Tugas Kecil 2 IF2211 Strategi Algoritma

Semester II tahun 2020/2021

Penyusunan Rencana Kuliah dengan *Topological Sort*

(Penerapan *Decrease and Conquer*)

Disusun Oleh:

Ryo Richardo / 13519193

Program Studi Teknik Informatika

Sekolah Teknik Elektro dan Informatika

Institut Teknologi Bandung

2021

1. **Algoritma *Topological Sort* dan kaitannya dengan pendekatan *Decrease and Conquer***

   Topological Sort untuk *Directed Acylic Graph* (DAG) adalah proses pengurutan sebuah graf sedimikian sehingga untuk semua sisi u v, simpul u dikunjungi lebih dahulu sebelum simpul v. Algoritma *Topological Sort* sering dikaitkan dengan algoritma *Depth First Search* (DFS) karena berhubungan dengan teori graf. Namun, keduanya memiliki perbedaan dimana DFS dapat selalu mengunjungi simpul yang bersisian dengan simpul sebelumnya, sedangkan *Topological Sort* harus mengunjungi semua simpul yang bersisian dengan simpul yang akan dikunjungi.

   Pada permasalahan penyusunan rencana kuliah, algoritma *Topological Sort* memiliki hubungan dengan algoritma *Decrease and Conquer*. Proses penyusunan rencana kuliah dengan algoritma *Topological Sort* akan diulangi terus-menerus untuk semua mata kuliah dalam sebuah senarai yang ukurannya akan berkurang setiap ada mata kuliah yang diambil. Adapun algoritma penyusunan rencana kuliah dengan *Topological Sort* dapat dijelaskan sebagai berikut:

   i. Menginisialisasi sebuah senarai dan mengisinya dengan daftar mata kuliah serta *prerequisite*-nya.
   ii. Mecari mata kuliah yang tidak memiliki *prerequisite* untuk diambil.
   iii. Menghapus mata kuliah yang akan diambil di poin sebelumnya dari daftar *prerequisite* mata kuliah lain.
   iv. Menghapus mata kuliah yang akan diambil dari senarai.
   v. Ulangi langkah-langkah di atas hingga senarai kosong atau tidak ada mata kuliah lain yang dapat diambil dengan memenuhi *prerequisite*-nya.

2. ***Source program***

```
// RENCANA STUDI ORGANIZER //

// Author: Ryo Richardo //

#include <stdio.h>
#include <stdlib.h>
#include <string.h>

// boolean.h
#define boolean unsigned char
#define true 1
#define false 0
```

```c
// DEKLARASI TIPE PREREQ //
typedef struct prereq *alamat;
typedef struct prereq
{
    char name[50];          // Nama prereq
    alamat next;            // Next element
} prereq;
//////////////////////////


// DEKLARASI TIPE MATKUL //
typedef struct matkul *address;
typedef struct matkul
{
    char name[50];          // Nama matkul
    alamat list_prereq;     // Array prereq
    boolean delete_soon;    // Matkul akan dihapus
    address next;           // Next element
} matkul;

typedef struct {
    address First;
} List;
//////////////////////////


// KONSTRUKTOR DAN DESTRUKTOR PREREQ //
alamat alokasiPrereq(char name[50]){
    // I.S. String nama prereq
    // F.S. Terbentuk prereq P

    alamat P = (alamat) malloc(sizeof(prereq));
    if (P != NULL){
        strcpy(P->name, name);
        P->next = NULL;
    }
    return P;
}

void addPrereq(matkul* M, alamat P){
    // I.S. Matkul M, alamat prereq P
    // F.S. Menambah P menjadi prereq M


    // Jika list prereq masih kosong
    if (M->list_prereq == NULL){
        M->list_prereq = P;
    }
```

```
    // Jika list prereq sudah terisi
    else{
        alamat Last = M->list_prereq;
        while (Last->next != NULL){
            Last = Last->next;
        }
        Last->next = P;
    }
}

void delPrereq(matkul *M, char P[50]){
    // I.S. Matkul M, string P
    // F.S. Prereq pada matkul M dihapus jika bersesuaian dgn P

    // Prekondisi: list prereq ada isinya
    if (M->list_prereq != NULL){
        alamat PQ = M->list_prereq;

        // Menghapus prereq PQ jika elemen pertama
        if (!strcmp(PQ->name, P)){
            M->list_prereq = PQ->next;
            PQ->next = NULL;
            free(PQ);
        }

        // Menghapus prereq PQ jika bukan elemen pertama
        else{
            while (PQ->next != NULL && strcmp(PQ->next->name, P)){
                PQ = PQ->next;
            }

            if (PQ->next != NULL){
                alamat PK = PQ->next;
                PQ->next = PK->next;
                PK->next = NULL;
                free(PK);
            }
        }
    }
}
////////////////////////////////////////////////////////////


// KONSTRUKTOR DAN DESTRUKTOR MATKUL //
void createEmpty(List *L){
    // I.S. List L
    // F.S. L.First menunjuk NULL
    L->First = NULL;
}
```

```
address alokasiMatkul(char name[50]){
    // I.S. String nama matkul
    // F.S. Terbentuk matkul M

    address M = (address) malloc(sizeof(matkul));
    if (M != NULL){
        strcpy(M->name, name);
        M->list_prereq = NULL;
        M->delete_soon = false;
        M->next = NULL;
    }
    return M;
}

void addMatkul(List* L, address M){
    // I.S. List L, alamat matkul M
    // F.S. Menambah M menjadi anggota L

    // Jika L masih kosong
    if (L->First == NULL){
        L->First = M;
    }

    // Jika L sudah terisi
    else{
        address N = L->First;
        while (N->next != NULL){
            N = N->next;
        }
        N->next = M;
    }
}

void delMatkul(List *L, address M){
    // I.S. List matkul L, alamat matkul M
    // F.S. Matkul M dihapus dari list, matkul lain yg prereqnya M dihapus dgn
 prosedur delPrereq

    // Memanggil prosedur delPrereq untuk semua matkul
    address MKb, MK = L->First;
    while (MK != NULL){

        // Menyimpan address matkul sebelum M
        if (MK->next != NULL && !strcmp(MK->next->name, M->name)){
            MKb = MK;
        }
        delPrereq(MK, M->name);
```

```
        MK = MK->next;
    }

    // Menghapus matkul M jika elemen pertama
    if (L->First == M){
        L->First = M->next;
    }
    // Menghapus matkul M jika bukan elemen pertama
    else{
        MKb->next = M->next;
    }
}
////////////////////////////////////////////////////////////////


// FUNGSI MEMBACA FILE //
void bacaFile(List *L, char namafile[100]){
    // I.S. List kosong L, string namafile
    // F.S. Membaca file dan menyalinnya menjadi list L

    // Kamus konstanta
    #define splitter ','
    #define stopper '.'
    #define blank ' '
    #define enter '\n'

    // Membuka file
    FILE *f;
    f = fopen(namafile, "r");
    if (f == NULL){
        printf("Error!");
        exit(1);
    }

    // Membaca file
    char c;
    c = getc(f);

    // EOF
    while (c != EOF){

        // Var count berfungsi membedakan matkul dgn prereqnya
        int count = 0;
        address MK;

        // Membaca tiap matkul
        while (c != stopper && c != EOF){

            // Inisialisasi tempat penyimpanan kata
```

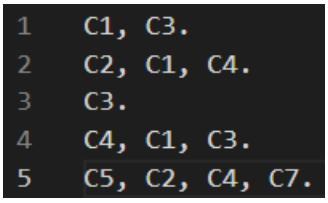
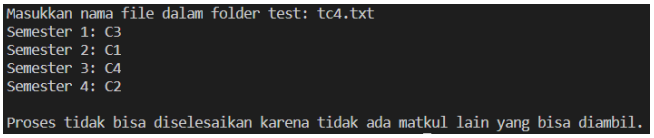
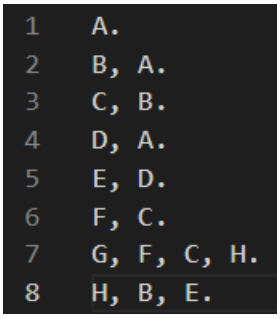
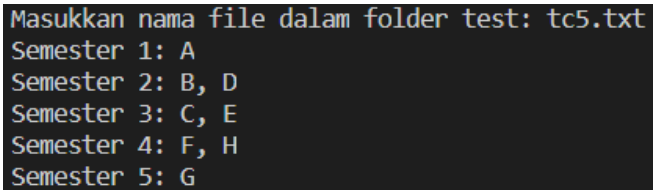
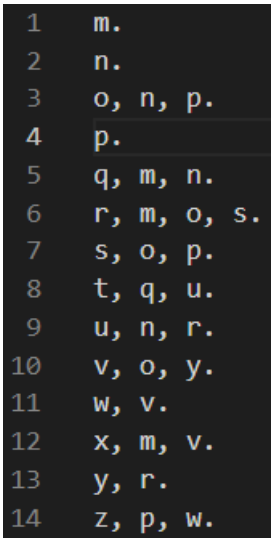
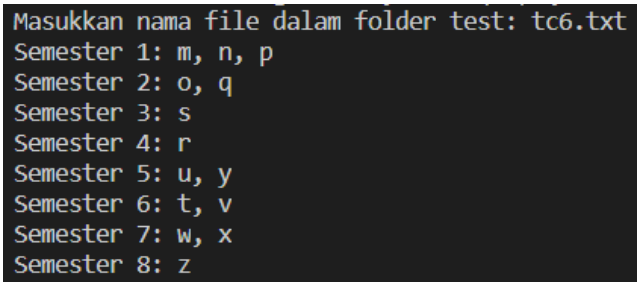
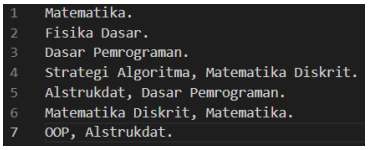
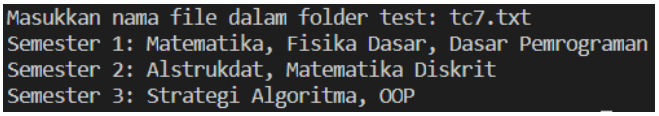
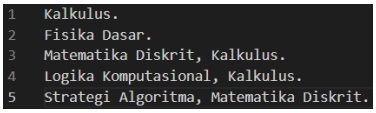
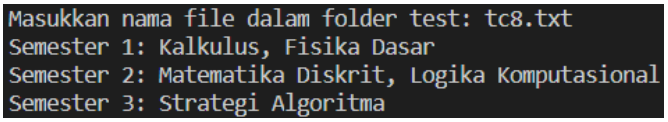
```
            char name[50];
            int i = 0;

            // Membuang blank atau enter
            while (c == blank || c == enter){
                c = getc(f);
            }

            // Membaca dan memisahkan matkul dan prereqnya
            while (c != splitter && c != stopper && c != EOF){
                name[i] = c;
                i++;
                c = getc(f);
            }
            name[i] = '\0';

            // Jika count = 0, maka adalah matkul
            if (count == 0){
                MK = alokasiMatkul(name);
            }
            // Jika count > 0, maka adalah prereq
            else{
                alamat MP = alokasiPrereq(name);
                addPrereq(MK, MP);
            }

            // Next element
            count++;
            if (c == splitter){
                c = getc(f);
            }
        }

        // Menambah MK ke list L
        addMatkul(L, MK);

        // Next element
        if (c == stopper){
            c = getc(f);
        }
    }

    // Jika terjadi kesalahan pada getc()
    if (!feof(f)){
        printf("Terjadi kesalahan pada proses pembacaan file.\n");
    }
    fclose(f);
}
```

```
////////////////////////////////////////////////////////////////////


// MAIN PROGRAM //
int main(){
    // KAMUS DAN DEKLARASI VARIABEL //
    char namafile[100] = "../test/";  // Letak file yang akan dibuka
    char input[50];                   // variabel penyimpanan input
    int i = 1;                        // Counter semester ke-i
    int found = 1;                    // Counter ditemukan matkul tanpa prereq
    List L;                           // List matkul
    address M;                        // Variabel penyimpanan matkul


    // ALGORITMA PROGRAM UTAMA //

    // Inisialisasi dan Input
    printf("Masukkan nama file dalam folder test: ");
    scanf("%s", input);
    strcat(namafile, input);
    createEmpty(&L);
    bacaFile(&L, namafile);

    // Proses sorting
    while (L.First != NULL && found){
        found = 0;
        M = L.First;

        // Mencari matkul yang tidak memiliki prereq untuk dihapus
        while (M != NULL){
            if (M->list_prereq == NULL){

                // Elemen pertama
                if (found == 0){
                    printf("Semester %d: %s", i, M->name);
                }

                // Bukan elemen pertama
                else{
                    printf(", %s", M->name);
                }

                // Menambah matkul ke list matkul yang akan dihapus
                M->delete_soon = true;
                found++;
            }
            M = M->next;
        }
```

```
        // Jika terdapat matkul yang akan dihapus
        if (found){
            M = L.First;

            // Menghapus matkul yang sudah ditandai
            while (M != NULL){
                if (M->delete_soon){
                    delMatkul(&L, M);
                }
                M = M->next;
            }
        }
        printf("\n");
        i++;
    }

    // Jika tidak ada matkul lain yang bisa diambil
    if (!found){
        printf("Proses tidak bisa diselesaikan karena tidak ada matkul lain ya
ng bisa diambil.\n");
    }

    while(1);

    return 0;
}
```

3. Contoh *input* dan *output*

No	Input	Output
1	C1, C3. C2, C1, C4. C3. C4, C1, C3. C5, C2, C4.	Masukkan nama file dalam folder test: tc1.txt Semester 1: C3 Semester 2: C1 Semester 3: C4 Semester 4: C2 Semester 5: C5
2	C1. C2, C1, C4. C3. C4, C1, C3. C5, C2, C4.	Masukkan nama file dalam folder test: tc2.txt Semester 1: C1, C3 Semester 2: C4 Semester 3: C2 Semester 4: C5
3	C1, C3. C2, C1, C4. C3, C1. C4, C1, C3. C5, C2, C4.	Masukkan nama file dalam folder test: tc3.txt Proses tidak bisa diselesaikan karena tidak ada matkul lain yang bisa diambil.

4		
5		
6		
7		
8		

4. Alamat kode sumber

Tucil2_13519193/src/13519193.c

5. Checklist

Poin	Ya	Tidak
1. Program berhasil dikompilasi	✓	
2. Program berhasil *running*	✓	
3. Program dapat menerima berkas input dan menuliskan output.	✓	
4. Luaran sudah benar untuk semua kasus input.	✓	