Tugas Kecil IF2121 Strategi Algoritma

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

Nama: Muhammad Zubair
NIM: 13519172

Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung
2021

Algoritma topological sort:
Melakukan iterasi sampai graf kosong dengan mencari mata kuliah yang bisa diambil pada semester tersebut lalu menghilangkannya dari graf sehingga persoalan direduksi sebesar banyaknya mata kuliah yang tidak ada prequisitenya per iterasi sehingga variannya pada decrease and conquer adalah decrease by a variable size. Kemudian dilakukan terus sampai graf kosong.

Source Code:

```
#include <iostream>
#include <fstream>
#include <bits/stdc++.h>

class matkul;
class prerequisite;

// struktur data linked list bernama prequisite
class prerequisite {
private:
    matkul *g;
    prerequisite *next;

public:
    // class matkul dapat mengakses private variable dari class prequisite
    friend class matkul;
    prerequisite(matkul *_g): g(_g), next(nullptr) {}
    prerequisite(matkul *_g, prerequisite *_next): g(_g), next(_next) {}
    ~prerequisite() {}
};

// struktur data linked list bernama matkul
// yang tersambung ke prequisite dari matkul tersebut dan matkul lain.
class matkul {
private:
    std::string name;   // nama matkul
    int count_prereq;   // banyaknya prequisite
    prerequisite *prereqs;  // menyambungkan ke prequsite dari matkul
    matkul *next;   // menyambungkan ke matkul lain

public:
    // class graph dapat mengakses private variable dari class matkul
    friend class graph;
    matkul(std::string _name) {
        name = _name;
        count_prereq = 0;
        prereqs = nullptr;
```

```
    next = nullptr;
}

// jika matkul didelete maka prequisitenya juga didelete
~matkul() {
    prerequisite *node = prereqs;
    while (prereqs) {
        node = prereqs;
        prereqs = node->next;
        delete node;
    }
}

// menambahkan prequisite ke dalam matkul
void add_prereq(matkul* g) {
    if (!prereqs) {
        prerequisite *node = new prerequisite(g);
        prereqs = node;
    } else {
        prerequisite *new_neighbor = new prerequisite(g, prereqs);
        prereqs = new_neighbor;
    }
    count_prereq++;
}

// menghapus prequisite dari matkul jika ada
// jika tidak ada prequisitenya maka tidak melakukan apa-apa
bool del_prereq(matkul* g) {
    if (count_prereq > 0) {
        // jika ada prequisitenya
        if (prereqs->g == g) {  // elmt >= 1, delete first
            prerequisite *node = prereqs;
            prereqs = prereqs->next;
            delete node;
            count_prereq--;
            return true;
        } else {
            prerequisite *node = prereqs, *prev = nullptr;
            while (node && node->g != g) {
                prev = node;
                node = node->next;
            }
            if (node) {
                prev->next = node->next;
                delete node;
```

```
                    count_prereq--;
                    return true;
                }
            }
        }
        return false;
    }

    // mengakses matkul selanjutnya yang tersambung
    matkul *Next() { return next; }

    std::string get_name() { return name; }
    int get_num_prereq() { return count_prereq; }

};

class graph {
private:
    std::string name;   // nama mahasiswa
    matkul *first;
public:
    graph(std::string _name): name(_name), first(nullptr) {}

    ~graph() {
        matkul *node = nullptr;
        while (first) {
            node = first;
            delete node;
            first = first->next;
        }
    }

    // jika matkul yang belum diambil sudah habis atau belum
    bool empty() { return (first) ? false : true; }

    // menambahkan matkul ke dalam graph jika belum ditambahkan
    // jika sudah ditambahkan maka tidak melakukan apa-apa
    // dan mengembalikan address dari matkul yang dicari atau ditambahkan tersebut
    matkul *add_matkul(std::string name_matkul) {
        // cek apakah sudah ditambahkan
        matkul *g = check_matkul(name_matkul);
        if (g == nullptr) {
            // jika belum ditambahkan
            g = new matkul(name_matkul);
            if (!first) {
```

```
            first = g;
        } else {
            g->next = first;
            first = g;
        }
    }
    return g;
}

// menghapus matkul beserta prequisitenya
matkul *del_matkul(matkul *matkul_a) {

    if (first == matkul_a) {
        first = matkul_a->next;
        matkul *node = first;
        while (node) {
            // menghapus prequisitenya
            node->del_prereq(matkul_a);
            node = node->next;
        }
        delete matkul_a;
        return first;

    } else {
        matkul *node = first, *prev = nullptr;
        while (node && node != matkul_a) {
            prev = node;
            node = node->next;
        }
        if (node) {
            prev->next = node->next;
            matkul *next_after_del_node = prev->next;
            matkul *del_preq = first;
            while (del_preq) {
                // menghapus prequisitenya
                del_preq->del_prereq(node);
                del_preq = del_preq->next;
            }
            delete node;
            return prev->next;
        }
        return nullptr;
    }
}
```

```
    // mengecek apakah matkul sudah ditambahkan ke dalam graph
    matkul *check_matkul(std::string name_matkul) {
        matkul *node = first;
        while (node && node->name != name_matkul) {
            node = node->next;
        }
        return node;
    }

    // mendapatkan address pertama matkul dalam graph
    matkul *get_first() { return first; }

    // mendapatkan nama graph
    std::string get_name() { return name; }

};

int main() {
    std::string file;
    std::cin >> file;
    graph G_Matkul("Rencana Kuliah Mahasiswa A");
    {   // baca file dan masukkan ke dalam graph
        std::string r_line;
        file = "../test/" + file;
        std::ifstream r_file(file);
        std::string name_matkul;
        // baca file per baris dan masukkan ke variabel r_line
        while (getline(r_file, r_line)) {
            int idx = 0;
            // iterasi dan cek per karakter pada r_line
            // jika ada ',' atau '.' maka masukkan ke dalam graf nama matkul
            while (r_line[idx] != ',' && r_line[idx] != '.') {
                char& c = r_line[idx++];
                // mengumpulkan nama matkul
                if (c != ' ')
                    name_matkul.push_back(c);
            }
            // masukkan nama matkul ke dalam graf jika belum ada
            // lalu simpan addressnya pada variabel g
            matkul *g = G_Matkul.add_matkul(name_matkul);
            name_matkul.clear();
            // jika sudah tanda '.', baca baris selanjutnya
            if (r_line[idx] == '.') continue;

            // karena index pada r_line berada pada tanda ',' maka index tambah 1
```

```
            idx++;

            // iterasi kembali per karakter pada r_line
            while (idx < r_line.length()) {
                char &c = r_line[idx++];
                // jika ada spasi, dilewati
                if (c == ' ') continue;
                // jika ada ',' atau '.' maka nama matkul dimasukkan ke dalam graph jika belum ada
                // dan masukkan kedalam prequisite matkul g.
                if (c == ',' || c == '.') {
                    // tambahkan ke dalam graf jika belum ada pada graph
                    // lalu simpan prequisitenya kedalam graf
                    matkul *prereq = G_Matkul.add_matkul(name_matkul);
                    // masukkan prequisitenya ke dalam matkul g
                    g->add_prereq(prereq);
                    name_matkul.clear();
                } else {
                    name_matkul.push_back(c);
                }
            }
        }
    }

    std::vector <std::vector <std::string>> semester;   // menyimpan kumpulan matkul per semester
    // iterasi sampai tidak ada matkul yang bisa diambil
    while (!G_Matkul.empty()) {
        // iterasi mulai dari matkul awal
        // untuk cek apakah ada matkul yg bisa diambil pada semester tsb
        matkul *node = G_Matkul.get_first();
        std::vector <std::string> semester_a;   // menyimpan nama matkul
        std::vector <matkul*> addr_semester_a;  // menyimpan address matkul
        // iterasi
        while (node) {
            // jika ada matkul yg bisa diambil
            if (node->get_num_prereq() == 0) {
                // dimasukkan ke dalam semester_a dan addr_semester_a
                semester_a.push_back(node->get_name());
                addr_semester_a.push_back(node);
            }
            node = node->Next();
        }
        // delete matkul yang sudah diambil di dalam graph
        for (auto matkul_a = addr_semester_a.begin(); matkul_a != addr_semester_a.end(); matkul_a++) {
            G_Matkul.del_matkul(*matkul_a);
        }
```

```
        // menyimpan kumpulan matkul yang bisa diambil pada semester tersebut
        semester.push_back(semester_a);
    }

    // Mencetak matkul-matkul yang bisa diambil per semester
    std::cout << '\n';
    std::cout << G_Matkul.get_name() << ": \n";
    for (int i = 0; i < semester.size(); i++) {
        std::cout << "Semester " << i+1 << ": ";
        int j = 0;
        while (j < semester[i].size()-1)
            std:: cout << semester[i][j++] << ", ";
        std::cout << semester[i][j] << '\n';
    }
    std::cout << '\n';

    return 0;
}
```

Screenshoot Input dan Output

File test.txt

StrategiAlgoritma > Tucil > tucil2 > test > test.txt

```
8   C1, C3.
7   C2, C1, C4.
6   C3.
5   C4, C1, C3.
4   C5, C2, C4.
3   C6, C2.
2   C7.
1   C8, C9.
9   C9.
```

Hasil:

```
PS C:\Users\zubai\OneDrive - Institut Teknologi Bandung\Programs\ProgrammingLanguage\C++\StrategiAlgoritma\Tucil\tucil2\bin> ./a
test.txt

Rencana Kuliah Mahasiswa A:
Semester 1: C9, C7, C3
Semester 2: C8, C1
Semester 3: C4
Semester 4: C2
Semester 5: C6, C5
```

File test2.txt

StrategiAlgoritma > Tucil > tucil2 > test > test2.txt

```
6   C1.
5   C2, C3, C4.
4   C3, C1, C4.
3   C4.
2   C5, C1, C2.
1   C6.
7
```

Hasil:

```
PS C:\Users\zubai\OneDrive - Institut Teknologi Bandung\Programs\ProgrammingLanguage\C++\StrategiAlgoritma\Tucil\tucil2\bin> ./a
test2.txt

Rencana Kuliah Mahasiswa A:
Semester 1: C6, C4, C1
Semester 2: C3
Semester 3: C2
Semester 4: C5
```

File test3.txt

```
StrategiAlgoritma > Tucil > tucil2 > test > test3.txt
  4    A, C.
  3    C, D.
  2    D.
  1    F, A, D.
5      R, D, F.
```

Hasil:

```
PS C:\Users\zubai\OneDrive - Institut Teknologi Bandung\Programs\ProgrammingLanguage\C++\StrategiAlgoritma\Tucil\tucil2\bin> ./a
test3.txt

Rencana Kuliah Mahasiswa A:
Semester 1: D
Semester 2: C
Semester 3: A
Semester 4: F
Semester 5: R
```

File test4.txt

```
StrategiAlgoritma > Tucil > tucil2 > test > test4.txt
  5    A, B.
  4    B, C, D.
  3    C, E.
  2    D, E.
  1    E.
6      F.
```

Hasil:

```
PS C:\Users\zubai\OneDrive - Institut Teknologi Bandung\Programs\ProgrammingLanguage\C++\StrategiAlgoritma\Tucil\tucil2\bin> ./a
test4.txt

Rencana Kuliah Mahasiswa A:
Semester 1: F, E
Semester 2: D, C
Semester 3: B
Semester 4: A
```

File test5.txt

```
StrategiAlgoritma > Tucil > tucil2 > test > test5.txt
  4    A, S, E.
  3    C.
  2    D, A, S.
  1    E, C.
5      S, E.
```

Hasil:

```
PS C:\Users\zubai\OneDrive - Institut Teknologi Bandung\Programs\ProgrammingLanguage\C++\StrategiAlgoritma\Tucil\tucil2\bin> ./a
test5.txt

Rencana Kuliah Mahasiswa A:
Semester 1: C
Semester 2: E
Semester 3: S
Semester 4: A
Semester 5: D
```

File test6.txt

```
StrategiAlgoritma > Tucil > tucil2 > test > test6.txt
  6    A, R, T.
  5    B, C, A.
  4    C, A.
  3    F, B, I.
  2    I.
  1    R, I.
7      T, I.
```

Hasil:

```
PS C:\Users\zubai\OneDrive - Institut Teknologi Bandung\Programs\ProgrammingLanguage\C++\StrategiAlgoritma\Tucil\tucil2\bin> ./a
test6.txt

Rencana Kuliah Mahasiswa A:
Semester 1: I
Semester 2: T, R
Semester 3: A
Semester 4: C
Semester 5: B
Semester 6: F
```

File test7.txt

```
StrategiAlgoritma > Tucil > tucil2 > test > test7.txt
  5    A, E, R.
  4    E.
  3    F, R, K, J.
  2    J, K, A.
  1    K, E.
6      R, K.
```

Hasil:

```
PS C:\Users\zubai\OneDrive - Institut Teknologi Bandung\Programs\ProgrammingLanguage\C++\StrategiAlgoritma\Tucil\tucil2\bin> ./a
test7.txt

Rencana Kuliah Mahasiswa A:
Semester 1: E
Semester 2: K
Semester 3: R
Semester 4: A
Semester 5: J
Semester 6: F
```

File test8.txt

```
StrategiAlgoritma > Tucil > tucil2 > test > test8.txt
  5    A, E, W.
  4    D, E.
  3    E.
  2    F, E.
  1    R, F, D.
6      W, E, R.
```

Hasil:

```
PS C:\Users\zubai\OneDrive - Institut Teknologi Bandung\Programs\ProgrammingLanguage\C++\StrategiAlgoritma\Tucil\tucil2\bin> ./a
test8.txt

Rencana Kuliah Mahasiswa A:
Semester 1: E
Semester 2: F, D
Semester 3: R
Semester 4: W
Semester 5: A
```

Alamat tempat kode sumber program:
https://drive.google.com/drive/folders/1lBRpWLs2LzyMCCnikD3W1zpFIM_Hiv_K?usp=sharing

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.	√	