

Tugas Kecil 2 IF2121 Strategi Algoritma

Penyusunan Rencana Kuliah dengan Topological Sort (Penerapan Decrease and Conquer)



Nama: Leonard Matheus

NIM: 13519215

**PROGRAM STUDI TEKNIK INFORMATIKA
SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG**

2021

ALGORITMA TOPOLOGICAL SORT DAN KAITANNYA DENGAN ALGORITMA DIVIDE AND CONQUER

Algoritma Divide and Conquer adalah metode perancangan algoritma dengan mereduksi persoalan menjadi dua upa-persoalan (sub-problem) yang lebih kecil, tetapi selanjutnya hanya memproses satu sub-persoalan saja. Topological sorting secara umum adalah metode yang bertujuan untuk mengurutkan simpul-simpul pada suatu graf dari simpul yang paling penting atau diutamakan sampai simpul yang paling dapat diabaikan atau prioritas terakhir.

Akan tetapi, karena adanya batasan bahwa maksimal penyusunan rencana studi hingga 8 semester saja, maka harus ada penyesuaian terhadap pengolahan graf yang ada. Adapun penjelasan algoritma Topological Sort yang digunakan adalah sebagai berikut:

1. Buatlah suatu Matriks Ordering kosong yang diinisiasi dengan Nil sesuai ukuran matriks yaitu jumlah node x jumlah node (persegi)
2. Buatlah suatu array of boolean yang memiliki jumlah elemen sesuai dengan jumlah node pada graf dan diinisiasi dengan nilai False.
3. Inisiasi nilai i dan j sebesar nilai maksimal pada ukuran matriks yang ada. Hal ini berguna agar mata kuliah yang memiliki banyak prasyarat akan ditempatkan di akhir.
4. Untuk basis yang digunakan, carilah node dimana node belum dikunjungi sama sekali dan bernilai false.
5. Apabila ditemukan node yang belum pernah dikunjungi, panggilan fungsi dfs supaya bisa masuk ke dalam bagian rekursif.
6. Pada saat masuk, node yang dikunjungi langsung di-set True dan terus menjalankan operasi.
7. Pada rekursif kedua, sama dengan langkah 4, hanya yang berbeda, node asal yang ditentukan adalah node yang menembakkan panah ke node yang dituju selanjutnya.
8. Ulangi langkah ini terus menerus hingga seluruh node pada 1 operasi dikunjungi.
9. Mulailah operasi pengisian Matriks Ordering hingga yang terakhir dimasukkan adalah node yang paling awal menjadi prasyaratnya, kurangi angka i sebanyak 1 agar bisa diisi menaik.
10. Setelah selesai, kembalikan nilai j menjadi berkurang satu yang artinya berpindah kolom. Inisiasi nilai i kembali sebesar ukuran jumlah node pada grafik agar bisa melakukan pengisian kedua, dst.
11. Langkah 10 berguna untuk memampatkan mata kuliah yang tidak ada kaitannya dapat diambil di semester awal dan mengefisienkan masa kuliah agar tidak lebih dari 8 semester.

SOURCE CODE (C++)

13519215-Graph.hpp

```
1  /*File Graph
2  Nama : Leonard Matheus
3  NIM : 13519215
4  Kelas : K-04
5  */
6  #ifndef GRAPH_HPP
7  #define GRAPH_HPP
8
9  #include <iostream>
10 #include <map>
11 #include <fstream>
12 #include <vector>
13 #include <string>
14
15 using namespace std;
16
17 #define BrsMin 0
18 #define BrsMax 99
19 #define KolMin 0
20 #define KolMax 99
21
22 typedef int indeks; /* indeks baris, kolom */
23 typedef int ElType;
24 typedef struct {
25     ElType Mem[BrsMax+1][KolMax+1];
26     int NBrEff; /* banyaknya/ukuran baris yg terdefinisi */
27     int NKolEff; /* banyaknya/ukuran kolom yg terdefinisi */
28 } Graph;
29
30 #define NBrEff(G) (G).NBrEff
31 #define NKolEff(G) (G).NKolEff
32 #define Elmt(G,i,j) (G).Mem[(i)][(j)]
33
```

```

34 //Membaca input file dan membuat vektor string untuk dibuatkan mapnya
35 vector<string> bacaGraph(string &filePath, Graph * G);
36 //Mengekstraksi list mata kuliah yang dibentuk menjadi matriks graph
37 void makeGraph(Graph * G, map<string,int> list_matkul, vector<string> inputmatkul);
38 //Menampilkan matriks graph ke layar
39 void TulisGraph (Graph * G);
40 //Getter Indeks Pertama Baris Matriks Adjacency List
41 indeks GetFirstIdxBrs (Graph G);
42 //Getter Indeks Pertama Kolom Matriks Adjacency List
43 indeks GetFirstIdxKol (Graph G);
44 //Getter Indeks Terakhir Baris Matriks Adjacency List
45 indeks GetLastIdxBrs (Graph G);
46 //Getter Indeks Terakhir Kolom Matriks Adjacency List
47 indeks GetLastIdxKol (Graph G);
48 //Getter jumlah node yang ada di graph
49 int GetnumberOfNodes(Graph G);
50 //Membuat map antara nama matkulnya
51 map<string, int> matkul (vector<string> senaraimatkul);
52 //Untuk mendapatkan key dari indeks integer matkul yang ditanya
53 string key_matkulnya_apa (map<string, int> matakuliah, int valuenya);
54
55 #endif

```

13519215-Graph.cpp

```
1  ▾ /*File Graph
2     Nama : Leonard Matheus
3     NIM : 13519215
4     Kelas : K-04
5     */
6
7  ▾ #include <iostream>
8     #include <fstream>
9     #include <string>
10    #include <vector>
11    #include <map>
12    #include "13519215-Graph.hpp"
13
14    using namespace std;
15
16    //Membaca input file dan membuat vektor string untuk dibuatkan mapnya
17  ▾ vector<string> bacaGraph(string &filePath, Graph * G) {
18      //merujuk file input
19      ifstream in(filePath);
20      char c;
21      vector<string> bahan;
22
23      int jumlahkata = 0;
24      int i = 0;
25      //Jika file terbuka, maka ...
26      if(in.is_open()) {
27          //Ketika file terbuka, maka ...
28          while(in.good()) {
29              string tp;
30              while(getline(in, tp)){ //membaca dari objek file dan menyimpan dalam string.
31                  //Inisiasi filter bila ada koma dan titik
32                  string delimiter = ", ";
33                  string titik = ".";
34                  size_t pos = 0;
35                  string token;
36                  while ((pos = tp.find(delimiter)) != string::npos || (pos = tp.find(titik)) != string::npos) {
37                      //ekstraksi token dari awal sampai ditemukan koma atau titik
38                      token = tp.substr(0, pos);
39                      //Push back ke loyang kita yang berupa vektor string
40                      bahan.push_back(token);
41                      //Kalau ketemu delimiternya, hapus sepanjang panjang demiliter
42                      if(tp.find(delimiter)){
43                          tp.erase(0, pos + delimiter.length());
44                      }else if (tp.find(titik)){
45                          tp.erase(0, pos + titik.length());
46                      }
47                  }
48                  //masukkan enter apabila berbeda baris
49                  bahan.push_back("\n");
50              }
51          }
52      }
53
54      //Error saat membaca file
55      if(!in.eof() && in.fail())
56          cout << "Error saat membuka " << filePath << endl;
57
58      //Tutup File
59      in.close();
```

```

60
61     return bahan;
62 }
63
64 //Mengekstraksi list mata kuliah yang dibentuk menjadi matriks graph
65 void makeGraph(Graph * G, map<string,int> list_matkul, vector<string> inputmatkul){
66     //inisiasi baris efektif dan kolom efektif dari list_matkul yang dibaca dari fungsi bacagraph
67     NBrseff(*G)=list_matkul.size();
68     NKoleff(*G)=list_matkul.size();
69     int NB, NK = list_matkul.size();
70     int valuekiri, valuekanan = 0;
71     indeks i,j,k;
72     //Isi matriks dengan 0 semuanya
73     for (i = BrsMin; i < NB; i++){
74         for (j = KolMin; j < NK; j++){
75             Elmt(*G, i, j) = 0;
76         }
77     }
78     string kanan = "";
79     string kiri = "";
80     for (k = 0; k<(inputmatkul.size()); k++){
81         //Bila string kiri tidak ada isi, langsung jadikan sebagai graph dependencies dan masukkan ke graph
82         if (kiri == "" || inputmatkul.at(k-1)=="\n"){
83             kiri = inputmatkul.at(k);
84         }
85         //Isi matkul di kanannya
86         kanan = inputmatkul.at(k);
87         if (kiri != kanan && kanan != "\n"){
88             valuekiri = list_matkul.at(kiri);
89             valuekanan = list_matkul.at(kanan);
90             //Apabila sudah didapatkan pasangan simpul graph yang tepat yaitu valuekanan dan kirinya, masukkan ke matriks graph
91             Elmt(*G, valuekanan, valuekiri) = 1;
92         }
93     }
94 }
95

```

```

96 //Menampilkan matriks graph ke layar
97 void TulisGraph (Graph *G){
98     indeks i,j;
99     for (i = GetFirstIdxBrs(*G); i <= GetLastIdxBrs(*G); i++){
100         for (j = GetFirstIdxKol(*G); j <= GetLastIdxKol(*G); j++){
101             cout << Elmt(*G,i,j);
102             if (j!=GetLastIdxKol(*G)){
103                 cout << " ";
104             }
105         }
106         //Apabila sudah di akhir baris, print newline
107         if (i!=GetLastIdxBrs(*G)){
108             cout << endl;
109         }
110     }
111 }
112

```

```

113 //Getter Indeks Pertama Baris Matriks Adjacency List
114 indeks GetFirstIdxBrs (Graph G){
115     return (BrsMin);
116 }
117
118 //Getter Indeks Pertama Kolom Matriks Adjacency List
119 indeks GetFirstIdxKol (Graph G){
120     return (KolMin);
121 }
122
123 //Getter Indeks Terakhir Baris Matriks Adjacency List
124 indeks GetLastIdxBrs (Graph G){
125     return(NBrsEff(G)-1+BrsMin);
126 }
127
128 //Getter Indeks Terakhir Kolom Matriks Adjacency List
129 indeks GetLastIdxKol (Graph G){
130     return(NKolEff(G)-1+KolMin);
131 }
132

```

```

133 //Getter jumlah node yang ada di graph
134 int GetnumberOfNodes(Graph G){
135     return NBrseff(G) - BrsMin;
136 }
137
138 //Membuat map antara nama matkulnya
139 map<string, int> matkul (vector<string> senaraimatkul){
140     map<string, int> dict;
141     int indeks = 0;
142     for (int i=0; i<senaraimatkul.size(); i++){
143         //Jika jumlah elemen senarai mata kuliah pada suatu key masih nol dan elemen bukan baris baru
144         if (dict.count(senaraimatkul.at(i))<=0 && senaraimatkul.at(i)!="\n"){
145             //Insert elemen senarai mata kuliah pada map dict untuk key baru berupa integer
146             dict.insert(make_pair(senaraimatkul.at(i), indeks));
147             indeks++;
148         }
149     }
150     return dict;
151 }
152
153 //Untuk mendapatkan key dari indeks integer matkul yang ditanya
154 string key_matkulnya_apa (map<string, int> matakuliah, int valuenya){
155     string key = "";
156     //Auto iterasi pada mapnya
157     for (auto &i : matakuliah) {
158         if (i.second == valuenya) {
159             key = i.first;
160             break;
161         }
162     }
163     return key;
164 }

```

13519215-Topological.hpp

```
1  /*File Topological
2  Nama : Leonard Matheus
3  NIM : 13519215
4  Kelas : K-04
5  */
6
7  #ifndef TOPOLOGICAL_HPP
8  #define TOPOLOGICAL_HPP
9
10 #include "13519215-Graph.hpp"
11 #include <vector>
12
13 //Loyang untuk Toposort yang berisi matriks vector of integer ordering dan array of boolean V
14 typedef struct {
15     vector<vector<int>> ordering;
16     vector<bool> V;
17 } Toposort;
18
19 //Macro untuk mengakses elemen dalam struct
20 #define Ordering(T) (T).ordering
21 #define V(T) (T).V
22
23 //Definisi Nil
24 #define Nil 99
25
26 //Memproses Graph yang terbentuk dan mengembalikan matriks vektor integer yang sudah terurut
27 void topsort(Toposort *T, Graph *G);
28 //Membantu topsort dalam melakukan rekursif, menerima integer i,j dan node yang dituju
29 int dfs(int i, int j, int at, Toposort *T, Graph *G);
30 //Merapihkan matriks vektor integer sehingga tidak ada mata kuliah yang tumpang tindih di semester lain
31 void sikat (Toposort *T);
32 //Mengubah integer menjadi string menurut map of pair(string, key) yang telah didefinisikan di awal
33 vector<vector<string>> terjemahkan_key (Toposort *T, map<string,int> matkul);
34 //Menampilkan Mata Kuliah yang disarankan di layar
35 void printSemester(vector<vector<string>> hasil);
36 //Menunjukkan pasangan mata kuliah dengan representasi integer yang dimasukkan ke dalam suatu map
37
38 void show_Matkul_key(map<string,int> mapnya);
39 //Mengecek apakah pada suatu baris/kolom terdapat nilai Nil semuanya secara simultan
40 bool cek_nil (vector<int> produk);
41
42 #endif
```


13519215-Topological.cpp

```
1  /*File Topological
2  Nama : Leonard Matheus
3  NIM : 13519215
4  Kelas : K-04
5  */
6  #include <iostream>
7  #include <vector>
8  #include <string>
9  #include "13519215-Topological.hpp"
10 #include "13519215-Graph.hpp"
11
12 using namespace std;
13
14 //Memproses Graph yang terbentuk dan mengembalikan matriks vektor integer yang sudah terurut
15 void topsort(Toposort *T, Graph *G){
16     int N = GetNumberOfNodes(*G); //Mendapatkan jumlah node pada grafik
17     for (int a = 0; a < N; a++){
18         Ordering(*T).push_back(vector<int>()); //Menyiapkan loyang untuk suatu baris vektor integer
19         V(*T).push_back(false); //Mendefinisikan seluruh node berstatus false
20         for(int b = 0; b < N; b++){
21             //Menginisiasi semua nilai dalam matriks vektor integer menjadi Nil
22             Ordering(*T).at(a).push_back(0);
23         }
24     }
25     //Inisiasi nilai indeks i dan j yang akan diproses
26     int i = N - 1;
27     int j = N - 1;
28
29     for (int at = 0; at < N; at++){
30         //Mengecek apakah node yang akan dikunjungi bernilai false
31         if (V(*T).at(at) == false){
32             //Masuk ke dalam dfs secara rekursif
33             i = dfs(i, j, at, T, G);
34         }
35         //Menyimpan di indeks terpisah setelah dfs selesai dipanggil
36         j -= 1;
37         i = N-1;
38     }
39 }
40
41 //Membantu topsort dalam melakukan rekursif, menerima integer i,j dan node yang dituju
42 int dfs(int i, int j, int at, Toposort *T, Graph *G){
43     //Saat node sudah dikunjungi, set status ke true
44     V(*T).at(at) = true;
45     for (int b = 0; b < (V(*T).size()); b++){ //mencari node yang dituju pada graph
46         if (Elmt(*G,b,at) == 1){
47             // Jika ada node yang dituju, kunjungi saja selama belum tersentuh
48             if(V(*T).at(b) == false){
49                 //Rekursif lagi dong
50                 i = dfs(i, j, b, T, G);
51             }
52         }
53     }
54     //Masukkan ke vektor integer ordering
55     Ordering(*T).at(i).at(j) = at;
56     //Kurangi i sebesar 1
57     return i-1;
58 }
```

```

60 //Merapihkan matriks vektor integer sehingga tidak ada mata kuliah yang tumpang tindih di semester lain
61 void sikat (Toposort *T){
62     vector<vector<int>> produk;
63     vector<vector<int>> produk2;
64     for (int i = Ordering(*T).size()-1; i >= 0 ; i--){
65         //Menyiapkan vektor integer baru untuk membalik segala matriks yang tadi diinput terbalik di topsort
66         produk.push_back(vector<int>());
67         if (cek_nil(Ordering(*T).at(i))){
68             produk.pop_back();
69         }else{
70             for (int j = Ordering(*T).size()-1; j >= 0; j--){
71                 produk.at(produk.size()-1).push_back(Ordering(*T).at(i).at(j));
72             }
73         }
74     }
75
76     //Isi semua vektor produk2 dengan Nil
77     for (int i = 0; i < produk.size();i++){
78         produk2.push_back(vector<int>());
79         for (int j = 0; j < produk.at(i).size();j++){
80             produk2.at(i).push_back(0);
81         }
82     }
83
84     //Mampatkan isi produk2 dengan produk (ditempel ke atas)
85     for (int j = 0; j < produk.at(0).size();j++){
86         int k = 0;
87         for (int i = 0; i < produk.size();i++){
88             if (produk.at(i).at(j)!=0){
89                 produk2.at(k).at(j) = produk.at(i).at(j);
90                 k++;
91             }
92         }
93     }
94 }

```

```

94
95 //Ordering diset menjadi matriks vektor integer dari produk2
96 Ordering(*T) = produk2;
97 }
98
99 //Mengubah integer menjadi string menurut map of pair(string, key) yang telah didefinisikan di awal
100 vector<vector<string>> terjemahkan_key (Toposort *T, map<string,int> matkul){
101     vector<vector<string>> hasil;
102     for (int i= 0; i <= Ordering(*T).size()-1; i++){
103         hasil.push_back(vector<string>());
104         for (int j = 0; j <= Ordering(*T).at(i).size()-1; j++){
105             if (Ordering(*T).at(i).at(j)==0){
106                 //bila bernilai 0, masukkan spasi
107                 hasil.at(hasil.size()-1).push_back("");
108             }else if (Ordering(*T).at(i).at(j)!=0){
109                 //Masukan hasil Ordering untuk dikonversi menjadi string matakuliah
110                 hasil.at(hasil.size()-1).push_back(key_matkulnya_apa(matkul, Ordering(*T).at(i).at(j)));
111             }
112         }
113     }
114     return hasil;
115 }
116

```

```

117 //Menampilkan Mata Kuliah yang disarankan di layar
118 void printSemester(vector<vector<string>> hasil){
119     if (hasil.size()>8){ //Maksimal semester 8 ya
120         cout << "Maaf, Penyusunan mata kuliah dibatasi hanya untuk 8 semester saja!" << endl;;
121         cout << "Silahkan kurangi matakuliah pada prereq.txt" << endl;
122     }else{
123         //Mekanisme print ke layar
124         cout << "<<Saran Pengambilan Mata Kuliah>>" << endl;
125         for (int i=0; i< hasil.size(); i++){
126             cout << "Semester "<<i+1<< " : ";
127             for(int j=0; j <= hasil.at(i).size()-1; j++){
128                 if (hasil.at(i).at(j) == ""){
129                     cout<<"";
130                 }else if (j == hasil.at(i).size()-1){
131                     cout << hasil.at(i).at(j);
132                 }else{
133                     cout << hasil.at(i).at(j) << ", ";
134                 }
135             }
136             cout << endl;
137         }
138     }
139 }
140
141 //Menunjukkan pasangan mata kuliah dengan representasi integer yang dimasukkan ke dalam suatu map
142 void show_Matkul_key(map<string,int> mapnya){
143     for(auto it = mapnya.cbegin(); it != mapnya.cend(); ++it){
144         cout << it->first << " " << it->second << endl;
145     }
146 }

```

```

148 //Mengecek apakah pada suatu baris/kolom terdapat nilai Nil semuanya secara simultan
149 bool cek_nil (vector<int> produk){
150     bool found = true;
151     int i = 0;
152     while (i < produk.size() && found){
153         if (produk.at(i) != Nil){
154             found = false;
155         }
156         i++;
157     }
158     return found;
159 }

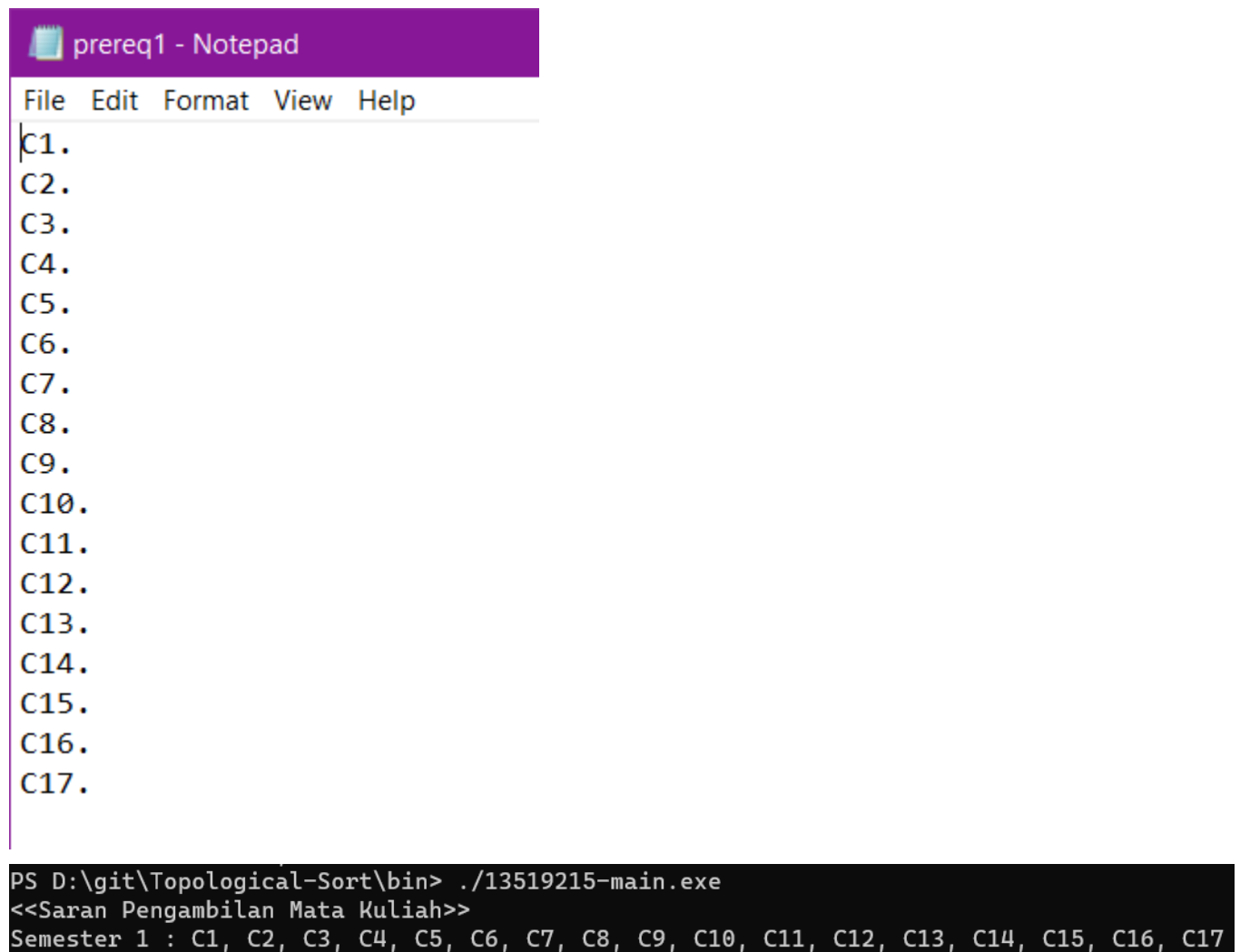
```

13519215-main.cpp

```
1  /*File Main
2  Nama : Leonard Matheus
3  NIM : 13519215
4  Kelas : K-04
5  */
6  #include <iostream>
7  #include <vector>
8  #include <string>
9  #include "13519215-Topological.cpp"
10 #include "13519215-Graph.cpp"
11
12 using namespace std;
13
14 int main(int argc, char const *argv[]) {
15     Graph G;
16     Toposort T;
17     vector<vector<string>> hasil;
18     string a = "../test/prereq.txt";
19     vector<string> wow = bacaGraph(a, &G);
20     map<string,int> mapnya = matkul(wow);
21     makeGraph(&G, mapnya, wow);
22     //show_Matkul_key(mapnya);
23     topsort(&T, &G);
24     sikat(&T);
25     hasil = terjemahkan_key(&T, mapnya);
26     printSemester(hasil);
27
28     return 0;
29 }
30
```

HASIL KOMPILASI DAN TESTING

Hasil Kompilasi uji dapat dilihat pada gambar di bawah ini:



The image shows two windows. The top window is a Notepad application titled 'prereq1 - Notepad'. It contains a list of course prerequisites from C1 to C17. The bottom window is a terminal window showing the execution of a program named '13519215-main.exe'. The program outputs a list of recommended courses for Semester 1, which matches the list in the Notepad window.

```
prereq1 - Notepad
File Edit Format View Help
C1.
C2.
C3.
C4.
C5.
C6.
C7.
C8.
C9.
C10.
C11.
C12.
C13.
C14.
C15.
C16.
C17.

PS D:\git\Topological-Sort\bin> ./13519215-main.exe
<<Saran Pengambilan Mata Kuliah>>
Semester 1 : C1, C2, C3, C4, C5, C6, C7, C8, C9, C10, C11, C12, C13, C14, C15, C16, C17
```

prereq2 - Notepad

File Edit Format View Help

IPS.
PKN.
Nuklir, Fisika, IPA, Matematika.
Statistika, Matematika.
Fisika, IPA, Matematika.
IPA, Matematika.
Kimia, Fisika, Biologi, IPA.
Biologi, IPA.

```
PS D:\git\Topological-Sort\bin> ./13519215-main.exe
<<Saran Pengambilan Mata Kuliah>>
Semester 1 : IPS, PKN, Matematika, Statistika, Biologi,
Semester 2 : IPA, Kimia,
Semester 3 : Fisika,
Semester 4 : Nuklir,
```

prereq3 - Notepad

File Edit Format View Help

MA1101.
MA1201.
KU1211.
IF1110.
IF2112, IF2111.
IF2110, KU1210, IF1110.
IF2111, IF2110.

```
PS D:\git\Topological-Sort\bin> ./13519215-main.exe
<<Saran Pengambilan Mata Kuliah>>
Semester 1 : MA1101, MA1201, KU1211, IF1110, KU1210,
Semester 2 : IF2110,
Semester 3 : IF2111,
Semester 4 : IF2112,
```

```
prereq4 - Notepad
File Edit Format View Help
A, B.
B, C.
C, D.
D, E.
E, F.
F.
```

```
PS D:\git\Topological-Sort\bin> ./13519215-main.exe
<<Saran Pengambilan Mata Kuliah>>
Semester 1 : F,
Semester 2 : E,
Semester 3 : D,
Semester 4 : C,
Semester 5 : B,
Semester 6 : A,
```

```
prereq5 - Notepad
File Edit Format View
b, a.
a.
```

```
PS D:\git\Topological-Sort\bin> ./13519215-main.exe
<<Saran Pengambilan Mata Kuliah>>
Semester 1 : a,
Semester 2 : b,
```

```
prereq6 - Notepad
File Edit Format View Help
C1, C2.
C2, C3.
C3, C4.
C4, C5.
C5, C6.
C6, C7.
C7, C8.
C8, C9.
C9, C10.
C10, C11.
```

```
PS D:\git\Topological-Sort\bin> ./13519215-main.exe
Maaf, Penyusunan mata kuliah dibatasi hanya untuk 8 semester saja!
Silahkan kurangi matakuliah pada prereq.txt
```

```
prereq7 - Notepad
File Edit Format View Help
C13.
C15.
C9.
C1, C2, C13.
C2, C3, C15.
C3, C4, C9.
```

```
PS D:\git\Topological-Sort\bin> ./13519215-main.exe
<<Saran Pengambilan Mata Kuliah>>
Semester 1 : C13, C15, C9, C4,
Semester 2 : C3,
Semester 3 : C2,
Semester 4 : C1,
```




prereq8 - Notepad

File Edit Format View Help

Image_Processing, AI.
AI, Logkom, Algeo, Grafika_Komputasi.
Grafika_Komputasi, Algeo.
BigData, MBD, Basdat.
MBD, Basdat.
Matematika.
Basdat.
Logkom, Matematika, Kalkulus.
Algeo, Kalkulus, Matematika.
Kalkulus, Matematika.

```
PS D:\git\Topological-Sort\bin> ./13519215-main.exe
<<Saran Pengambilan Mata Kuliah>>
Semester 1 : Matematika, Basdat,
Semester 2 : Kalkulus, MBD,
Semester 3 : Logkom, BigData,
Semester 4 : Algeo,
Semester 5 : Grafika_Komputasi,
Semester 6 : AI,
Semester 7 : Image_Processing,
```

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.	✓	

LETAK DRIVE UNTUK DIUJI COBA

<https://github.com/leomatt547/Topological-Sort>

REFERENSI

1. <https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2012-2013/Makalah2012/Makalah-IF3051-2012-060.pdf>
2. <http://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Algoritma-Decrease-and-Conquer-2021-Bagian1.pdf>
3. <https://www.youtube.com/watch?v=eL-KzMXSXXI>