TUGAS KECIL II

ALGORITMA DECREASE AND CONQUER

IF2211 – Strategi Algoritma

Oleh

Ariya Adinatha (13519048) K01

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

2020

Algoritma Decrease and Conquer

Decrease and Conquer merupakan salah satu algoritma yang mirip dengan *divide and conquer*, dimana algoritma *decrease and conquer* ini bekerja dengan memecahkan permasalahan dalam bentuk n-1 atau secara konstan. Lalu setelah permasalahan dicari solusi terbaiknya. Berbeda dengan divide and conquer yang menyelesaikan setiap permasalahan dalam sub permasalahannya lalu menggabungkan seluruh solusi di akhir, *decrease and conquer* menyelesaikan satu sub permasalahan saja. Penyelesaian *decrease and conquer* dapat dilakukan dengan metode *top down* maupun metode *bottom up*. Terdapat 3 jenis reduksi yang umum dalam *decrease and conquer* yaitu *decrease by constant* dimana biasanya dikurangi sebanyak 1, *decrease by constant factor* yang biasanya mengurangi sebanyak faktor setengah, *variable size decrease*. Beberapa permasalahan yang dapat diselesaikan dengan menggunakan algoritma *decrease and conquer* ini adalah *insertion sort*, *selection sort,* dan juga *topological sort*.

Topological Sort merupakan algoritma pengurutan yang dapat mengurutkan *directed acyclic graph* sehingga keterhubungan antar node pada graph tersebut dapat diurutkan berdasarkan kebutuhannya. *Topological Sort* ini sering digunakan dalam penyelesaian masalah di dunia nyata seperti pengambilan mata kuliah yang memiliki syarat, instalasi program yang membutuhkan *dependencies* terhadap program lain, dan lain lain.

Pada tugas kecil kali ini, dibuat sebuah program sederhana berdasarkan algoritma *decrease and conquer* dan juga *topological sort* yang dapat membantu pengguna dalam proses pengambilan mata kuliah. Di dalam program yang telah dibuat pengguna dapat memasukkan file yang berisi mata kuliah yang tersedia dan juga prasyarat yang harus diambil sebelum mengikuti kuliah tersebut. Pertama program akan membaca isi dari file dan membuat sebuah array yang berisi *vertices* dan *edges*. Dimana *vertices* digunakan sebagai mata kuliah yang dipilih dan *edges* sebagai prasyarat dari mata kuliah tersebut. Dalam hal ini algoritma *decrease and conquer* dapat melakukan *topological sort*. Yang mana proses ini akan mengunjungi sebuah node dan menelusuri setiap *edges* yang dimiliki olehnya. Setelah tidak ada yang bisa dikunjungi, akan dipilih node yang lain untuk ditelusuri *edges*-nya. Begitulah garis besar cara kerja *topological sort*. Dalam hal ini node dibagi dan menyelesaikan masalah terhadap node tersebut sebelum berpindah ke persoalan berikutnya, sama seperti cara kerja algoritma *decrease and conquer*

SOURCE CODE

src > 13519048.py > ...

```python
import os

path_parent = os.path.dirname(os.getcwd())
os.chdir(path_parent)


def getRow(namaFile):
    filepath = str(os.getcwd())+"/test/"+str(namaFile)
    row = 0
    with open(filepath) as fp:
        line = fp.readline()
        while line:
            row += 1
            line = fp.readline()
    return row


def fileTodaGraph(namaFile):
    filepath = str(os.getcwd())+"/test/"+str(namaFile)
    row = getRow(namaFile)
    arraydaGra = [[] for i in range(row)]
    with open(filepath) as fp:
        line = fp.readline()
        countIndex = 0
        while line:
            striped = (line.rstrip(".\n").split(", "))
            arrayEdges = []
            index = 0
            for content in striped:
                if (index == 0):
                    vertices = content
                    index += 1
                else:
                    arrayEdges.append(content)
            arraydaGra[countIndex] += [[vertices], arrayEdges]
            countIndex += 1
            line = fp.readline()
    return (arraydaGra)


def head(array):
    return array[0]


def tail(array):
    return array[1:]
```

```
48
49  def isOneElmt(array):
50      return len(array) == 1
51
52
53  def isEmpty(array):
54      return len(array) == 0
55
56
57  # mencari nilai minimum dari edges
58  def minArray(array, minimum, index):
59      if (minimum > len(head(array)[1])):
60          minimum = len(head(array)[1])
61          index = 0
62      if (isOneElmt(array)):
63          if (minimum > len(array[0][1])):
64              minimum = len(array[0][1])
65          return minimum
66      minArray(tail(array), minimum, index)
67      return minArray(tail(array), minimum, index)
68
69
70  # Melakukan sorting pada array
71  def selectionSort(array):
72      sortedArray = [[] for i in range(row)]
73      indexSort = 0
74      while (len(array) != 0):
75          minimum = 99
76          for i in range(len(array)):
77              if minimum > len(array[i][1]):
78                  minimum = len(array[i][1])
79                  index = i
80          sortedArray[indexSort] = array[index]
81          indexSort += 1
82          array.remove(array[index])
83      return sortedArray
84
85
86  def isTaken(taking, taken):
87      for i in taken:
88          if (taking == [i]):
89              return True
90      return False
91
```

```python
def takeCourse(array):
    course = [[] for i in range(7)]

    # Mengambil course tanpa syarat
    for i in range(len(array)):
        if len(array[i][1]) == 0:
            course[0] += array[i][0]

    # list course yang sudah diambil
    taken = [elm for index in course for elm in index]
    semester = 1
    totalCourse = len(array)
    while (len(taken) != totalCourse):
        goingToTake = []
        for i in range(len(array))[1:]:
            preq = []
            preq += (array[i][1])
            if all(elmt in taken for elmt in preq):
                if (isTaken(array[i][0], taken) == False):
                    take = (array[i][0])
                    goingToTake += (array[i][0])
        if (isTaken(take, taken) == False):
            taken += goingToTake
            course[semester] += goingToTake
        semester += 1
    return course


# namaFile = input("Masukkan nama file : ")
namaFile = input("Masukkan nama file : ")
row = getRow(namaFile)
array = (fileTodaGraph(namaFile))
sortedDAG = selectionSort(array)
taken = takeCourse(sortedDAG)

for i in range(len(taken)):
    if len(taken[i]) != 0:
        print(f"Semester {i+1} :", end=" ")
        index = 0
        for j in taken[i]:
            if index == len(taken[i])-1:
                print(j)
            else:
                print(j, end=", ")
            index += 1

```

SCREENSHOT

```
(base) adinatha@insightharbor:~/Documents/Programming/Python/Penerapan Rencana Kuliah/src$ python3 1351
9048.py
Masukkan nama file : test1.txt
Semester 1 : C3
Semester 2 : C1
Semester 3 : C4
Semester 4 : C2
Semester 5 : C5
```

```
(base) adinatha@insightharbor:~/Documents/Programming/Python/Penerapan Rencana Kuliah/src$ python3 1351
9048.py
Masukkan nama file : test2.txt
Semester 1 : MA1101, FI1101, KU1001, KU1102, KU1011, KU1024, IF1210, KU1202, IF2121, IF2110, IF2120, IF
2124, IF2130, IF2211, IF2230, IF2240, IF2250, IF3140, IF3280, IF4091, KU2071, KU206X, AS2005
Semester 2 : MA1201, FI1201, EL1200, IF2123, IF2210, IF3130, IF3150, IF3151, IF4090, IF4092, IF3141, IF
3210
Semester 3 : IF3230, IF3110, IF3250, IF2220, IF3260
Semester 4 : IF3170
Semester 5 : IF3270
```

```
(base) adinatha@insightharbor:~/Documents/Programming/Python/Penerapan Rencana Kuliah/src$ python3 1351
9048.py
Masukkan nama file : test3.txt
Semester 1 : MA1101, FI1101, KU1001, KU1102, KU1011, KU1024
Semester 2 : MA1201, FI1201, IF1210, KU1202, KI1002, EL1200
Semester 3 : IF2124, IF2123, IF2130, IF2110, IF2120, IF2121
Semester 4 : IF2210, IF2211, IF2230, IF2240, IF2250, IF2220
Semester 5 : IF3130, IF3150, IF3140, IF3151, IF3110, IF3141
(base) adinatha@insightharbor:~/Documents/Programming/Python/Penerapan Rencana Kuliah/src$
```

```
(base) adinatha@insightharbor:~/Documents/Programming/Python/Penerapan Rencana Kuliah/src$ python3 1351
9048.py
Masukkan nama file : test4.txt
Semester 1 : Kalkulus, Fisika
Semester 2 : Matdis
Semester 3 : Kriptografi, TBFO, Stima
(base) adinatha@insightharbor:~/Documents/Programming/Python/Penerapan Rencana Kuliah/src$
```

```
(base) adinatha@insightharbor:~/Documents/Programming/Python/Penerapan Rencana Kuliah/src$ python3 1351
9048.py
Masukkan nama file : test5.txt
Semester 1 : KU1102, MA1101, FI1101, KU1011
Semester 2 : MA1201, FI1201, IF1210, KU1202, KI1002, EL1200
(base) adinatha@insightharbor:~/Documents/Programming/Python/Penerapan Rencana Kuliah/src$
```

```
(base) adinatha@insightharbor:~/Documents/Programming/Python/Penerapan Rencana Kuliah/src$ python3 1351
9048.py
Masukkan nama file : test6.txt
Semester 1 : MA1101, FI1101, KU1001, KU1102, KU1011, KU1024, IF1210, KU1202, IF2121, IF2110, IF2120, IF
2124, IF2130, IF2211, IF2230, IF2240, IF2250, IF3140, IF3280, IF4091, KU2071, KU206X, AS2005
Semester 2 : MA1201, FI1201, EL1200, IF2123, IF2210, IF3130, IF3150, IF3151, IF4090, IF4092, IF3141, IF
3210
Semester 3 : IF3230, IF3110, IF3250, IF2220, IF3260
Semester 4 : IF3170
Semester 5 : IF3270
(base) adinatha@insightharbor:~/Documents/Programming/Python/Penerapan Rencana Kuliah/src$
```

```
(base) adinatha@insightharbor:~/Documents/Programming/Python/Penerapan Rencana Kuliah/src$ python3 1351
9048.py
Masukkan nama file : test7.txt
Semester 1 : Bangun
Semester 2 : Makan, Kelas, Stress
Semester 3 : Praktikum, Belajar, Tugas
Semester 4 : Tubes
Semester 5 : Keos
Semester 6 : Meninggal
(base) adinatha@insightharbor:~/Documents/Programming/Python/Penerapan Rencana Kuliah/src$
```

```
(base) adinatha@insightharbor:~/Documents/Programming/Python/Penerapan Rencana Kuliah/src$ python3 1351
9048.py
Masukkan nama file : test8.txt
Semester 1 : Belajar, Olahraga, Organisasi, Rajin,
Semester 2 : Sehat, Budak, NilaiBagus, Berotot
Semester 3 : Ganteng, Gym, PunyaPacar
Semester 4 : Pelangi
```

LINK SOURCE CODE

https://github.com/ariyaadinatha/Course-Prequisite.git

Poin	Ya	Tidak
1. Program berhasil dikompilasi	CENTANG	
2. Program berhasil *running*	CENTANG	

3. Program dapat menerima berkas input dan menuliskan output	CETANG	
4. Luaran sudah benar untuk semua kasus input	CENTANG	
5. Saya bosen hidup	CENTANG	