Marmara University

Faculty of Engineering

Computer Engineering

CSE2046 - Analysis of Algorithms Homework 2

Yasin Tuğra Taş 150120048	Report, Visualization, Bug Fixing, Testing
Musa Meriç 150119751	Report, Implementation, Testing
Ayberk Türksoy 150119919	Report, Research, Optimization, Testing

Instructor: Ömer Korçak

Due Date: 07.06.2022

Algorithm

```c
int main(){
    int i = 0;
    FILE *input;
    input = fopen("input.txt","r+");
    char firstLine[1];
    int intinput1,intinput2;

    //Reading input file and making Graph
    if (input == NULL) {
        printf("Input file name should be input.txt");
        return;
    }
```

First of all, our program opens the input.txt on reading mode and checks it is whether empty or not. If it is empty, then returns an error message and exits.

```c
    else {
        fscanf(input,"%s %d %d", &firstLine[0], &intinput1, &intinput2);
        //printf("%s %d %d\n",firstLine,intinput1,intinput2);
        char pORe[intinput2-2][1];
        char adjMatrix[intinput1][intinput1];
        int j=0;
    //Set all values 0 in the adjacency matrix
    for(i=0;i<=intinput1-1;i++){
        for(j=0;j<=intinput1-1;j++){
            adjMatrix[i][j] = 0;
        }
    }//End of setting 0
```

After that, it creates an 2D array and fills its elements with 0 since it is empty at the start.

```c
    //Designing the adjacency matrix
        i=0;
        while (!feof(input)) {
            fscanf(input,"%s %d %d", &pORe[i], &intinput1, &intinput2);
            adjMatrix[intinput1-1][intinput2-1] = 1;
            //printf("%d %d\n",intinput1,intinput2);
            i++;
        }
        fclose(input);
```

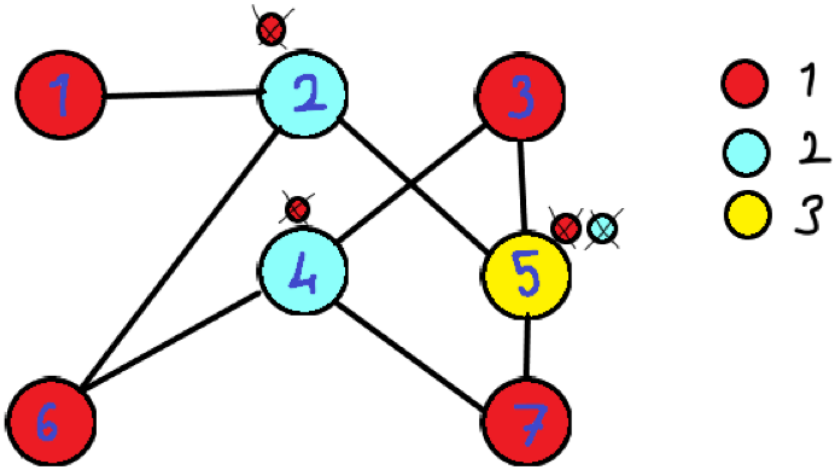
Then, it checks adjacency relations of all elements and turns the neighbours' into 1.

```
//Creating color graph and put default value as 0
int colorGraph[intinput1];
for(i=0;i<=intinput1-1;i++){
    colorGraph[i] = 0;
}


//Putting color values starting from 1
for(i=0;i<=intinput1-1;i++){
int j=-1;
int value = 1;

while(j<=(intinput1-1)){
j++;
if(adjMatrix[i][j] == 1){
    if(colorGraph[j] == value){
        value++;
        j = -1;
    }
}
}
colorGraph[i] = value;
}//End of putting color values.
```

In that section, it creates an array named colorGraph which stores the colors of elements. After that it checks every adjacency relation and tries to give different colors from its neighbours. It starts from 1 and increases until it finds a proper number, then assigns this number into according colorGraph element.



```
        //Finding how many different color used
        int max = 0;
        for(i=0;i<=intinput1-1;i++){
            if(colorGraph[i] > max){
                max = colorGraph[i];
            }
        }

        //Printing colorGraph
        printf("%d\n",max);
        for(i=0;i<=intinput1-1;i++){
            printf("%d ",colorGraph[i]-1);
        }
    //End of designing the adjacency matrix

    FILE * fp;

    fp = fopen ("output.txt", "w+");
    fprintf(fp, "%d\n",max);
        for(i=0;i<=intinput1-1;i++){
            fprintf(fp,"%d ",(colorGraph[i]-1));
        }
    fclose(fp);

    //End of printing function
    //End of reading File and Creating Graph

    return 0;
}}
```

After all that process, it finds how many colors are used and prints the color assigned to each node.

Resources

- https://www.youtube.com/watch?v=052VkKhIaQ4
- Lecture records and notes.