskripsi[0]
        deskripsi.remove(key)
        prec[key] = deskripsi

    return prec

#MAIN ALGORITHM
exit_loop = 1
while(exit_loop):
    fl = input("Enter filename : ")

    prerequisite = getPrecDict(readfile(fl))

    graf = DAG(prerequisite)
    prerec_dict = graf.getderajatmasuk()

    result, schedule = topological_sort(graf, prerec_dict)

    printSchedule(schedule)

    decision = input("Try another file? (Y/n) : ")
    if(decision == "Y" or decision == "y"):
        exit_loop = 1
    else :
        exit_loop = 0

```

Fungsi – fungsi yang ada :

- read_code(file_name) : fungsi ini membaca daftar mata kuliah yang ada didalam file dan mengembalikannya dalam bentuk array
- getPrecDict(course) : fungsi ini mengubah membentuk dictionary dengan mata kuliah sebagai key
- topological_sort(graf, prec) : Menyusun rencana kuliah berdasarkan algoritma decrease and conquer
- printSchedule(schedule) : menampilkan solusi

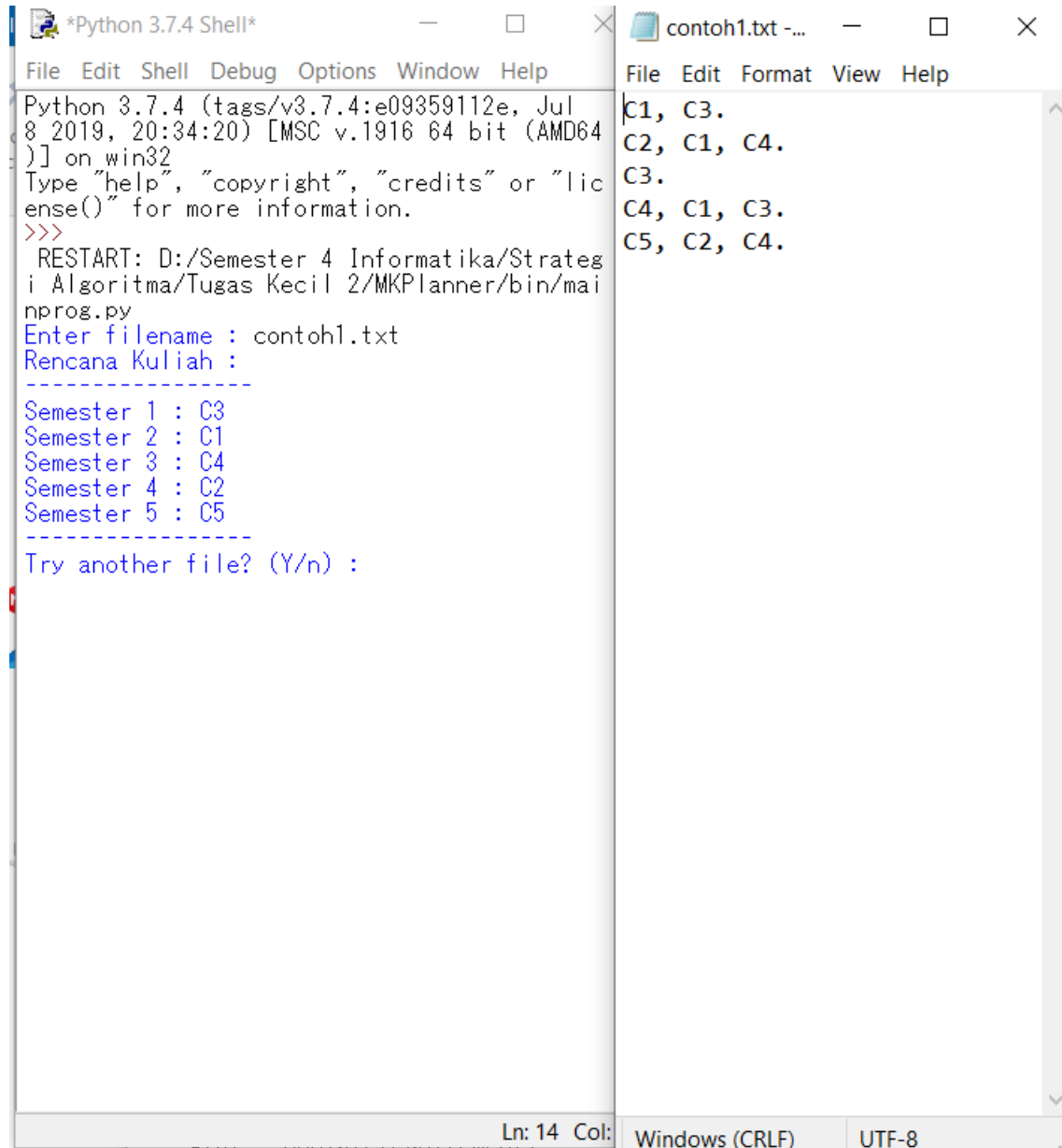
Class yang ada :

Class DAG : class untuk merekonstruksi graf

BAB III

EKSPERIMEN

Contoh 1 (contoh1.txt):

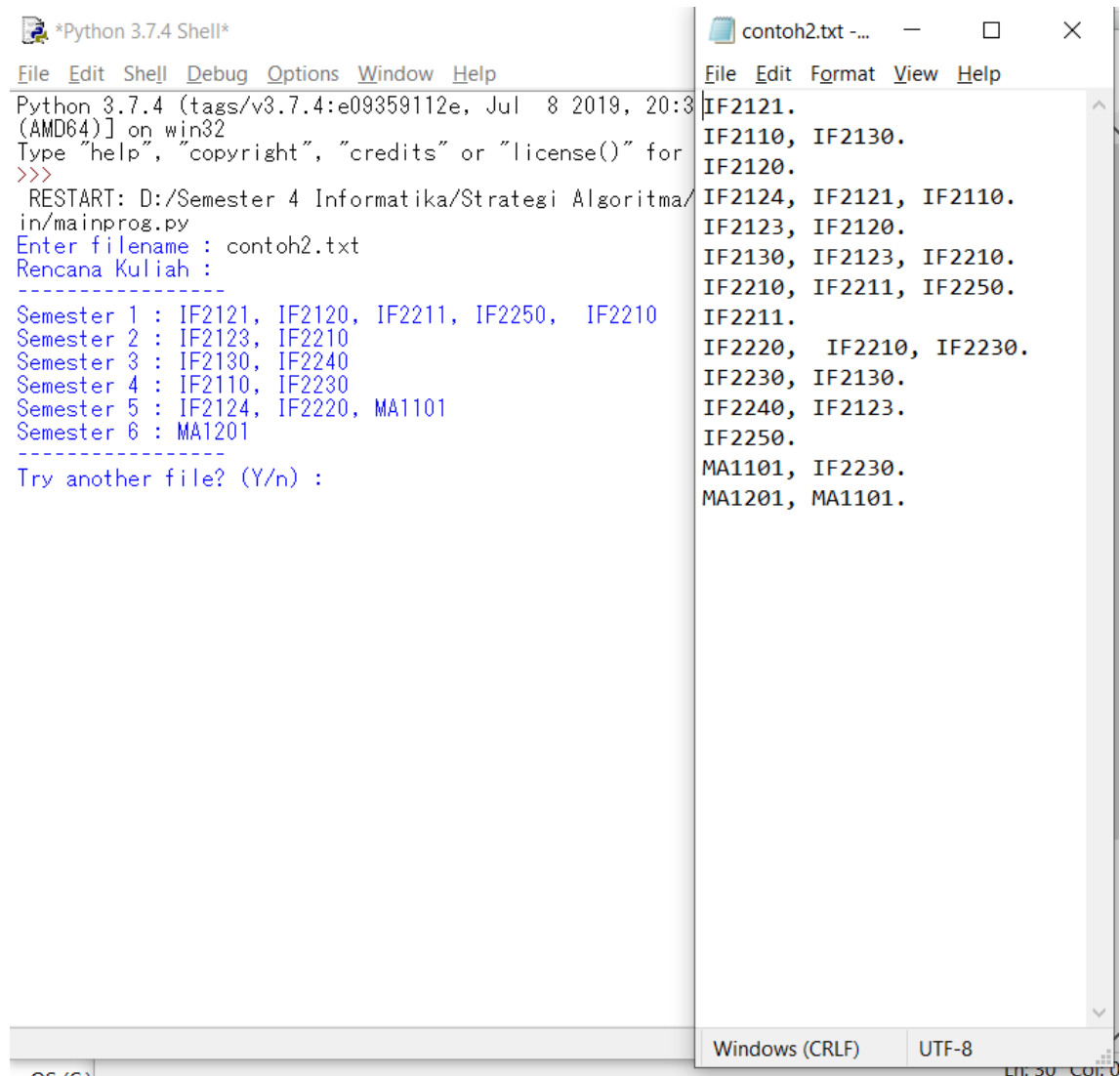


```
Python 3.7.4 (tags/v3.7.4:e09359112e, Jul 8 2019, 20:34:20) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
RESTART: D:/Semester 4 Informatika/Strategi Algoritma/Tugas Kecil 2/MKPlanner/bin/mainprog.py
Enter filename : contoh1.txt
Rencana Kuliah :
-----
Semester 1 : C3
Semester 2 : C1
Semester 3 : C4
Semester 4 : C2
Semester 5 : C5
-----
Try another file? (Y/n) :
```

```
C1, C3.
C2, C1, C4.
C3.
C4, C1, C3.
C5, C2, C4.
```

Ln: 14 Col: Windows (CRLF) UTF-8

Contoh 2 (contoh2.txt):



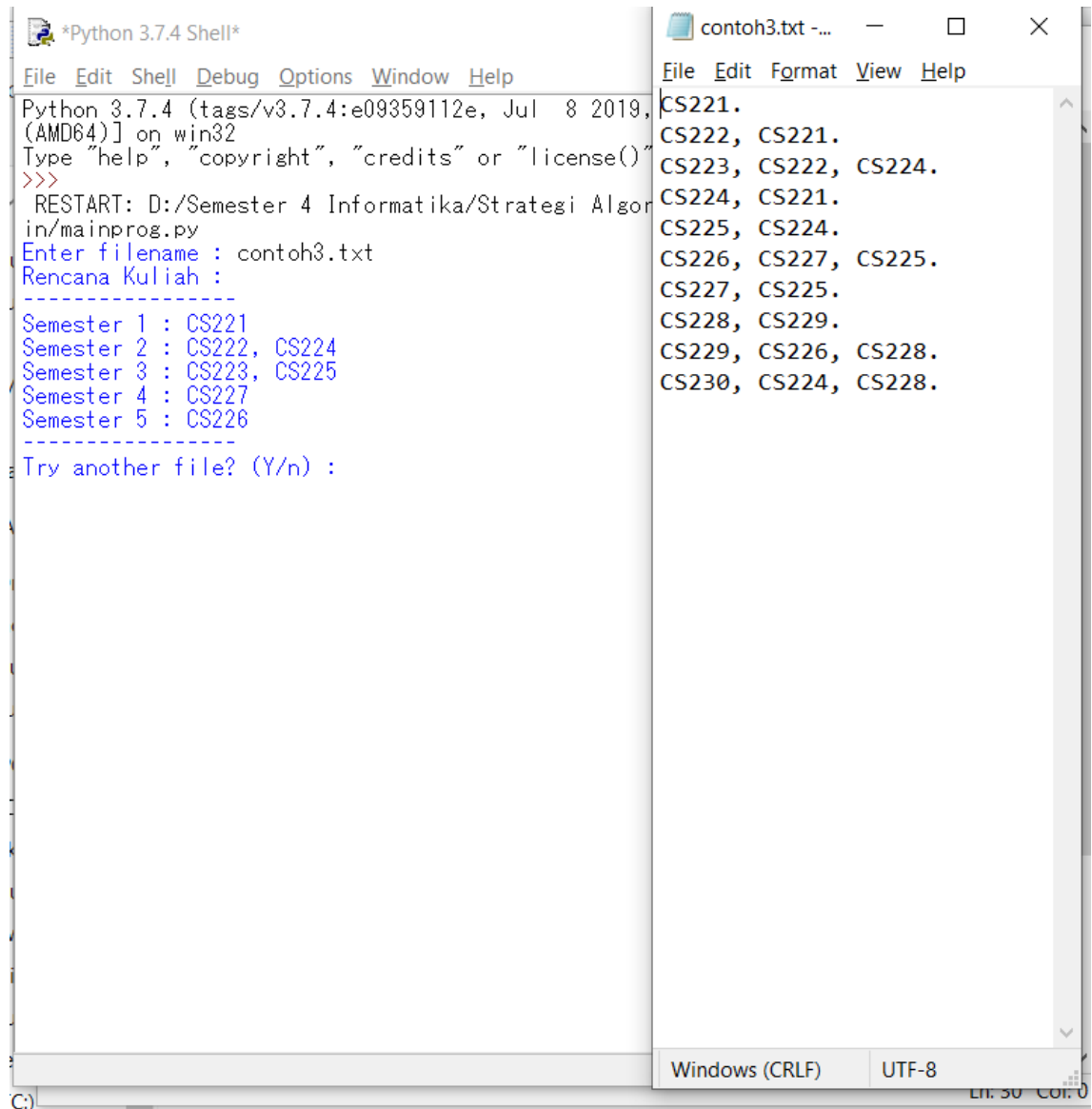
The image shows two overlapping windows. The background window is a Python 3.7.4 Shell with a menu bar (File, Edit, Shell, Debug, Options, Window, Help). The text in the shell shows the execution of a script named mainprog.py, which reads a file named contoh2.txt. The script outputs a list of course codes grouped by semester. The foreground window is a text editor titled 'contoh2.txt -...' with a menu bar (File, Edit, Format, View, Help). It displays the content of the file, which is a list of course codes grouped by semester, matching the output of the script in the shell.

```
Python 3.7.4 (tags/v3.7.4:e09359112e, Jul 8 2019, 20:30:00) on win32
Type "help", "copyright", "credits" or "license()" for more
>>>
RESTART: D:/Semester 4 Informatika/Strategi Algoritma/
in/mainprog.py
Enter filename : contoh2.txt
Rencana Kuliah :
-----
Semester 1 : IF2121, IF2120, IF2211, IF2250, IF2210
Semester 2 : IF2123, IF2210
Semester 3 : IF2130, IF2240
Semester 4 : IF2110, IF2230
Semester 5 : IF2124, IF2220, MA1101
Semester 6 : MA1201
-----
Try another file? (Y/n) :
```

```
File Edit Format View Help
IF2121.
IF2110, IF2130.
IF2120.
IF2124, IF2121, IF2110.
IF2123, IF2120.
IF2130, IF2123, IF2210.
IF2210, IF2211, IF2250.
IF2211.
IF2220, IF2210, IF2230.
IF2230, IF2130.
IF2240, IF2123.
IF2250.
MA1101, IF2230.
MA1201, MA1101.
```

Windows (CRLF) UTF-8

Contoh 3 (contoh3.txt):

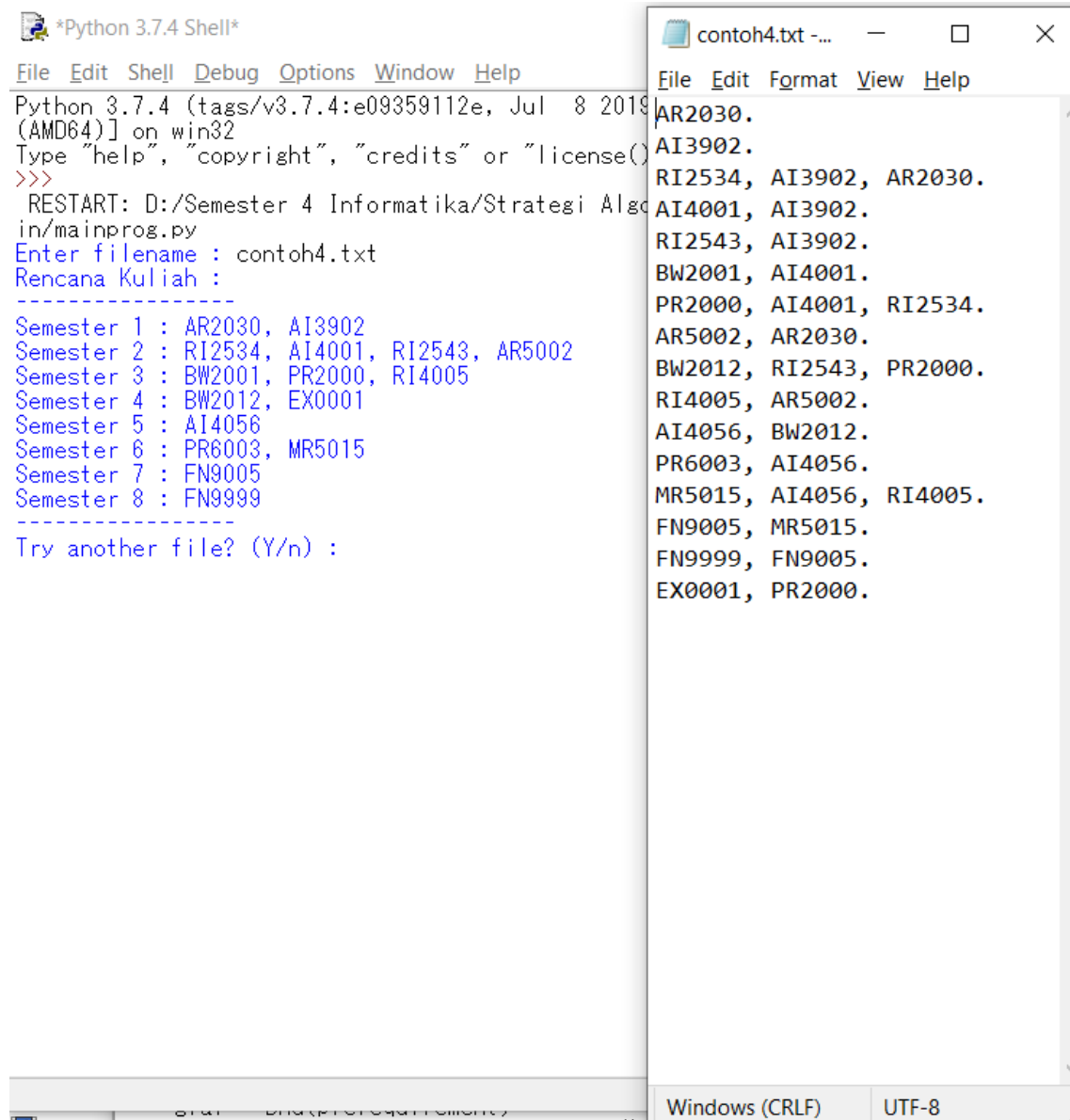


```
*Python 3.7.4 Shell*
File Edit Shell Debug Options Window Help
Python 3.7.4 (tags/v3.7.4:e09359112e, Jul 8 2019,
(AMD64)] on win32
Type "help", "copyright", "credits" or "license()"
>>>
RESTART: D:/Semester 4 Informatika/Strategi Algor
in/mainprog.py
Enter filename : contoh3.txt
Rencana Kuliah :
-----
Semester 1 : CS221
Semester 2 : CS222, CS224
Semester 3 : CS223, CS225
Semester 4 : CS227
Semester 5 : CS226
-----
Try another file? (Y/n) :

contoh3.txt -...
File Edit Format View Help
CS221.
CS222, CS221.
CS223, CS222, CS224.
CS224, CS221.
CS225, CS224.
CS226, CS227, CS225.
CS227, CS225.
CS228, CS229.
CS229, CS226, CS228.
CS230, CS224, CS228.

Windows (CRLF) UTF-8
```

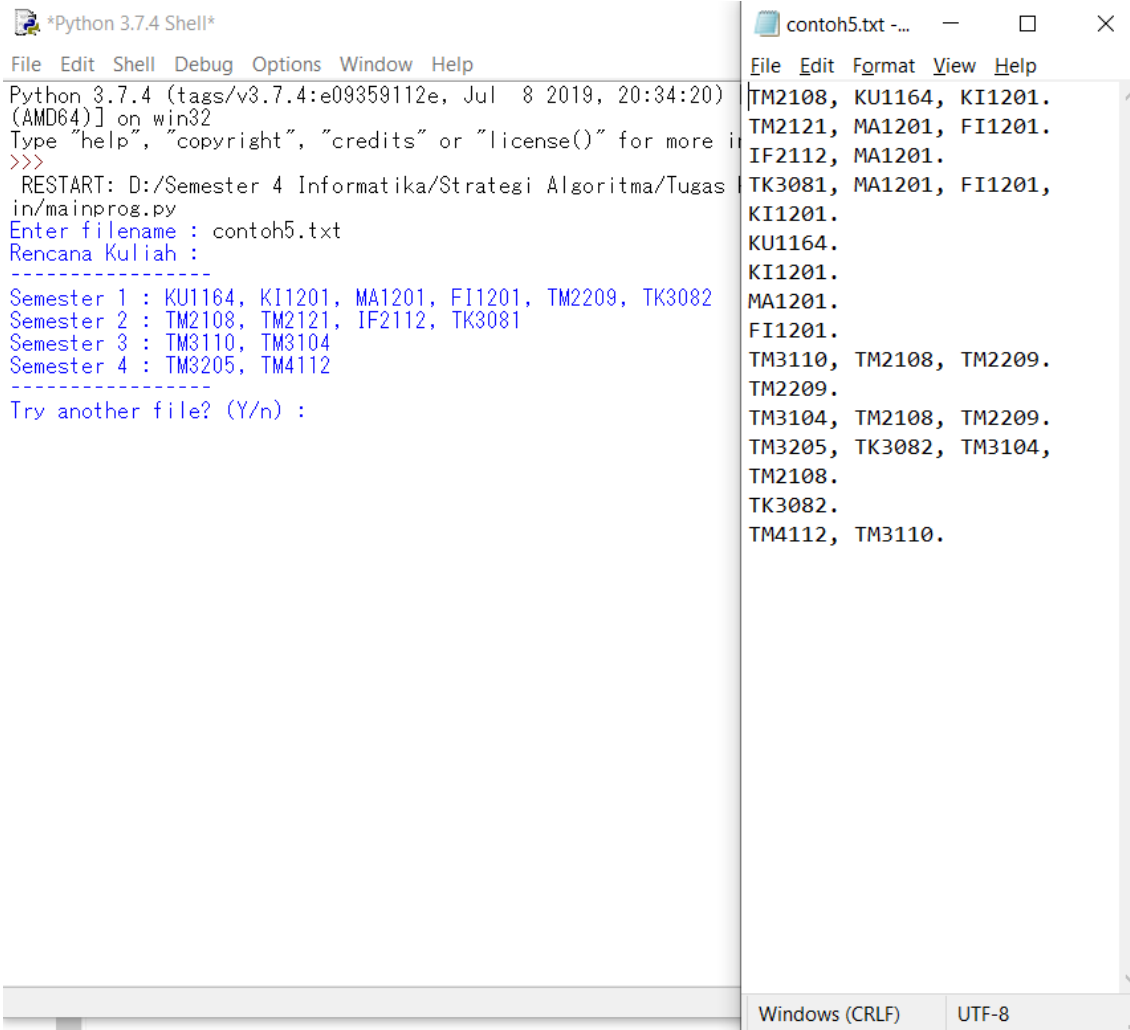
Contoh 4 (contoh4.txt):



```
*Python 3.7.4 Shell*
File Edit Shell Debug Options Window Help
Python 3.7.4 (tags/v3.7.4:e09359112e, Jul 8 2019) [AMD64] on win32
Type "help", "copyright", "credits" or "license()"
>>>
RESTART: D:/Semester 4 Informatika/Strategi Algoritma
in/mainprog.py
Enter filename : contoh4.txt
Rencana Kuliah :
-----
Semester 1 : AR2030, AI3902
Semester 2 : RI2534, AI4001, RI2543, AR5002
Semester 3 : BW2001, PR2000, RI4005
Semester 4 : BW2012, EX0001
Semester 5 : AI4056
Semester 6 : PR6003, MR5015
Semester 7 : FN9005
Semester 8 : FN9999
-----
Try another file? (Y/n) :

contoh4.txt -...
File Edit Format View Help
AR2030.
AI3902.
RI2534, AI3902, AR2030.
AI4001, AI3902.
RI2543, AI3902.
BW2001, AI4001.
PR2000, AI4001, RI2534.
AR5002, AR2030.
BW2012, RI2543, PR2000.
RI4005, AR5002.
AI4056, BW2012.
PR6003, AI4056.
MR5015, AI4056, RI4005.
FN9005, MR5015.
FN9999, FN9005.
EX0001, PR2000.
```

Contoh 5 (contoh5.txt):

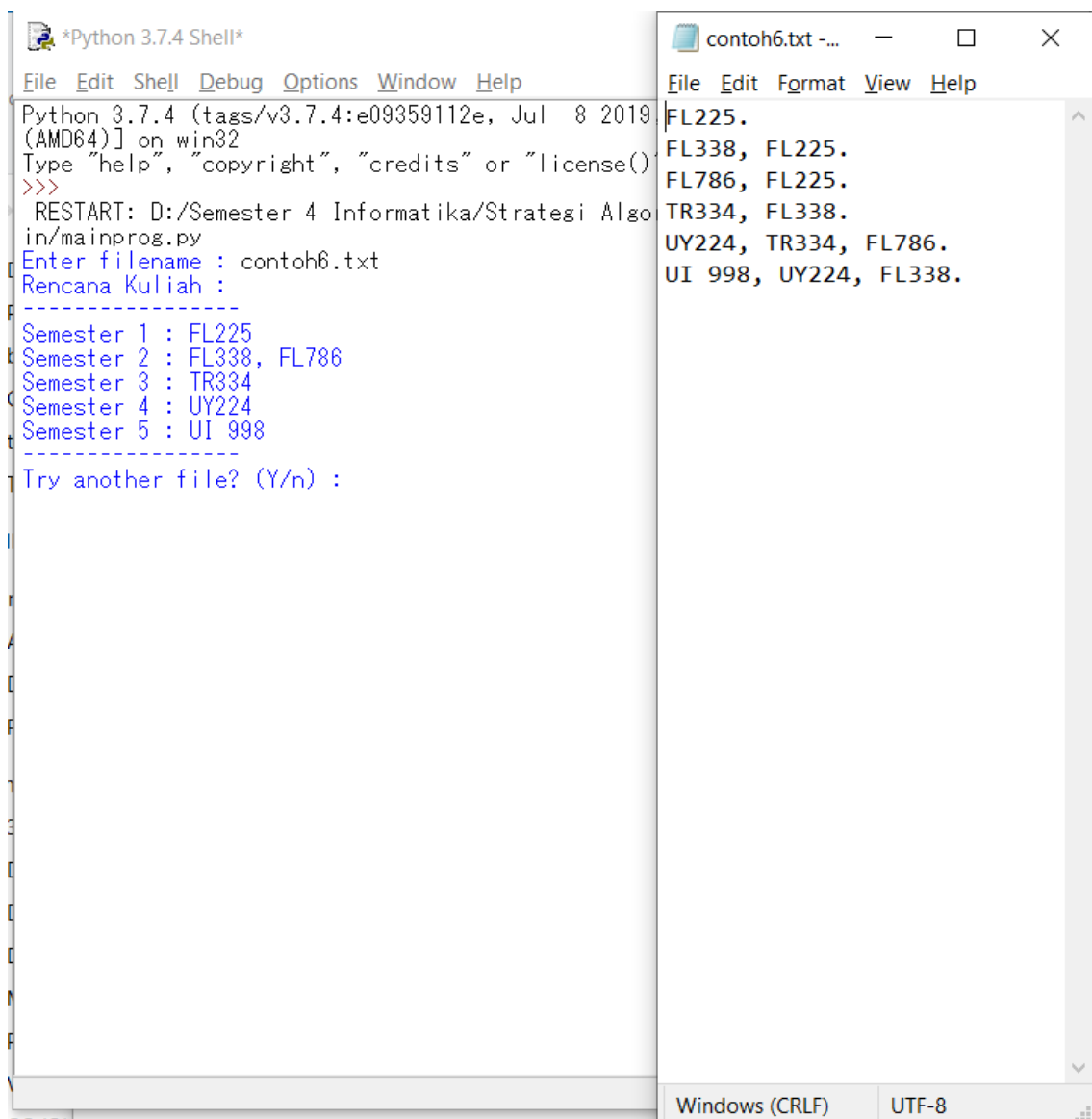


```
*Python 3.7.4 Shell*
File Edit Shell Debug Options Window Help
Python 3.7.4 (tags/v3.7.4:e09359112e, Jul 8 2019, 20:34:20) [AMD64] on win32
Type "help", "copyright", "credits" or "license()" for more
>>>
RESTART: D:/Semester 4 Informatika/Strategi Algoritma/Tugas 1
in/mainprog.py
Enter filename : contoh5.txt
Rencana Kuliah :
-----
Semester 1 : KU1164, KI1201, MA1201, FI1201, TM2209, TK3082
Semester 2 : TM2108, TM2121, IF2112, TK3081
Semester 3 : TM3110, TM3104
Semester 4 : TM3205, TM4112
-----
Try another file? (Y/n) :
```

```
contoh5.txt -...
File Edit Format View Help
TM2108, KU1164, KI1201.
TM2121, MA1201, FI1201.
IF2112, MA1201.
TK3081, MA1201, FI1201,
KI1201.
KU1164.
KI1201.
MA1201.
FI1201.
TM3110, TM2108, TM2209.
TM2209.
TM3104, TM2108, TM2209.
TM3205, TK3082, TM3104,
TM2108.
TK3082.
TM4112, TM3110.
```

Windows (CRLF) UTF-8

Contoh 6 (contoh6.txt):

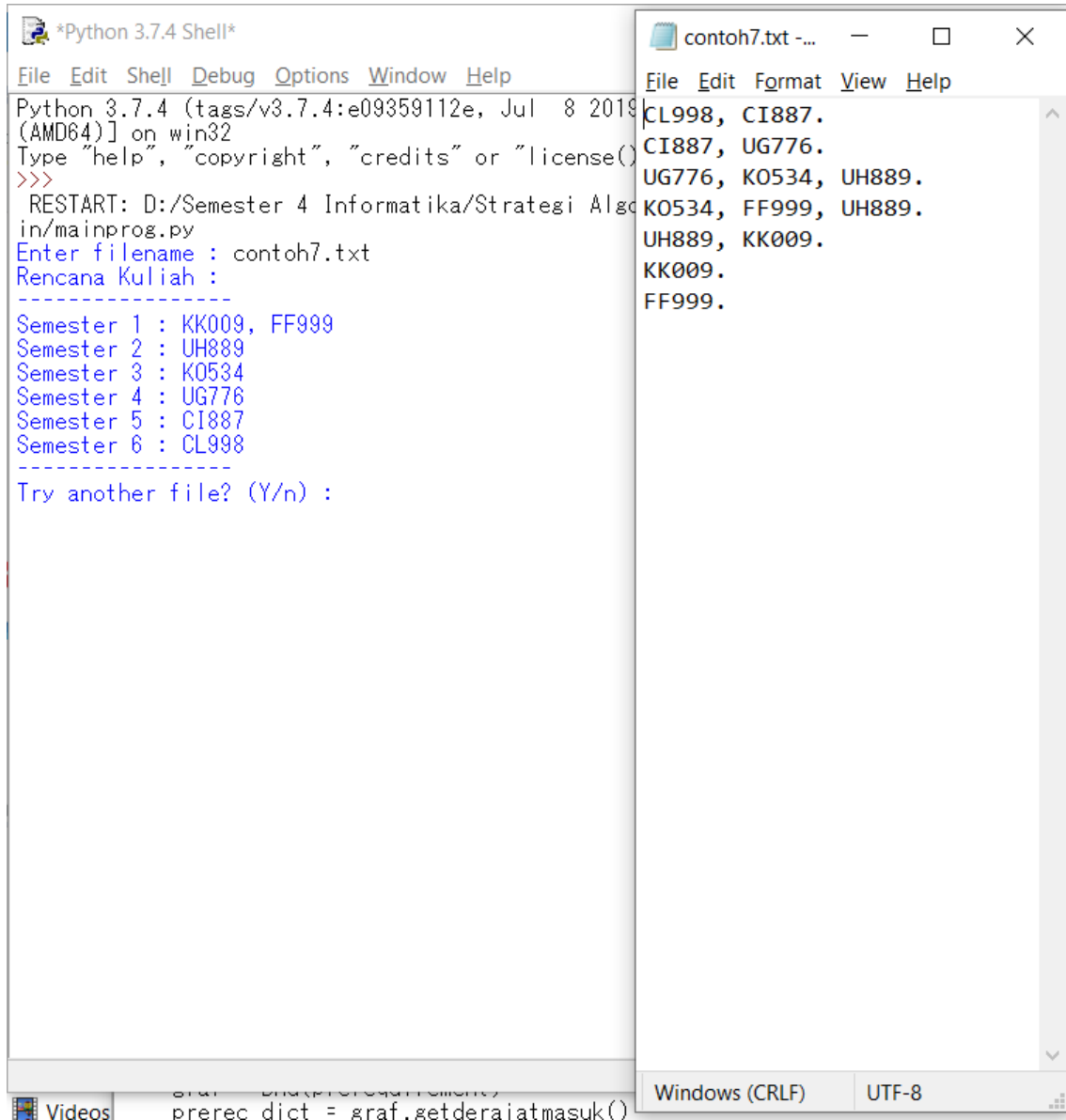


The image shows two overlapping windows. The left window is titled '*Python 3.7.4 Shell*' and displays the Python interpreter's startup sequence, including the version and platform information, followed by a restart command and the execution of a script named 'mainprog.py'. The script prompts the user to enter a filename, which is 'contoh6.txt', and then displays a list of course plans for five semesters. The right window is titled 'contoh6.txt -...' and shows the content of the file, which is a list of course plans for five semesters, matching the output of the script in the left window.

```
*Python 3.7.4 Shell*
File Edit Shell Debug Options Window Help
Python 3.7.4 (tags/v3.7.4:e09359112e, Jul 8 2019) on win32
Type "help", "copyright", "credits" or "license()"
>>>
RESTART: D:/Semester 4 Informatika/Strategi Algoritma
in/mainprog.py
Enter filename : contoh6.txt
Rencana Kuliah :
-----
Semester 1 : FL225
Semester 2 : FL338, FL786
Semester 3 : TR334
Semester 4 : UY224
Semester 5 : UI 998
-----
Try another file? (Y/n) :

contoh6.txt -...
File Edit Format View Help
FL225.
FL338, FL225.
FL786, FL225.
TR334, FL338.
UY224, TR334, FL786.
UI 998, UY224, FL338.
```

Contoh 7 (contoh7.txt):



The image shows two overlapping windows. The background window is a Python 3.7.4 Shell with the following text:

```
Python 3.7.4 (tags/v3.7.4:e09359112e, Jul 8 2019) [AMD64] on win32
Type "help", "copyright", "credits" or "license()"
>>>
RESTART: D:/Semester 4 Informatika/Strategi Algoritma/Python/
in/mainprog.py
Enter filename : contoh7.txt
Rencana Kuliah :
-----
Semester 1 : KK009, FF999
Semester 2 : UH889
Semester 3 : K0534
Semester 4 : UG776
Semester 5 : CI887
Semester 6 : CL998
-----
Try another file? (Y/n) :
```

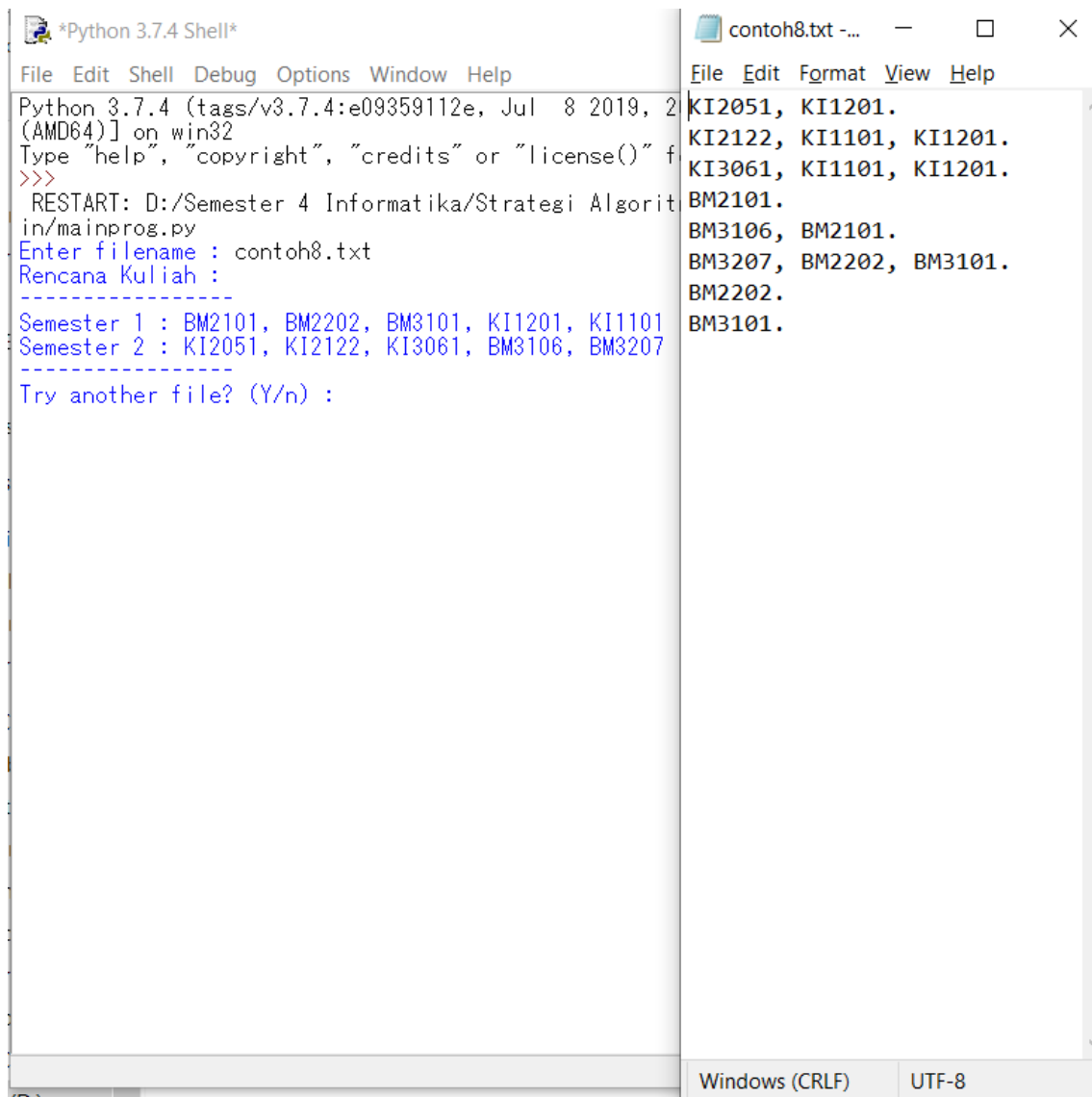
The foreground window is a text editor titled 'contoh7.txt -...' with the following text:

```
CL998, CI887.
CI887, UG776.
UG776, K0534, UH889.
K0534, FF999, UH889.
UH889, KK009.
KK009.
FF999.
```

At the bottom of the Python Shell window, a snippet of code is visible:

```
prerec dict = graf.getderajatmasuk()
```

Contoh 8 (contoh 8.txt):



The image shows two overlapping windows. The background window is a Python 3.7.4 Shell with a menu bar (File, Edit, Shell, Debug, Options, Window, Help). The text in the shell shows the execution of a program that reads 'contoh8.txt' and prints a list of course codes for Semester 1 and Semester 2. The foreground window is a text editor titled 'contoh8.txt -...' with a menu bar (File, Edit, Format, View, Help). It contains the same list of course codes as the shell output.

```
Python 3.7.4 (tags/v3.7.4:e09359112e, Jul 8 2019, 20:04:42) [AMD64] on win32
Type "help", "copyright", "credits" or "license()" for more
>>>
RESTART: D:/Semester 4 Informatika/Strategi Algoritma/mainprog.py
Enter filename : contoh8.txt
Rencana Kuliah :
-----
Semester 1 : BM2101, BM2202, BM3101, KI1201, KI1101
Semester 2 : KI2051, KI2122, KI3061, BM3106, BM3207
-----
Try another file? (Y/n) :
```

```
File Edit Format View Help
KI2051, KI1201.
KI2122, KI1101, KI1201.
KI3061, KI1101, KI1201.
BM2101.
BM3106, BM2101.
BM3207, BM2202, BM3101.
BM2202.
BM3101.
```

Windows (CRLF) UTF-8

BAB IV

LAMPIRAN

5.1 Alamat Repository

<https://github.com/reynull20/MKPlanner>

5.2 Penilaian

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.	✓	