LAPORAN TUGAS KECIL 2

IF2211 STRATEGI ALGORITMA

SEMESTER II TAHUN 2020/2021

PENYUSUNAN RENCANA KULIAH DENGAN *TOPOLOGICAL SORT*

Disusun oleh:

Ruhiyah Faradishi Widiaputri

13519034

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

2021

DAFTAR ISI

BAB I

TOPOLOGICAL SORT DAN KAITANNYA DENGAN PENDEKATAN *DECREASE AND CONQUER*

A. *Topological Sort*

Topological sort adalah pengurutan yang dilakukan terhadap simpul-simpul dari sebuah graf berarah sedemikian sehingga untuk setiap busur uv dari simpul u ke simpul v dengan *topological sort* ini menghasilkan u lebih dulu daripada v. *Topological sort* ini hanya dapat dilakukan untuk graf yang tidak mempunyai siklus berarah. Dengan kata lain *topological sort* hanya dapat diterapkan untuk graf berarah asiklik / *directed acyclic graph* (DAG).

Menurut Levitin(2012) ada dua algoritma efisien yang dapat digunakan untuk menghasilkan urutan simpul-simpul yang menyelesaikan permasalahan *topological sort*. Algoritma pertama adalah aplikasi sederhana dari algoritma *depth first search*, yaitu dengan melakukan DFS secara traversal. Algoritma kedua didasarkan dari implementasi dari algoritma *decrease and conquer*.

B. Algoritma *Decrease and Conquer*

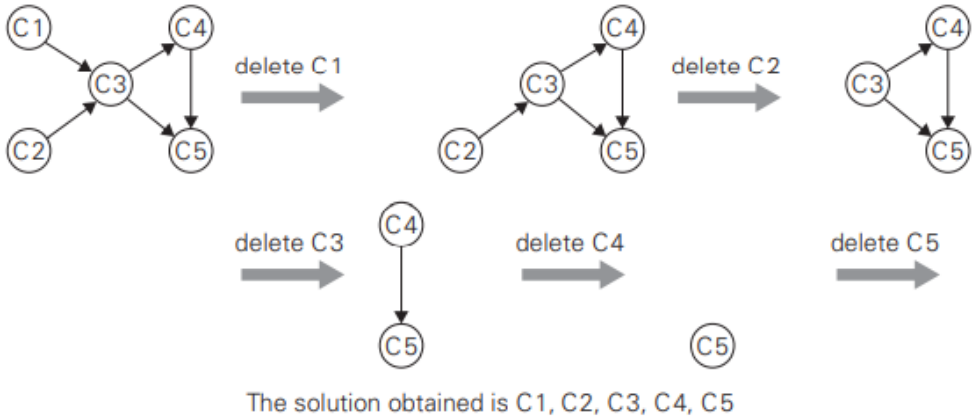
Decrease and conquer merupakan metode perancangan algoritma dengan mereduksi persoalan menjadi beberapa upa-persoalan (*sub-problem*) yang lebih kecil (biasanya dua sub-persoalan), tetapi selanjutnya hanya memproses satu sub-persoalan saja. Algoritma ini memiliki sedikit perbedaan dengan algoritma *divide and conquer*, yaitu algoritma *divide and conquer* memproses semua sub-persoalan dan menggabung semua solusi setiap sub-persoalan. Algoritma *decrease and conquer* terdiri dari dua tahapan, yaitu 1) *decrease*, yaitu mereduksi persoalam menjadi beberapa sub-persoalan yang lebih kecil (biasanya 2 sub-persoalan) dan 2) *conquer*, yaitu memproses salah satu sub-persoalan.

Ada tiga varian dari *decrease and conquer* yaitu varian *decrease and conquer* yang mereduksi ukuran instans persoalan sebesar konstanta yang sama (biasanya 1) di setiap iterasi algoritma (*decrease by a constant*), varian *decrease and conquer* yang mereduksi ukuran instans persoalan sebesar faktor konstanta yang sama (biasanya 2) pada setiap iterasi algoritma (*decrease by constant factor*), dan varian *decrease and conquer* yang mereduksi ukuran instans persoalan secara bervariasi pada setiap iterasi algoritma (*decrease by a variable size*).

C. *Topological Sort* dengan Pendekatan Agoritma *Decrease and Conquer*

Algoritma *topological sort* dapat didasarkan dari implementasi algoritma *decrease and conquer*, tepatnya *decrease by a constant* yaitu dengan mencari simpul sumber di dalam graf yang tersisa yaitu simpul yang tidak memiliki busur yang mengarah padanya. Simpul-simpul sumber tersebut kemudian disalin sebagai jawaban, dihapus dari graf, dan busur-busur yang keluar dari simpul tersebut juga dihapus dari graf. Proses ini dilakukan secara berulang-ulang sampai graf kosong. Apabila graf

masih belum kosong sedangkan tidak ada lagi simpul sumber di graf yang tersesa maka artinya pada graf tersebut tidak dapat dilakukan *topological sort*.



Gambar 1 Ilustrasi *topological sort* dengan pendekatan *decrease and conquer*

Secara ringkas, *topological sorting* dengan menggunakan pendekatan algoritma *decrease and conquer* adalah sebagai berikut.

1. Dari graf (DAG) yang terbentuk, hitung semua derajat-masuk (in-degree) setiap simpul, yaitu banyaknya busur yang masuk pada simpul tersebut.
2. Pilih sembarang simpul yang memiliki derajat-masuk 0.
3. 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.
4. Ulangi langkah (2) dan (3) hingga semua simpul pada DAG terpilih.

BAB II

SOURCE CODE

Dalam menyelesaikan masalah penyusunan rencana kuliah dengan *topological sort* kami menggunakan bahasa pemrograman C. Struktur program yang digunakan yaitu sebagai berikut.

Adapun source program dalam bahasa C yang kami gunakan dalam menyelesaikan penyusunan rencana kuliah dengan *topological sort* adalah sebagai berikut.

A. 13519034-prog.c (berisi program utama)

```c
// RUHIYAH FARADISHI WIDIAPUTRI
// 13519034

#include <stdio.h>
#include "13519034-mesinkar.h"
#include "13519034-katamatkul.h"
#include "13519034-boolean.h"

int main(){
    //KAMUS
    char namafile[50];
    List data;
    hasil jawaban;

    // menyimpan dari txt dan menyimpannya
    printf("Masukkan directory file matkul (misalnya ../test/tes1.txt): \n");
    scanf("%s",namafile);
    START(namafile);
    CreateList(&data);
    bacaTeks(&data);
```

```
    // mencari susunan matkul yang harus diambil per semester
    int banymatkul = getBanyakMatkul(data);
    int sem = 0;
    while (banymatkul != 0 ){
        address P = data.First;
        int i = 0;
        // menyimpan matkul dengan 0 pre-requisite ke semester
        while (P != Nil){
            if (P->info.banyak == 0){
                salinkata(P->info.nama, &jawaban.diambil[sem].matakuliah[i]);
                i++;
                banymatkul--;
            }
            P = Next(P);
        }
        jawaban.diambil[sem].length = i;
        // menghapus node yang prereq-nya sudah dicatat
        for (int k=0; k<i; k++){
            P = Search(data, jawaban.diambil[sem].matakuliah[k]);
            hapusSemuaPrereq(&data,P->info.nama);
            Delete(&data, P);
        }
        sem++;
    }
    jawaban.banyaksem = sem;

    // mencetak hasil pencarian ke layar
    cetakRencanaMatkul(jawaban);

    return 0;
}
```

B. 13519034-katamatkul.h

```
#ifndef _KATAMATKUL_H_
#define _KATAMATKUL_H_

#define NMax 50
#define Nil NULL
#include "13519034-mesinkar.h"
#include <stdio.h>
#include <stdlib.h>

typedef struct{
    char TabKata[NMax];
    int Length;
}Kata;

typedef struct matkul *address;
typedef struct prereq *addrpreq;
typedef struct prereq{
    address pre;
    addrpreq next;
}prereq;
```

```c
typedef struct infotype{
    Kata nama;
    int banyak;
    addrpreq trail;
}infotype;
typedef struct matkul{
    infotype info;
    address next;
}matkul;

typedef struct {
    address First;
}List;

#define First(L) ((L).First);
#define Info(P) (P)->info
#define Next(P) (P)->next

typedef struct kuliah{
    Kata matakuliah[25];
    int length;
}kuliah;
```

```c
typedef struct hasil{
    int banyaksem;
    kuliah diambil[10];
}hasil;

//FUNGSI DAN PROSEDUR
boolean isKataSama(Kata k1, Kata k2);
void salinkata(Kata k1, Kata *k2);
void printKata(Kata kata);

void CreateList (List *L);
address Alokasi (infotype X);
void Dealokasi (address *P);
addrpreq Alokpreq(address r);
void Dealokpreq (addrpreq *P);

void bacaTeks(List *L);
void bacabaris(List *L);

address Search(List L, Kata k);
void InsLastList(List *L, Kata k, address *last);
void InsLastPrereq(address *Psekarang, address Pisi);
void hapusSemuaPrereq(List *L, Kata kata);
void hapusPrereq(List *L, address P, Kata kata);
void Delete(List *L, address P);

address alamatNol(List L);
int getBanyakMatkul(List L);

void salinGraf(List L);
void cetakRencanaMatkul (hasil jawab);

#endif
```

C. 13519034-katamatkul.c

```c
#include "13519034-katamatkul.h"
#include "13519034-mesinkar.h"

boolean isKataSama(Kata k1, Kata k2){
    if (k1.Length == k2.Length){
        int i = 0;
        boolean sama = true;
        while (sama && i < k1.Length){
            if (k1.TabKata[i] != k2.TabKata[i]){
                sama=false;
            }
            else{
                i++;
            }
        }
        return sama;
    }
    else{
        return false;
    }
}

void salinkata(Kata k1, Kata *k2){
    for (int i = 0; i < k1.Length; i++){
        k2->TabKata[i] = k1.TabKata[i];
    }
    k2->Length = k1.Length;
}

void printKata(Kata kata){
    for (int i = 0; i<kata.Length; i++){
        printf("%c",kata.TabKata[i]);
    }
}

void CreateList (List *L){
    (*L).First = Nil;
}
```

```c
address Alokasi (infotype X){
    address P;
    P = (address) malloc (1*sizeof(matkul));
    if (P != Nil){
        Info(P) = X;
        Next(P) = Nil;
        return P;
    }
    else{
        return Nil;
    }
}
```

```
void Dealokasi (address *P){
    free(*P);
}

addrpreq Alokpreq(address r){
    addrpreq P;
    P = (addrpreq) malloc (1*sizeof(prereq));
    if (P != Nil){
        P->pre = r;
        P->next = Nil;
        return P;
    }
    else{
        return Nil;
    }
}

void Dealokpreq (addrpreq *P){
    free (*P);
}
```

```
address Search(List L, Kata k){
    address P;
    boolean found = false;
    P = First(L);
    while (!found && P != Nil){
        if (isKataSama(P->info.nama, k)){
            found = true;
        }
        else{
            P = Next(P);
        }
    }
    return P;
}

address alamatNol(List L){
    address P;
    boolean found=false;
    P = First(L);
    while (!found && P != Nil){
        if (P->info.banyak == 0){
            found = true;
        }
        else{
            P = Next(P);
        }
    }
    return P;
}
```

```
void bacabaris(List *L){
    //baca matkulnya dulu
    Kata kata;
    address node; //untuk alamat matkul sekarang
    address P;
    int i = 0;
    while (CC != ',' && CC != '.'){
        kata.TabKata[i] = CC;
        i++;
        ADV();
    }
    kata.Length = i;

    node = Search(*L,kata);
    if (node == Nil){
        InsLastList(L,kata, &node);
    }

    if (CC != '.'){ // artinya punya prereq
        //baca pre-requisites nya
        ADV();
        while (CC != '\n' && CC !='\0' && CC !='.'){
            // baca kata
            Kata kata1;
            i = 0;
            while (CC != ',' && CC != '.'){
                kata1.TabKata[i] = CC;
                i++;
                ADV();
            }
            kata1.Length=i;
            // cari udah didaftarkan belum terus isi prereq
            P = Search(*L,kata1);
            if (P != Nil){
                InsLastPrereq(&node, P);
            }
            else{
                address dummy;
                InsLastList(L,kata1,&dummy);
                InsLastPrereq(&node, dummy);
            }
            // setelan akhir
            node->info.banyak++;
            if (CC == '.'){
                ADV();
            }
            ADV();
        }
    }
    else{
        ADV();
        ADV();
    }
}
```

```
void bacaTeks(List *L){
    while (CC != '\0'){
        bacabaris(L);
        ADV();
    }
}
```

```
void InsLastList(List *L, Kata k, address *P){
    address last;
    infotype in;

    in.banyak = 0;
    salinkata(k, &in.nama);
    in.trail=Nil;
    *P = Alokasi(in);
    if (*P != Nil){
        if (L->First==Nil){
            Next(*P) = First(*L);
            L->First = *P;
        }
        else{
            last = First(*L);
            while (Next(last) != Nil){
                last = Next(last);
            }
            Next(last) = *P;
        }
    }
}

void InsLastPrereq(address *Psekarang, address Pisi){
    addrpreq alm, pred;
    addrpreq P = Alokpreq(Pisi);
    if (P != Nil){
        alm = (*Psekarang)->info.trail;
        pred = Nil;
        while (alm != Nil){
            pred = alm;
            alm = alm->next;
        }
        if (pred != Nil){
            alm = P;
            Next(pred) = alm;
        }
        else{
            (*Psekarang)->info.trail = P;
        }
    }
}
```

```
int getBanyakMatkul(List L){
    address P;
    int i = 0;
    P = First(L);
    while (P != Nil){
        P = Next(P);
        i++;
    }
    return i;
}

void hapusSemuaPrereq(List *L, Kata kata){
    address P = First(*L);
    while (P != Nil){
        hapusPrereq(L,P,kata);
        P = Next(P);
    }
}
```

```
void hapusPrereq(List *L, address P, Kata kata){
    addrpreq alm, pred, hapus;
    int dihapus = 0;
    pred = Nil;
    alm = P->info.trail;
    while (alm != Nil){
        if (isKataSama(alm->pre->info.nama,kata)){
            if (pred == Nil){
                P->info.trail = Next(alm);
                hapus = alm;
                alm = alm->next;
                Dealokpreq(&hapus);
            }
            else{
                Next(pred) = Next(alm);
                hapus = alm;
                alm= alm->next;
                Dealokpreq(&hapus);
            }
            dihapus++;
        }
        else{
            pred = alm;
            alm = Next(alm);
        }
    }
    P->info.banyak = P->info.banyak - dihapus;
}
```

```
void Delete(List *L, address P){
    address prec, now;
    boolean found = false;
    prec = Nil;
    now = First(*L);
    while (now != Nil && now != P){
        prec = now;
        now = Next(now);
    }
    if (prec == Nil){
        (*L).First = Next(P);
        Dealokasi(&P);
    }
    else{
        Next(prec) = Next(P);
        Dealokasi(&P);
    }
}

void salinGraf (List L){
    address P = L.First;
    while (P != Nil){
        printf("Induknya: \n");
        printKata(P->info.nama);
        printf("\n");
        printf("banyak anak = %d\n",P->info.banyak);
        if (P->info.banyak > 0){
            addrpreq alm;
            alm = P->info.trail;
            printf("anaknya: ");
            while (alm != Nil){
                printKata(alm->pre->info.nama);
                printf(",");
                alm = Next(alm);
            }
        }
        P = Next(P);
        printf("\n\n");
    }
}
```

```
void cetakRencanaMatkul (hasil jawab){
    for (int i = 0; i < jawab.banyaksem; i++){
        printf("Semester %d : ",i+1);
        for (int j = 0; j < jawab.diambil[i].length; j++){
            printKata(jawab.diambil[i].matakuliah[j]);
            if (j != jawab.diambil[i].length - 1){
                printf(", ");
            }
        }
        printf("\n");
    }
}
```

D. 13519034-mesinkar.h

```c
/* File: 13519034-mesinkar.h */
/* Definisi Mesin Karakter */

#ifndef __MESIN_KAR_H_
#define __MESIN_KAR_H_

#include "13519034-boolean.h"

#define MARK NULL
/* State Mesin */
extern char CC;
extern boolean EOP;

void START(char*);
/* Mesin siap dioperasikan. Pita disiapkan untuk dibaca.
   Karakter pertama yang ada pada pita posisinya adalah pada jendela.
   I.S. : sembarang
   F.S. : CC adalah karakter pertama pada pita
          Jika CC != MARK maka EOP akan padam (false)
          Jika CC = MARK maka EOP akan menyala (true) */

void ADV();
/* Pita dimajukan satu karakter.
   I.S. : Karakter pada jendela = CC, CC != MARK
   F.S. : CC adalah karakter berikutnya dari CC yang lama,
          CC mungkin = MARK
          Jika  CC = MARK maka EOP akan menyala (true) */

#endif
```

E. 13519034-mesinkar.c

```c
/* File: 13519034-mesinkar.c */
/* Implementasi Mesin Karakter */

#include "13519034-mesinkar.h"
#include <stdio.h>

char CC;
boolean EOP;

static FILE * teks;
static int retval;

void START(char* namafile) {
/* Mesin siap dioperasikan. Pita disiapkan untuk dibaca.
   Karakter pertama yang ada pada pita posisinya adalah pada jendela.
   I.S. : sembarang
   F.S. : CC adalah karakter pertama pada pita. Jika CC != MARK maka EOP akan padam
          Jika CC = MARK maka EOP akan menyala (true) */

    /* Algoritma */
    teks = fopen(namafile,"r");
    EOP = false;
    ADV();
}
```

```
void ADV() {
/* Pita dimajukan satu karakter.
   I.S. : Karakter pada jendela =
          CC, CC != MARK
   F.S. : CC adalah karakter berikutnya dari CC yang lama,
          CC mungkin = MARK.
               Jika  CC = MARK maka EOP akan menyala (true) */

      /* Algoritma */
      retval = fscanf(teks,"%c",&CC);
      if (retval == EOF) {
            fclose(teks);
            EOP = true;
      }
}
```

F. 13519034-boolean.h

```
/* Definisi type boolean */

#ifndef _BOOLEAN_h
#define _BOOLEAN_h

#define boolean unsigned char
#define true 1
#define false 0

#endif
```

BAB III

HASIL PERCOBAAN

A. Percobaan ke-1

a. Input (tes1.txt)

```
C1,C2,C3.
C2,C3.
C3.
```

b. Output

```
ruhiyahf@LAPTOP-0LUHMJ0M:/mnt/c/users/farad/onedrive/documents/smt4/tucil/tucil2final/stima_tucil2/bin$ ./prog
Masukkan directory file matkul (misalnya ../test/tes1.txt):
../test/tes1.txt
Semester 1 : C3
Semester 2 : C2
Semester 3 : C1
```

B. Percobaan ke-2

a. Input (tes2.txt)

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

b. Output

```
ruhiyahf@LAPTOP-0LUHMJ0M:/mnt/c/users/farad/onedrive/documents/smt4/tucil/tucil2final/stima_tucil2/bin$ ./prog
Masukkan directory file matkul (misalnya ../test/tes1.txt):
../test/tes2.txt
Semester 1 : C3
Semester 2 : C1
Semester 3 : C4
Semester 4 : C2
Semester 5 : C5
```

C. Percobaan ke-3

a. Input (tes3.txt)

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

b. Output

```
ruhiyahf@LAPTOP-0LUHMJ0M:/mnt/c/users/farad/onedrive/documents/smt4/tucil/tucil2final/stima_tucil2/bin$ ./prog
Masukkan directory file matkul (misalnya ../test/tes1.txt):
../test/tes3.txt
Semester 1 : C1, C2
Semester 2 : C3
Semester 3 : C4
Semester 4 : C5
```

D. Percobaan ke-4

a. Input (tes4.txt)

```
C1,C4.
C2,C1,C4.
C3,C1,C4.
C4.
C5,C2,C7.
C6,C3,C4,C7.
C7,C2,C4.
```

b. Output

```
ruhiyahf@LAPTOP-0LUHMJ0M:/mnt/c/users/farad/onedrive/documents/smt4/tucil/tucil2final/stima_tucil2/bin$ ./prog
Masukkan directory file matkul (misalnya ../test/tes1.txt):
../test/tes4.txt
Semester 1 : C4
Semester 2 : C1
Semester 3 : C2, C3
Semester 4 : C7
Semester 5 : C5, C6
```

E. Percobaan ke-5

a. Input (tes5.txt)

```
C1,C3,C5.
C2,C3.
C3.
C4,C1,C2,C5.
C5.
C6,C5.
C7,C4,C8,C4.
C9,C7,C8.
C10,C6,C8,C11.
C11,C6.
C12,C9,C10.
C13,C10.
```

b. Output

```
ruhiyahf@LAPTOP-0LUHMJ0M:/mnt/c/users/farad/onedrive/documents/smt4/tucil/tucil2final/stima_tucil2/bin$ ./prog
Masukkan directory file matkul (misalnya ../test/tes1.txt):
../test/tes5.txt
Semester 1 : C3, C5, C8
Semester 2 : C1, C2, C6
Semester 3 : C4, C11
Semester 4 : C7, C10
Semester 5 : C9, C13
Semester 6 : C12
```

F. Percobaan ke-6

a. Input (tes6.txt)

```
C1,C7.
C2,C1,C7.
C3,C4,C7,C9.
C4.
C5,C4,C10.
C6,C1.
C7.
C8,C2,C4.
C9,C4,C5.
C10.
```

b. Output

```
ruhiyahf@LAPTOP-0LUHMJ0M:/mnt/c/users/farad/onedrive/documents/smt4/tucil/tucil2final/stima_tucil2/bin$ ./prog
Masukkan directory file matkul (misalnya ../test/tes1.txt):
../test/tes6.txt
Semester 1 : C7, C4, C10
Semester 2 : C1, C5
Semester 3 : C2, C9, C6
Semester 4 : C3, C8
```

G. Percobaan ke-7

a. Input (tes7.txt)

```
MATH101.
ICS102.
COE202.
ICS253,ICS102.
ICS201,ICS102.
ICS202,ICS201.
ICS343,ICS201.
ICS233,ICS201,COE202.
ICS353,ICS253,ICS202.
ICS444,ICS343,ICS431.
ICD431,ICS233.
```

b. Output

```
ruhiyahf@LAPTOP-0LUHMJ0M:/mnt/c/users/farad/onedrive/documents/smt4/tucil/tucil2final/stima_tucil2/bin$ ./prog
Masukkan directory file matkul (misalnya ../test/tes1.txt):
../test/tes7.txt
Semester 1 : MATH101, ICS102, COE202, ICS431
Semester 2 : ICS253, ICS201
Semester 3 : ICS202, ICS343, ICS233
Semester 4 : ICS353, ICS444, ICD431
```

H. Percobaan ke-8

c. Input (tes8.txt)

Catatan: input ini bukan dari keadaan sebenarnya.

```
MA1101.
KU1102.
IF4092,IF4091.
IF4090,IF3280.
IF3280.
IF3260,IF2130,IF2110,IF2123.
IF3250,IF3150,IF2250.
IF3230,IF3130.
IF3270,IF3170,IF2110.
IF3210,IF2130,IF2110.
IF3151,IF2250.
IF2210,IF2110.
IF2130.
IF2123,MA1101.
IF2124.
IF2120.
IF2110.
IF2121.
IF3140,IF2240.
IF3150,IF2250.
IF3141,IF2240,IF2250.
IF3130,IF2230.
IF3110,IF2210,IF2110.
IF3170,IF2121,IF2124,IF2220,IF2211.
IF2250.
IF2240,IF2110,IF2130,IF2121.
IF2230,IF2130.
IF2220,MA1101,MA1201,IF2120.
IF2211.
IF1210.
```

d. Output

```
ruhiyahf@LAPTOP-0LUHMJ0M:/mnt/c/users/farad/onedrive/documents/smt4/tucil/tucil2final/stima_tucil2/bin$ ./prog
Masukkan directory file matkul (misalnya ../test/tes1.txt):
../test/tes8.txt
Semester 1 : MA1101, KU1102, IF4091, IF3280, IF2130, IF2110, IF2250, IF2124, IF2120, IF2121, IF2211, MA1201, IF1210
Semester 2 : IF4092, IF4090, IF2123, IF3150, IF3210, IF3151, IF2210, IF2240, IF2230, IF2220
Semester 3 : IF3260, IF3250, IF3130, IF3170, IF3140, IF3141, IF3110
Semester 4 : IF3230, IF3270
```

BAB IV

EVALUASI DAN LINK KODE PROGRAM

A. Evaluasi

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	✓	

B. Link Source Code

Github: https://github.com/ruhiyahfw/stima_tucil2.git
Drive: https://drive.google.com/drive/folders/1yiOiDzAToFNS_1o5rtcw-TY4fcB9xRtk?usp=sharing