**LAPORAN TUGAS KECIL 2**

**IF2211 Strategi Algoritma**

**Penyusunan Rencana Kuliah dengan *Topological Sort***

**(Penerapan *Decrease and Conquer*)**



**Daffa Ananda Pratama Resyaly**

**13519107**

**Kelas 02**

1. **Algoritma Topological Sorting dan Kaitannya dengan Pendekatan “Decrease and Conquer”**

*Topological Sorting* adalah suatu metode pengurutan simpul pada graf berarah dimana untuk setiap sisi berarah dari simpul A ke simpul B, simpul A pasti muncul sebelum simpul B muncul di dalam pengurutan. Hasil dari *Topological Sorting* tidaklah unik, yang berarti kemungkinan terdapat lebih dari satu hasil *Topological Sorting* untuk setiap graf berarah. Selain itu, algoritma ini juga tidak dapat diaplikasikan pada segala macam graf, melainkan terbatas hanya pada graf yang tidak memiliki *cycle*. Tipe graf yang valid diimplementasikan untuk *Topological Sorting* hanyalah *Directed Acylic Graph* (DAG) yang merupakan sejenis graf berarah yang tidak memiliki *cycle* di dalamnya. Segala macam *tree* juga cocok untuk diimplementasikan untuk *Topological Sorting* karena *tree* tidak mungkin mempunyai *cycle* di dalamnya.

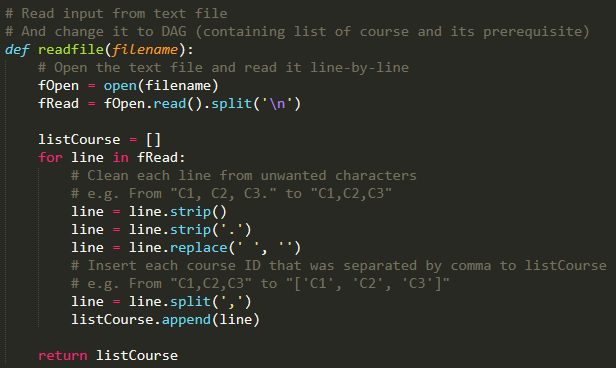
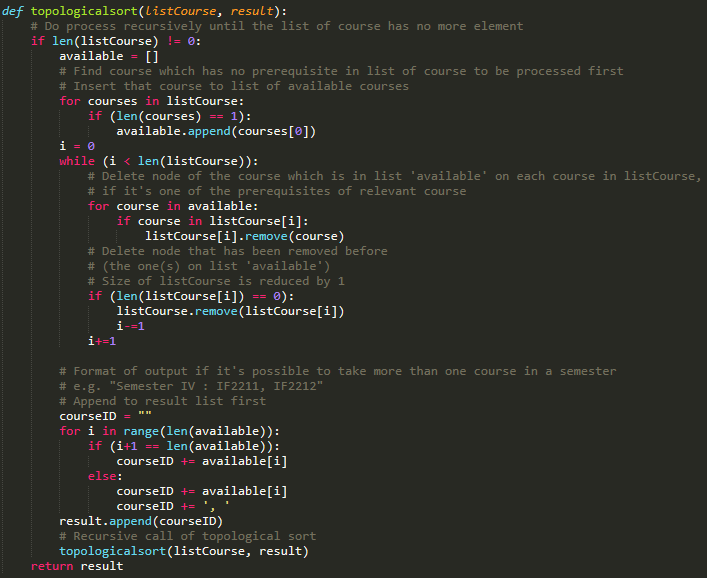
Algoritma *Topological Sorting* memiliki banyak aplikasi dalam dunia nyata karena banyak hal dalam dunia nyata dapat direpresentasikan dalam bentuk graf berarah dimana suatu kejadian harus terjadi sebelum kejadian berikutnya terjadi. Contoh dari pengaplikasian *Topological Sorting* pada dunia nyata adalah penyusunan rencana kuliah, *program dependencies*, penjadwalan suatu acara, dan instruksi dalam bahasa assembly. Algoritma pengurutan menggunakan *Topological Sorting* memiliki kompleksitas waktu O(V+E), dimana V adalah jumlah simpul dan E adalah jumlah sisi pada suatu graf berarah.

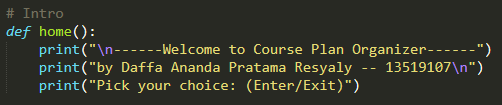
*Decrease and Conquer* merupakan suatu strategi metode perancangan algoritma dengan mereduksi persoalan menjadi dua upa persoalan yang lebih kecil, kemudian hanya satu upa persoalan dari kedua upa persoalan tersebut yang diproses atau diselesaikan. Di dalam algoritma *Decrease and Conquer*, tidak ada tahap penggabungan upa persoalan (tahap *combine*) karena yang diproses hanyalah satu upa persoalan saja. Algoritma *Decrease and Conquer* memiliki tiga variasi, yaitu *decrease by a constant* (ukuran instans persoalan direduksi sebesar konstanta yang sama, biasanya 1, pada setiap iterasi algoritma), *decrease by a constant factor* (ukuran instans direduksi sebesar faktor konstanta yang sama, biasanya 2, pada setiap iterasi algoritma), dan *decrease by a variable size* (ukuran instans persoalan direduksi bervariasi pada setiap iterasi algoritma).

Kaitan algoritma *Topological Sorting* dengan pendekatan *Decrease and Conquer* adalah algoritma *Topological Sorting* mereduksi suatu persoalan bervariasi tergantung dengan ukuran ruang lingkup persoalannya. Secara umum, algoritma *Topological Sorting* tersebut adalah sebagai berikut. Pertama, akan dipilih simpul secara acak dari graf yang belum pernah dikunjungi. Lalu, dimulai dari simpul yang terpilih tersebut, lakukan *Depth First Search* (DFS) yang mengunjungi setiap simpul tetangga yang belum pernah dikunjungi sebelumnya. Kemudian, DFS akan dipanggil kembali secara rekursif. Dalam pemanggilan rekursif tersebut, simpul tersebut ditambahkan ke list *ordering* pada posisi terakhir.

1. **Source Code (Python)**

Pada *source code* dari program yang telah dibuat, terdapat sebuah file python yang berisi tiga fungsi utama untuk menerapkan algoritma *Topological Sorting* untuk menyusun daftar mata kuliah dari file yang ada dan satu fungsi tambahan untuk memperindah penjalanan program pada CLI.

* 1. **Fungsi Utama**
     1. 
     2. 
     3. **E**
  2. **Fungsi Tambahan**



* 1. **Main Program**

1. **Screenshot dari Input dan Output**

**3.1.**

1. **Alamat tempat kode sumber program**

Repositori Github:

<https://github.com/slarkdarr/Tucil2_13519107.git>

1. **Checklist**

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