Tugas Kecil II Strategi Algoritma

Laporan Penyelesaian Topological Sort dengan Decrease and Conquer

Oleh:

Kinantan Arya Bagaspati

13519044

PROGRAM STUDI TEKNIK INFORMATIKA

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

2021

A. Algoritma Decrease and Conquer

Algoritma decrease and conquer merupakan salah satu variasi dari sebuah algoritma yang lebih generalnya yakni divide and conquer. Sesuai namanya, algoritma decrease and conquer memiliki 2 bagian utama yakni:

- Decrease : Mereduksi persoalan menjadi persoalan yang lebih kecil
- Conquer : Memproses satu upa persoalan secara rekursif

Berbeda dengan divide and conquer, tidak ada tahap combine dalam algoritma decrease and conquer, karena tidak terjadi pembagian persoalan, melainkan hanya pengecilan ukuran persoalan yang sudah ada.

Pengecilan persoalan ini juga dapat dibagi lagi menjadi 3 varian, yakni:

- Decrease by a constant
  Ukuran persoalan dikurangi sebesar sebuah konstanta (biasanya bernilai 1) setiap pemanggilan metode decrease dalam algoritma. Contoh: Topological sort, Insertion sort, Selection sort.
- Decrease by a constant factor
  Ukuran persoalan dibagi dengan sebuah konstanta setiap pemanggilan metode decrease dalam algoritma. Contoh: Binary search (konstanta 2), Pencarian koin palsu (konstanta 3, karena dapat dibagi 3 kelompok dan timbang 2 kelompok diantaraya)
- Decrease by a variable size
  Pengurangan ukuran persoalan berbeda-beda tiap pemanggilan metode decrease-nya, bergantung pada sebuah variabel. Contoh: Interpolation search, Mencari nilai median.

Sebagaimana algoritma lanjutan lain pada umumnya, decrease and conquer hadir sebagai optimasi dari algoritma yang paling memakan waktu paling banyak yakni brute force. Namun optimasi ini tentunya bervariasi tergantung jenis persoalan yang diberikan. Decrease and conquer sangat bagus dipakai saat penyelesaian dari data yang menjadi persoalan memiliki metode yang serupa dengan penyelesaian sebagian dari data tersebut, terlebih lagi bila hasil dari sebagian dari data digunakan untuk mempermudah penyelesaian data awal. Decrease and conquer juga tidak selalu memberikan optimasi yang signifikan bila data yang digunakan merupakan worst case dan kebetulan metode pengecilan yang digunakan ialah decrease by a variable size, meski tentunya kemungkinan hal ini terjadi cukup kecil

Dalam menyelesaikan tugas kecil topological sort yang diberikan, saya menggunakan algoritma decrease and conquer yang mengutamakan kecepatan program. Terdapat satu hal utama yang berpengaruh secara signifikan pada waktu eksekusi program saya, yakni struktur data yang saya gunakan dalam menyimpan setiap pasangan

mata kuliah {x, y} dengan x ialah prasyarat dari y. Daripada saya menyimpan array of vector prequisites[NMAX] yang perquisites[i] menyimpan semua ID mata kuliah yang menjadi prasyaratnya, lebih baik saya menyimpan array of vector preqOf[NMAX] yang preqOf[i] menyimpan semua ID matakuliah yang memuat mata kuliah ber-ID i sebagai prasyaratnya. Ini dilakukan agar memudahkan mengurangkan jumlah prasyarat mata kuliah yang bersangkutan setelah mata kuliah ber-ID i diambil. Selain itu digunakan pula struktur data queue bernama queueCourse dengan implementasi array sebagai tempat memasukkan mata kuliah yang siap untuk diambil karena jelas dapat diperlakukan secara First In First Out. Selanjutnya akan dijelaskan alur kerja program beserta kaitannya dengan decrease and conquer.

Pertama program membaca file masukan sebanyak 2 kali dengan prosedur mapping() dan setPreqOf(). Pada mapping(), program hanya membaca kode mata kuliah pertama setiap baris untuk memberi ID setiap string yang terbaca, dan kemudian memasukkannya dalam map bernama courseCodetoID dengan key string yang terbaca dan value ID string tersebut, serta dalam array of string IDtoCourseCode yang fungsinya sesuai nama arraynya. Pada setPreqOf(), program sudah memberikan ID pada setiap mata kuliah yang tercatat dalam file sehingga sudah dapat dimasukkan ke dalam array of vector preqOf sesuai yang dijelaskan di paragraph sebelumnya. Prosedur ini juga mengisi array nbPreq dengan banyaknya prasyarat tiap ID mata kuliah terkait, serta mengisi queue dengan ID mata kuliah yang sudah bisa diambil (nbPreq[ID] == 0).

Dari kedua prosedur di atas, semua data sudah siap untuk diproses. Selanjutnya program akan menjalankan prosedur solve() yang menyelesaikan persoalan topological sort. Program akan menyimpan nilai tail saat ini dalam tempTail, kemudian jelas bahwa semua mata kuliah yang ada dalam queue saat ini (dari index head hingga tempTail) tentunya dapat diambil saat semester ini. Oleh karena itu program akan melakukan pop() pada queue satu per satu hingga head = tempTail, dengan mengurangi jumlah prasyarat sejumlah 1 dari semua mata kuliah yang memuat mata kuliah dengan ID hasil pop tersebut sebagai prasyaratnya. Disinilah preqOf dirasa sebagai struktur data yang superior. Kemudian setelah head = tempTail, program akan mempunyai queue baru, sisa mata kuliah serta sisa pasangan mata kuliah dan prasyaratnya yang tentunya berkurang jumlahnya. Itulah bagian decrease dari topological sort, dilanjutkan dengan conquer karena metode penyelesaian sisa mata kuliah yang ada sama persis dengan metode awal.

Analisis kompleksitas dari program topological sort ini cukup mudah. Misalkan V dan E berturut-turut merupakan banyak node (mata kuliah) dan banyak edge (relasi prasyarat) dari DAG (daftar prasyarat). Diawal program terdapat penyimpanan pasangan string dan IDnya dalam map yang jelas merupakan O(Vlog(V)). Kemudian pembacaan pada prosedur setPreqOf() juga masih perlu menggunakan map untuk mengidentifikasi ID dari string sehingga bernilai O(Elog(V)). Selanjutnya setiap pem-pop-an queue,

dilakukan perubahan nilai array nbPreq sebanyak mata kuliah yang memiliki head queue sebagai prasyarat. Apabila semua operasi tersebut ditotal, jelas memiliki kompleksitas O(E). Dengan menjumlahkan semua kompleksitas diperoleh kompleksitas total ialah O((V+E)logV).

Selain topological sort, saya juga mengimplementasikan test case generator yang mengenerate directed acyclic graph yang menandakan pasangan mata kuliah dan prasyaratnya dan kemudian menuliskannya dalam file sesuai format yang dibutuhkan untuk tucil ini. Program ini bernama DAGgenerator.cpp yang meminta input:

- majorCode sebagai kode jurusan
- nbCourse sebagai banyak mata kuliah yang ingin dihasilkan
- MaxSemester yang berarti DAG yang dihasilkan pasti memiliki solusi untuk banyaknya semester sama dengan maxSemester
- Density yang menandakan jumlah edge yang mungkin dalam graph tersebut. Apabila density bernilai 0 maka tidak ada edge, sementara bila bernilai 100000 maka setiap edge yang mungkin dibentuk akan terbentuk
- Pseudorandomizer yakni agar matakuliah yang dihasilkan tidak urut, setiap ID matakuliah dikalikan dengan nilai pseudorandomizer ini kemudian dimodkan dengan 100000. Pseudorandomizer harus relative prima dengan 10

Algoritma yang digunakan generator saya cukup simpel, pertama saya menginisialisasi array of vector preqs yang menyimpan prasyarat setiap ID mata kuliah. Kemudian setelah mendapat banyak mata kuliah dan jumlah semester, program akan melakukan ramdomisasi untuk menghasilkan barisan partition[maxSemester+1] bilangan terurut menaik yang akan berperan menjadi partisi, sehingga node dengan ID diantara partition[i] dan partition [i+1] tidak boleh memiliki prasyarat node dengan ID diatas partition[i+1] namun boleh memiliki prasyarat node dengan ID dibawah partition[i]. Setiap edge yang mungkin ada pun ditambahkan dengan probabilitas sesuai dengan density/100000. Kemudian program menuliskan setiap ID yang sudah dipseudirandomize ditambahkan dengan kode kuliah di depannya dalam file input{nbCourse}.txt.

B. Source Program

1. toposort_13519044.cpp

```cpp
#include<bits/stdc++.h>
using namespace std;

const long long NMAX = 200000;
map <string, long long> courseCodetoID; //untuk setiap (x,y) dalam map, berarti string x diberi ID y
string IDtoCourseCode[NMAX]; //IDtoCourseCode[i] menyimpan CourseCode dengan id i

ifstream input;
string line, toInsert;

void mapping(string filename){ //prosedur memetakan setiap string yang muncul dalam file
    filename = "../test/" + filename;
    input.open(filename.c_str());
    long long currentID = 0;
    while(getline(input, line)){
        //Mengambil kode kuliah pertama dalam tiap baris dan memasukkan ke dalam map
        toInsert = "";
        for(int i=0; line[i]!='.' && line[i]!=','; i++){
            toInsert += " ";
            toInsert[i] = line[i];
        }
        courseCodetoID.insert({toInsert, currentID});
        IDtoCourseCode[currentID] = toInsert;
        currentID++;
    }
    input.close();
}

vector<long long> preqOf[NMAX]; //matkul dengan ID x ialah prasyarat dari matkul dengan ID preqOf[x][i]
long long nbPreq[NMAX]; //banyak prasyarat untuk matkul x ialah nbPreq[x]
long long courseID, preqID;

long long queueCourse[NMAX], head, tail, tempTail;
//queue dengan implementasi array,
//saya tidak menggunakan std::queue karena ingin menyimpan index head dan tail yang nantinya digunakan untuk tempTail

void setPreqOf(string filename){
//prosedur menyimpan setiap data prasyarat ke dalam array of vector preqOf sesuai definisi di atas
//prosedur ini juga mengisi array nbPreq dengan banyaknya prasyarat tiap ID mata kuliah terkait,
//serta mengisi queue dengan ID mata kuliah yang sudah bisa diambil (nbPreq[ID] == 0)
    filename = "../test/" + filename;
    input.open(filename.c_str());
    long long itr;
    //menginisialisasi queue
    head = 0;
    tail = 0;
    courseID = 0;
    while(getline(input, line)){
        nbPreq[courseID] =0;
        itr = IDtoCourseCode[courseID].length();
        while(line[itr] != '.'){
            itr+=2;
            //Menentukan preqID menggunakan map yang sudah ada
            toInsert = "";
            for(int i=itr; line[i]!='.' && line[i]!=','; i++){
                toInsert += " ";
                toInsert[toInsert.length()-1] = line[i];
                itr++;
            }
            preqID = courseCodetoID[toInsert];
            //Sesuai definisi preqOf, courseID lah yang dimasukkan dalam vector preqOf[preqID]
            preqOf[preqID].push_back(courseID);
            nbPreq[courseID]++;
        }
        //Memasukkan ke queue jika mata kuliah dapat langsung diambil
        if(nbPreq[courseID] == 0){
            queueCourse[tail] = courseID;
            tail++;
        }
        courseID++;
    }
    input.close();
}
```

```cpp
string toRoman(int number)
{
    int num[13] = {1,4,5,9,10,40,50,90,100,400,500,900,1000};
    string sym[13] = {"I","IV","V","IX","X","XL","L","XC","C","CD","D","CM","M"};
    int i=12;
    string result = "";
    while(number>0){
        int div = number/num[i];
        number = number%num[i];
        while(div--){
            result += sym[i];
        }
        i--;
    }
    return result;
}
```

```cpp
ofstream output;
void solve(long long semester){
    if(head<tail){
        output << "Semester " << toRoman(semester) << ":";
        tempTail = tail;
        bool first = true;
        //setiap matakuliah yang ada di queue saat ini juga bisa diambil untuk semester ini
        while(head < tempTail){
            if(first){
                output << " " << IDtoCourseCode[queueCourse[head]];
                first = false;
            }else{
                output << ", " << IDtoCourseCode[queueCourse[head]];
            }
            //setiap matakuliah yang diambil akan mengurangi jumlah prasyarat mata kuliah lain
            //yang memiliki matakuliah ini sebagai prasyaratnya, oleh karena itu array of vector
            //preqOf cocok untuk melakukan hal tersebut
            for(int i=0; i<preqOf[queueCourse[head]].size(); i++){
                preqID = preqOf[queueCourse[head]][i];
                nbPreq[preqID]--;
                //bila prasyarat habis, masukkan ke queue
                if(nbPreq[preqID] == 0){
                    queueCourse[tail] = preqID;
                    tail++;
                }
            }
            head++;
        }
        output << endl;
        //lakukan rekursi untuk semester berikutnya
        solve(semester+1);
    }else{
        output.close();
    }
}
```

```
clock_t start, endtime;
int main(){

    string filename;
    cout << "Masukkan namafile: "; cin >> filename;

    start = clock();
    mapping(filename);
    setPreqOf(filename);
    output.open("../test/output.txt");
    solve(1);

    endtime = clock();
    double time_taken = double(endtime - start) / double(CLOCKS_PER_SEC);
    cout << "Time taken by program is : " << fixed
        << time_taken << setprecision(5);
    cout << " sec " << endl;
}
```

2. DAGgenerator.cpp

```
#include<bits/stdc++.h>
using namespace std;
//Untuk generate file Directed Acyclic Graph
//yang ditulis dalam format sesuai input dari Tucil 2 STIMA

const long long NMAX = 100000;
long long pseudoRandomizer;
string majorCode;
long long density; //(peluang sebuah pasangan node connected)*NMAX
long long nbCourse; //(nbCourse**2) * density < 1e12 agar banyaknya edge kurang dari 1e6
long long maxSemester;
//solusi dari generator ini dapat diselesaikan dengan maksimal (maxSemester) semester saja
//Asumsikan variabel diatas kurang dari 99
long long partitions[1000];
bool included[NMAX];
long long prob;

vector <long long> preqs[NMAX];
ofstream output;

long long randomize(long long i){
    return (pseudoRandomizer*i)%NMAX;
}

string toString (long long x, long long digits){
    string res = "";
    for(int i=0; i<digits; i++){
        res += "0";
    }
    while(x>0){
        digits--;
        res[digits] += x%10;
        x/=10;
    }
    return res;
}
```

```
string toString (long long x){
    string result = "";
    while(x>0){
        result = "0"+result;
        result[0] += x%10;
        x/=10;
    }
    return result;
}
void write(){
    string filename = "../test/input"+toString(nbCourse)+".txt";
    output.open (filename.c_str());
    for(int i=0; i<NMAX; i++){
        if(included[i]){
            output << majorCode << toString(i, 5);
            for(int j=0; j<preqs[i].size(); j++){
                output << ", " << majorCode << toString(preqs[i][j], 5);
            }
            output << ".\n";
        }
    }
    output.close();
}
```

```
int main(){
    //inisialisasi randomizer dan input
    srand (time(NULL));
    cout << "Masukkan kode jurusan: "; cin >> majorCode;
    cout << "Masukkan density(<=100000): "; cin >> density;
    cout << "Masukkan nbCourse: "; cin >> nbCourse;
    cout << "Masukkan maxSemester (<99): "; cin >> maxSemester;
    cout << "Masukkan pseudorRandomizer(relatif prima dengan 10): "; cin >> pseudoRandomizer;
    for(int i=0; i<NMAX; i++){
        included[i] = false;
    }
    //Randomize barisan yang menjadi partisi
    partitions[0] = 1;
    for(int i=1; i<maxSemester; i++){
        partitions[i] = rand() % nbCourse;
        partitions[i]++;
    }
    partitions[maxSemester] = nbCourse+1;
    sort(partitions, partitions+maxSemester+1);
    //setiap edge yang mungkin untuk ada
    //(dari node pada partisi i ke node pada partisi j dengan i<j)
    //akan ditambahkan dengan probabilitas sebesar (density/100000)
    for(int i=0; i<maxSemester; i++){
        for(int j=partitions[i]; j<partitions[i+1]; j++){
            included[randomize(j)] = true;
            for(int k=partitions[0]; k<partitions[i]; k++){
                prob = rand()%NMAX;
                if(prob < density){
                    preqs[randomize(j)].push_back(randomize(k));
                }
            }
        }
    }
    write();
}
```

C. Input dan Output

input-0.txt

tucil2 > test > input-0.txt

```
1   C1, C3.
2   C2, C1, C4.
3   C3.
4   C4, C1, C3.
5   C5, C2, C4.
```

output.txt

tucil2 > test > output.txt

```
1   Semester I: C3
2   Semester II: C1
3   Semester III: C4
4   Semester IV: C2
5   Semester V: C5
```

C:\Windows\System32\cmd.exe

```
D:\Kuliah\Semester4\STIMA\tucil\tucil2\src>toposort
Masukkan namafile: input-0.txt
Time taken by program is : 0.001000 sec
```

input.txt | toposort_13519044.cpp | toposort

tucil2 > test > input.txt

```
1   Kalkulus Peubah Banyak, Kalkulus Diferensial, Ka
2   Kalkulus Integral, Kalkulus Diferensial.
3   Kalkulus Diferensial.
4   Analisis Vektor, Kalkulus Diferensial, Kalkulus
5   Geometri Analitik Ruang, Geometri Ruang, Geometr
6   Geometri Ruang, Geometri Bidang.
7   Geometri Analitik Bidang, Geometri Bidang.
8   Geometri Bidang.
9   Program Linear, Aljabar Matriks, Aljabar Linear.
10  Aljabar Matriks.
11  Aljabar Linear.
```

output.txt

tucil2 > test > output.txt

```
1   Semester I: Kalkulus Diferensial, Geometri Bidang, Aljabar Matr
2   Semester II: Kalkulus Integral, Geometri Ruang, Geometri Analit
3   Semester III: Kalkulus Peubah Banyak, Analisis Vektor, Geometri
4
```

C:\Windows\System32\cmd.exe

```
D:\Kuliah\Semester4\STIMA\tucil\tucil2\src>toposort
Masukkan namafile: input.txt
Time taken by program is : 0.000000 sec

D:\Kuliah\Semester4\STIMA\tucil\tucil2\src>
```

input-2.txt

tucil2 > test > input-2.txt

```
1   Bahasa Inggris.
2   Matematika I.
3   Pendidikan Kewarganegaraan.
4   Pengantar Akuntansi I.
5   Pengantar Bisnis.
6   Pengantar Ekonomi Mikro.
7   Praktikum Aplikasi Komputer.
8   Sertifikasi I.
9   Studi Islam 1.
10  Bahasa Inggris Ekonomi.
11  Matematika II, Matematika I.
12  Pengantar Akuntansi II, Pengantar Akuntansi I.
13  Pengantar Ekonomi Makro
14  Pengantar Manajemen.
15  Statistika I, Matematika I.
16  Teori Ekonomi Mikro, Pengantar Ekonomi Mikro.
17  Ekonomi Moneter, Pengantar Ekonomi Makro.
18  Ekonomi Pembangunan, Pengantar Ekonomi Makro.
19  Ekonomi Sumber Daya Alam dan Lingkungan, Pengantar
20  Ilmu Kealaman Dasar.
21  Sertifikasi Bahasa Inggris I.
22  Sertifikasi II.
23  Statistika II, Matematika II, Statistika I.
24  Studi Islam 2.
25  Teori Ekonomi Makro, Pengantar Ekonomi Makro.
26  Akuntansi Sektor Publik, Pengantar Akuntansi II.
27  Ekonometrika I, Matematika II, Statistika II.
28  Ekonomi Internasional, Teori Ekonomi Mikro.
29  Ekonomi Internasional, Teori Ekonomi Makro.
30  Ekonomi Publik, Ekonomi Pembangunan.
31  Ekonomi Publik, Teori Ekonomi Makro.
32  Ekonomi Sumber Daya Manusia, Teori Ekonomi Mikro.
33  Ekonomi Sumber Daya Manusia, Teori Ekonomi Makro.
```

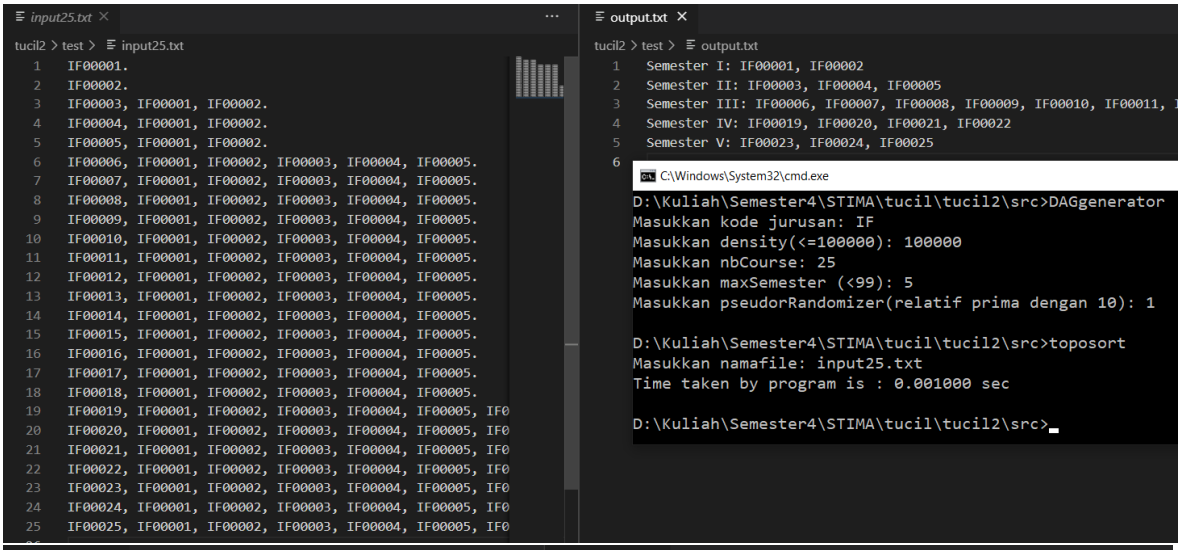
output.txt

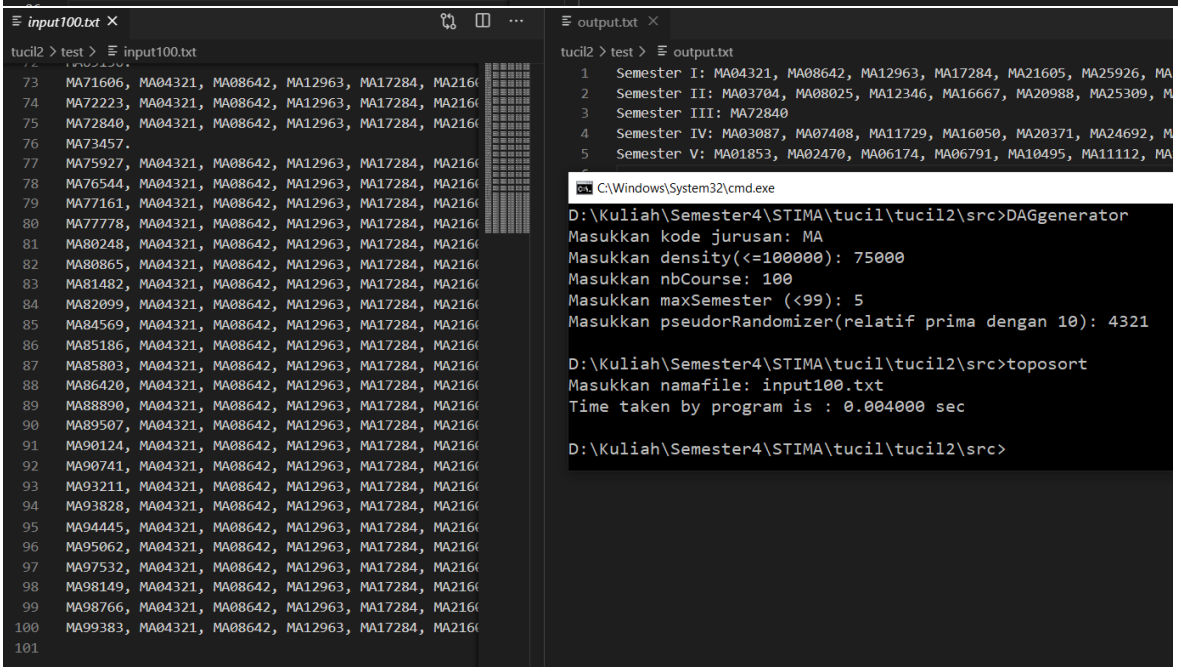
tucil2 > test > output.txt

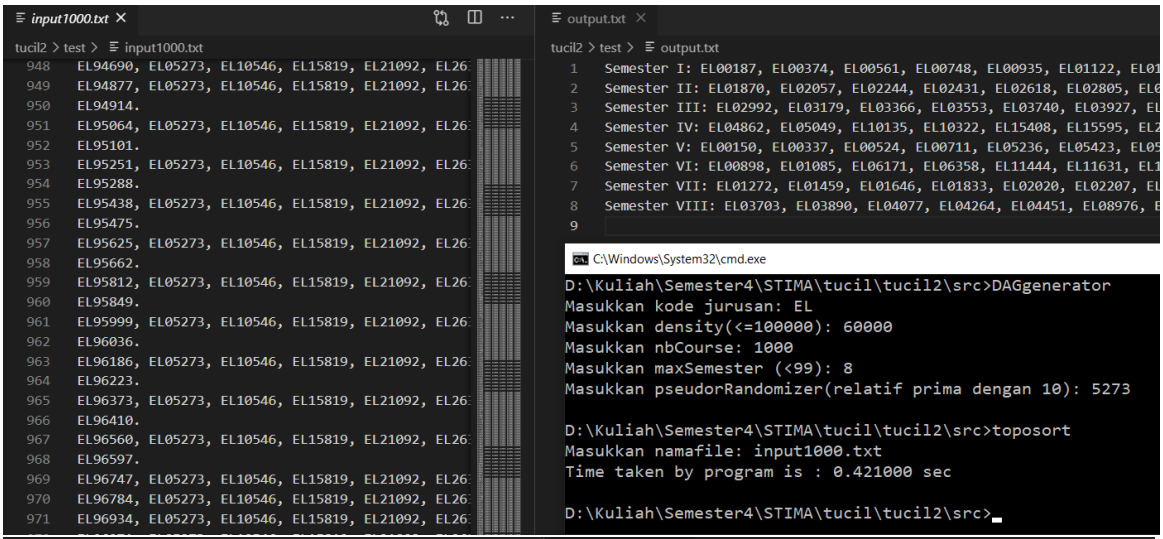
```
1   Semester I: Bahasa Inggris, Matematika I, Pendidikan Kewarganegaraan, Pengan
2   Semester II: Ekonomi Moneter, Ekonomi Pembangunan, Teori Ekonomi Makro, Mate
3   Semester III: Ekonomi Publik, Ekonomi Sumber Daya Alam dan Lingkungan, Ekono
4   Semester IV: Ekonometrika I
```

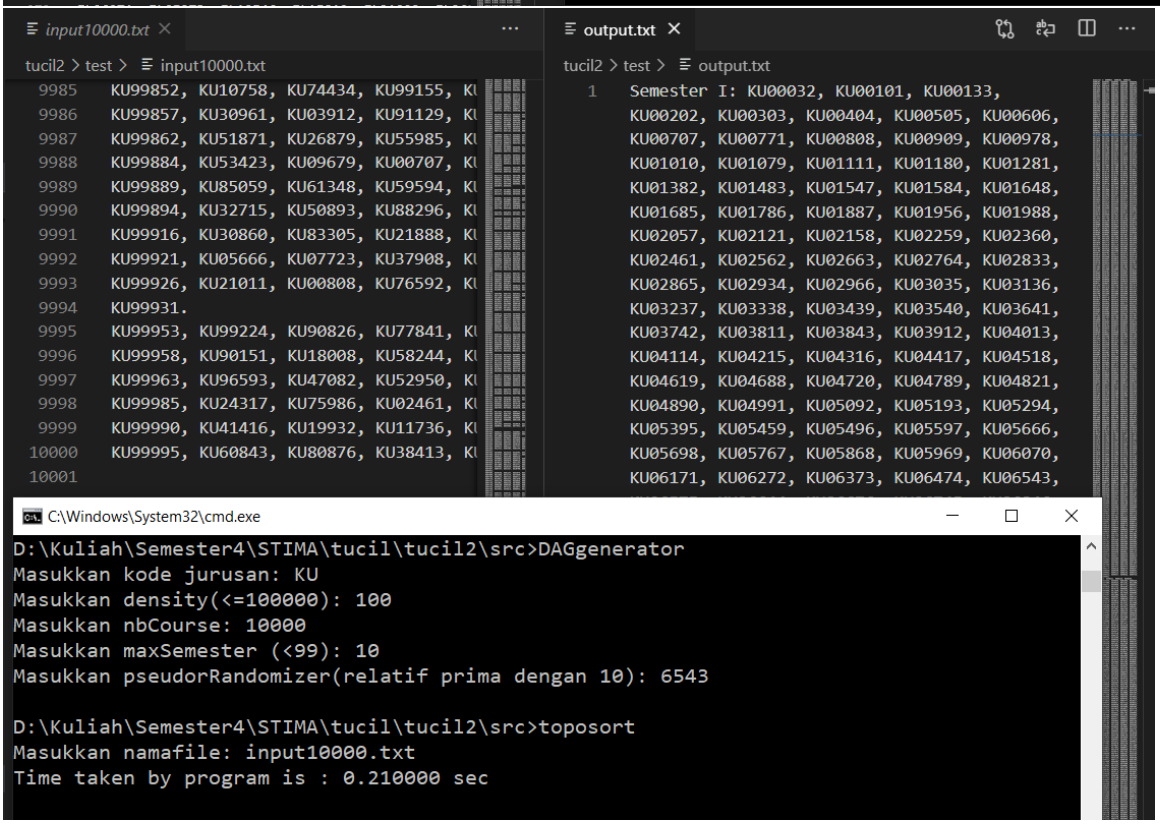
C:\Windows\System32\cmd.exe

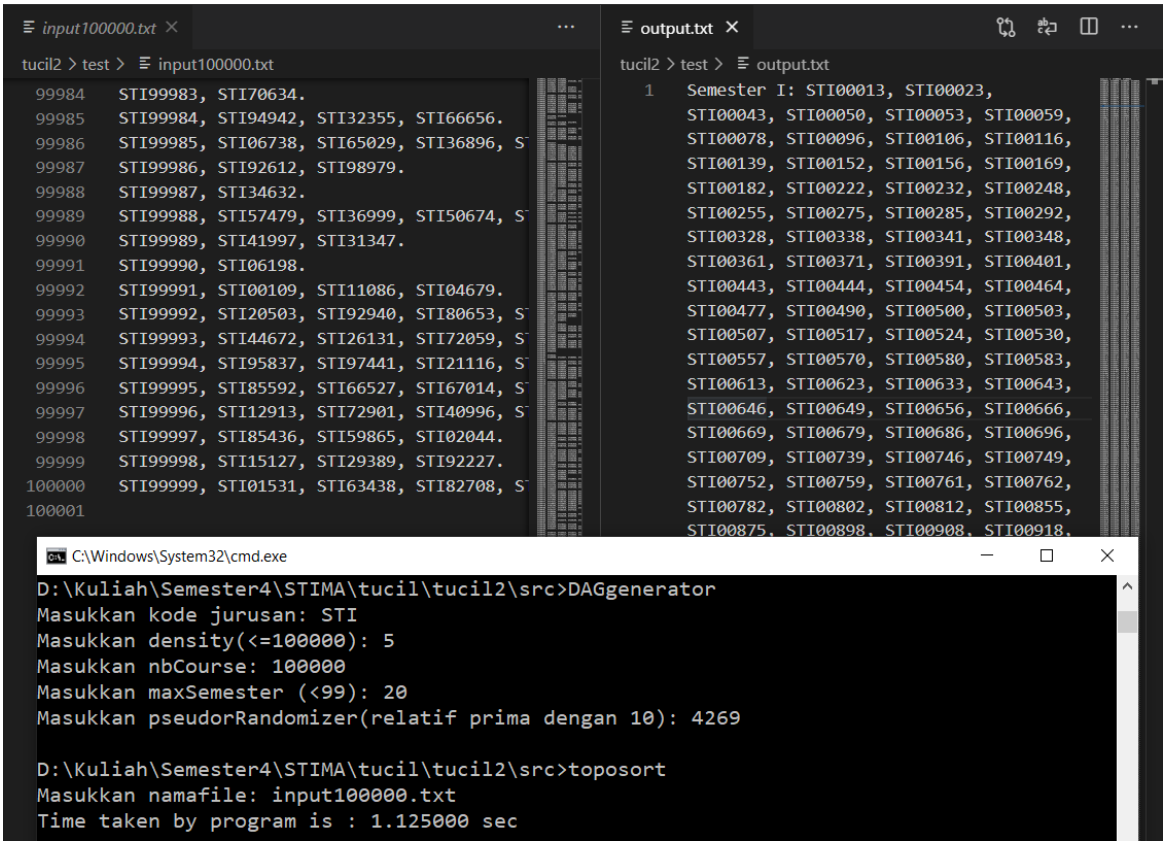
```
D:\Kuliah\Semester4\STIMA\tucil\tucil2\src>toposort
Masukkan namafile: input-2.txt
Time taken by program is : 0.005000 sec

D:\Kuliah\Semester4\STIMA\tucil\tucil2\src>
```


input25.txt
tucil2 > test > input25.txt
IF00001.
IF00002.
IF00003, IF00001, IF00002.
IF00004, IF00001, IF00002.
IF00005, IF00001, IF00002.
IF00006, IF00001, IF00002, IF00003, IF00004, IF00005.
IF00007, IF00001, IF00002, IF00003, IF00004, IF00005.
IF00008, IF00001, IF00002, IF00003, IF00004, IF00005.
IF00009, IF00001, IF00002, IF00003, IF00004, IF00005.
IF00010, IF00001, IF00002, IF00003, IF00004, IF00005.
IF00011, IF00001, IF00002, IF00003, IF00004, IF00005.
IF00012, IF00001, IF00002, IF00003, IF00004, IF00005.
IF00013, IF00001, IF00002, IF00003, IF00004, IF00005.
IF00014, IF00001, IF00002, IF00003, IF00004, IF00005.
IF00015, IF00001, IF00002, IF00003, IF00004, IF00005.
IF00016, IF00001, IF00002, IF00003, IF00004, IF00005.
IF00017, IF00001, IF00002, IF00003, IF00004, IF00005.
IF00018, IF00001, IF00002, IF00003, IF00004, IF00005.
IF00019, IF00001, IF00002, IF00003, IF00004, IF00005, IF0
IF00020, IF00001, IF00002, IF00003, IF00004, IF00005, IF0
IF00021, IF00001, IF00002, IF00003, IF00004, IF00005, IF0
IF00022, IF00001, IF00002, IF00003, IF00004, IF00005, IF0
IF00023, IF00001, IF00002, IF00003, IF00004, IF00005, IF0
IF00024, IF00001, IF00002, IF00003, IF00004, IF00005, IF0
IF00025, IF00001, IF00002, IF00003, IF00004, IF00005, IF0
output.txt
tucil2 > test > output.txt
Semester I: IF00001, IF00002
Semester II: IF00003, IF00004, IF00005
Semester III: IF00006, IF00007, IF00008, IF00009, IF00010, IF00011,
Semester IV: IF00019, IF00020, IF00021, IF00022
Semester V: IF00023, IF00024, IF00025
C:\Windows\System32\cmd.exe
D:\Kuliah\Semester4\STIMA\tucil\tucil2\src>DAGgenerator
Masukkan kode jurusan: IF
Masukkan density(<=100000): 100000
Masukkan nbCourse: 25
Masukkan maxSemester (<99): 5
Masukkan pseudorRandomizer(relatif prima dengan 10): 1
D:\Kuliah\Semester4\STIMA\tucil\tucil2\src>toposort
Masukkan namafile: input25.txt
Time taken by program is : 0.001000 sec
D:\Kuliah\Semester4\STIMA\tucil\tucil2\src>


input100.txt
tucil2 > test > input100.txt
MA71606, MA04321, MA08642, MA12963, MA17284, MA216
MA72223, MA04321, MA08642, MA12963, MA17284, MA216
MA72840, MA04321, MA08642, MA12963, MA17284, MA216
MA73457.
MA75927, MA04321, MA08642, MA12963, MA17284, MA216
MA76544, MA04321, MA08642, MA12963, MA17284, MA216
MA77161, MA04321, MA08642, MA12963, MA17284, MA216
MA77778, MA04321, MA08642, MA12963, MA17284, MA216
MA80248, MA04321, MA08642, MA12963, MA17284, MA216
MA80865, MA04321, MA08642, MA12963, MA17284, MA216
MA81482, MA04321, MA08642, MA12963, MA17284, MA216
MA82099, MA04321, MA08642, MA12963, MA17284, MA216
MA84569, MA04321, MA08642, MA12963, MA17284, MA216
MA85186, MA04321, MA08642, MA12963, MA17284, MA216
MA85803, MA04321, MA08642, MA12963, MA17284, MA216
MA86420, MA04321, MA08642, MA12963, MA17284, MA216
MA88890, MA04321, MA08642, MA12963, MA17284, MA216
MA89507, MA04321, MA08642, MA12963, MA17284, MA216
MA90124, MA04321, MA08642, MA12963, MA17284, MA216
MA90741, MA04321, MA08642, MA12963, MA17284, MA216
MA93211, MA04321, MA08642, MA12963, MA17284, MA216
MA93828, MA04321, MA08642, MA12963, MA17284, MA216
MA94445, MA04321, MA08642, MA12963, MA17284, MA216
MA95062, MA04321, MA08642, MA12963, MA17284, MA216
MA97532, MA04321, MA08642, MA12963, MA17284, MA216
MA98149, MA04321, MA08642, MA12963, MA17284, MA216
MA98766, MA04321, MA08642, MA12963, MA17284, MA216
MA99383, MA04321, MA08642, MA12963, MA17284, MA216
output.txt
tucil2 > test > output.txt
Semester I: MA04321, MA08642, MA12963, MA17284, MA21605, MA25926, MA
Semester II: MA03704, MA08025, MA12346, MA16667, MA20988, MA25309, M
Semester III: MA72840
Semester IV: MA03087, MA07408, MA11729, MA16050, MA20371, MA24692, M
Semester V: MA01853, MA02470, MA06174, MA06791, MA10495, MA11112, MA
C:\Windows\System32\cmd.exe
D:\Kuliah\Semester4\STIMA\tucil\tucil2\src>DAGgenerator
Masukkan kode jurusan: MA
Masukkan density(<=100000): 75000
Masukkan nbCourse: 100
Masukkan maxSemester (<99): 5
Masukkan pseudorRandomizer(relatif prima dengan 10): 4321
D:\Kuliah\Semester4\STIMA\tucil\tucil2\src>toposort
Masukkan namafile: input100.txt
Time taken by program is : 0.004000 sec
D:\Kuliah\Semester4\STIMA\tucil\tucil2\src>


input1000.txt
tucil2 > test > input1000.txt
EL94690, EL05273, EL10546, EL15819, EL21092, EL26
EL94877, EL05273, EL10546, EL15819, EL21092, EL26
EL94914.
EL95064, EL05273, EL10546, EL15819, EL21092, EL26
EL95101.
EL95251, EL05273, EL10546, EL15819, EL21092, EL26
EL95288.
EL95438, EL05273, EL10546, EL15819, EL21092, EL26
EL95475.
EL95625, EL05273, EL10546, EL15819, EL21092, EL26
EL95662.
EL95812, EL05273, EL10546, EL15819, EL21092, EL26
EL95849.
EL95999, EL05273, EL10546, EL15819, EL21092, EL26
EL96036.
EL96186, EL05273, EL10546, EL15819, EL21092, EL26
EL96223.
EL96373, EL05273, EL10546, EL15819, EL21092, EL26
EL96410.
EL96560, EL05273, EL10546, EL15819, EL21092, EL26
EL96597.
EL96747, EL05273, EL10546, EL15819, EL21092, EL26
EL96784, EL05273, EL10546, EL15819, EL21092, EL26
EL96934, EL05273, EL10546, EL15819, EL21092, EL26
output.txt
tucil2 > test > output.txt
Semester I: EL00187, EL00374, EL00561, EL00748, EL00935, EL01122, EL01
Semester II: EL01870, EL02057, EL02244, EL02431, EL02618, EL02805, EL0
Semester III: EL02992, EL03179, EL03366, EL03553, EL03740, EL03927, EL
Semester IV: EL04862, EL05049, EL10135, EL10322, EL15408, EL15595, EL2
Semester V: EL00150, EL00337, EL00524, EL00711, EL05236, EL05423, EL05
Semester VI: EL00898, EL01085, EL06171, EL06358, EL11444, EL11631, EL1
Semester VII: EL01272, EL01459, EL01646, EL01833, EL02020, EL02207, EL
Semester VIII: EL03703, EL03890, EL04077, EL04264, EL04451, EL08976, E
C:\Windows\System32\cmd.exe
D:\Kuliah\Semester4\STIMA\tucil\tucil2\src>DAGgenerator
Masukkan kode jurusan: EL
Masukkan density(<=100000): 60000
Masukkan nbCourse: 1000
Masukkan maxSemester (<99): 8
Masukkan pseudorRandomizer(relatif prima dengan 10): 5273
D:\Kuliah\Semester4\STIMA\tucil\tucil2\src>toposort
Masukkan namafile: input1000.txt
Time taken by program is : 0.421000 sec
D:\Kuliah\Semester4\STIMA\tucil\tucil2\src>


input10000.txt
tucil2 > test > input10000.txt
KU99852, KU10758, KU74434, KU99155, KU
KU99857, KU30961, KU03912, KU91129, KU
KU99862, KU51871, KU26879, KU55985, KU
KU99884, KU53423, KU09679, KU00707, KU
KU99889, KU85059, KU61348, KU59594, KU
KU99894, KU32715, KU50893, KU88296, KU
KU99916, KU30860, KU83305, KU21888, KU
KU99921, KU05666, KU07723, KU37908, KU
KU99926, KU21011, KU00808, KU76592, KU
KU99931.
KU99953, KU99224, KU90826, KU77841, KU
KU99958, KU90151, KU18008, KU58244, KU
KU99963, KU96593, KU47082, KU52950, KU
KU99985, KU24317, KU75986, KU02461, KU
KU99990, KU41416, KU19932, KU11736, KU
KU99995, KU60843, KU80876, KU38413, KU
output.txt
tucil2 > test > output.txt
Semester I: KU00032, KU00101, KU00133,
KU00202, KU00303, KU00404, KU00505, KU00606,
KU00707, KU00771, KU00808, KU00909, KU00978,
KU01010, KU01079, KU01111, KU01180, KU01281,
KU01382, KU01483, KU01547, KU01584, KU01648,
KU01685, KU01786, KU01887, KU01956, KU01988,
KU02057, KU02121, KU02158, KU02259, KU02360,
KU02461, KU02562, KU02663, KU02764, KU02833,
KU02865, KU02934, KU02966, KU03035, KU03136,
KU03237, KU03338, KU03439, KU03540, KU03641,
KU03742, KU03811, KU03843, KU03912, KU04013,
KU04114, KU04215, KU04316, KU04417, KU04518,
KU04619, KU04688, KU04720, KU04789, KU04821,
KU04890, KU04991, KU05092, KU05193, KU05294,
KU05395, KU05459, KU05496, KU05597, KU05666,
KU05698, KU05767, KU05868, KU05969, KU06070,
KU06171, KU06272, KU06373, KU06474, KU06543,
C:\Windows\System32\cmd.exe
D:\Kuliah\Semester4\STIMA\tucil\tucil2\src>DAGgenerator
Masukkan kode jurusan: KU
Masukkan density(<=100000): 100
Masukkan nbCourse: 10000
Masukkan maxSemester (<99): 10
Masukkan pseudorRandomizer(relatif prima dengan 10): 6543
D:\Kuliah\Semester4\STIMA\tucil\tucil2\src>toposort
Masukkan namafile: input10000.txt
Time taken by program is : 0.210000 sec



D.Alamat Drive

https://drive.google.com/drive/folders/11ujwLwBe8aRZPMXDNL8mpoIqBsVb5NvO?usp=sharing

E. Tabel Ceklist

Poin	Ya	Tidak
Program berhasil dikompilasi tanpa kesalahan	√	
Program berhasil running	√	
Program dapat membaca berkas input dan menuliskan output	√	
Luaran sudah benar untuk semua kasus input	√	