MARMARA UNIVERSITY
FACULTY OF ENGINEERING

CSE2246

Analysis of Algorithm Assignment 2

Graph Coloring Problem

Student Id:	Name & Surname:
150119908	Mertcan Çiy
150119651	Tayfur Şafak Gençay
150119910	Samet Köser

Algorithm Description:

Our algorithm is used to color a graph. The algorithm consists of 3 parts. The first part is to take the inputs from file and after splitting line by line properly, we create a adjacency dictionary for vertices and making their initial color number “-1” to accept them as uncolored. In the second part, we start to color vertices. The plan for coloring is like this:

- Take the first vertex as starting vertex and make its color “0” instead of -1.

- In adjacency dictionary, compare all of its adjacents one by one by checking their colors. If the current vertex’ color is same with the adjacent that we compare, increase by 1. Else, continue to check other adjacents.

Finally, the third part is to edit and print the results. In the algorithm, we take a txt file as input and give a txt file as output. The input file must contain the following; letter p, vertices number and edges number on the first line, and letter e, vertex number 1 and vertex number 2. on the other lines, there must be a space between each value. The output file contains the following; The first row contains the number of colors found in the total, and the second row contains the number of each edge's color.

General Variables:

Dict<int, list<int>> dictOfAdjacency: the keys of the dictionary are the numbers of each vertex. Its values are also the number list of the vertices to which those vertices are connected.

Dict<int, int> verticesAndColors: The keys of the dictionary are the numbers of each vertex. their values are the color numbers of those vertices.

List<int> lengthList: list of all distinct color number.

List<int> colorsList: list of color numbers of all vertices.

Code Description:

We will describe the code with separating code.

```
with open('sample1.txt') as file:
    linesExceptFirst = file.readlines()[1:]
    for lines in linesExceptFirst:  # All lines except first line
        theLine = lines.strip()
        splittedTheLine = theLine.split(" ")

        if int(splittedTheLine[1]) in dictOfAdjacency:
            dictOfAdjacency[int(splittedTheLine[1])].append(int(splittedTheLine[2]))
        else:
            dictOfAdjacency[int(splittedTheLine[1])] = [int(splittedTheLine[2])]
            verticesAndColors[int(splittedTheLine[1])] = -1  # Make default color value of
vertices -1 to think them as uncolored

        if int(splittedTheLine[2]) not in dictOfAdjacency:
            dictOfAdjacency[int(splittedTheLine[2])] = [int(splittedTheLine[1])]
            verticesAndColors[int(splittedTheLine[2])] = -1
        else:
            dictOfAdjacency[int(splittedTheLine[2])].append(int(splittedTheLine[1]))
    itemsPair = dictOfAdjacency.items()
```

In this part we find the txt file and read it line by line except first line. We add all the lines in linesExceptFirst list. After that we go through all the elements one by one with the for loop. After separating the line by space we add them to splittedTheLine list and control the second and third element(First vertex and second vertex) of list if the element is already added to the dictOfAdjacency dictionary as a key, if it is, we add the other element to the value list of the element key in dictionary. If it is not, we create the element as a key and we create a list with other element as value of the element key. At the end of this code part, we get a dictionary like this example:

Exampleinput.txt:

```
p 5 5
e 1 2
e 2 3
e 2 4
e 3 4
e 4 5
```

Result after this code part:

```
{
5: [5,4],
1: [2],
2: [1,3,4],
3: [2,4],
4: [2,3,5]
}
```

Also we define the value of element key -1 in **verticesAndColors** dictionary to make all vertices' status "uncolored". After doing this for all vertices, we add all the keys and its values to a list and add this list to itemsPair.

```python
for verts, listOfAdj in itemsPair:
    counter = 0
    verticesAndColors[verts] = 0
    while counter < len(listOfAdj):
        for adjsOfCurrentVert in listOfAdj:
            try:
                if verticesAndColors[verts] == verticesAndColors[adjsOfCurrentVert]:
                    verticesAndColors[verts] += 1
            except:
                continue
        counter += 1
```

In this part of code we are looking for all the values of the itemsPair. for each list we define the first value verts and second value listOfAdj. After defining counter to zero and value of verts key 0 in verticesAndColors dictionary we started two nested loops. First loop loops length of listOfAdj times. Second loop loops in listOfAdj list and define each value as adjsOfCurrentVert. In these two loop we put a try except to prevent errors. If there is no error, code control values of vert key and adjsOfCurrentVert key in verticesAndColors. If they are equal, we add 1 to the value of verts key in verticesAndColors dictionary. If there is error we continue with next element of listOfAdj list.

```python
lengthList = []
colorsList = []
for a in sorted(verticesAndColors.keys()):
    colorsList.append(verticesAndColors[a])
    if verticesAndColors[a] not in lengthList:
        lengthList.append(verticesAndColors[a])

firstLine = len(lengthList)
strFirstLine = str(firstLine)

nextLine = "\n"
```

In this part of code we define the lengthList and colorsList. After that, we started a for loop in list of verticesAndColors we define each key as a in each loop. We add the a to colorsList. We control if the value of a key in verticesAndColors not in lengthList we add it to the lengthList. Then after doing these for all keys of verticesAndColors dictionary. We define an integer named firstLine with value of length of lengthList and we define an string named strFirstLine with the value of str version of firstLine. Lastly we define a string named nextLine with value of "\n".

```
with open('output1.txt', 'w') as f:
    f.write(strFirstLine)
    f.write(nextLine)
    for i in colorsList:
        f.write(str(i)+" ")
```

In this part of code we open output txt and we print strFirstLine then we go to next line with nextLine. Lastly we print all the elements of colorList separated with space.

Task Distribution:

We wrote the code together by meeting on Discord and in the report and readme file, we all distributed the tasks equally and wrote it.