# ISTANBUL TECHNICAL UNIVERSITY

# COMPUTER ENGINEERING DEPARTMENT

# BLG 223E

# DATA STRUCTURES AND ALGORITHMS HOMEWORK 3 REPORT

| | | |
|---|---|---|
| **HOMEWORK NO** | : | 3 |
| **SUBMISSION DATE** | : | 10.07.2024 |
| **NAME** | : | İrem Kalay |
| **STUDENT ID** | : | 820220334 |

**SPRING 2024**

# Contents

# 1 General Layout of Homework

In this 3rd homework, we are given a graph and we were wanted to perform the wanted operations as finding the number of neighbor nodes of a particular node and printing them in part1, finding degree centrality in part2 and finding the shortest distance (shortest path) that two nodes have in part3. In order to understand the structure , at first I created a smaller tsv file and tried to grasp the idea. Then I controlled my functions with the given big size tsv file. I also created a nameMap to store the names of the nodes.

# 2 Methadology of Parts

## 2.1 Part1: Hello neighbor

In order to understand where the neighbours are stored, firstly I needed to grasp the "Node" structure. The neighbours were being kept in the adj vector of a node. Firstly I checked wheteher that searched node is exists or not and if it is exists , I accessed my node as grapgh[MID] and then: I took the size of adj vector for determining the number of neighbors later on, I traverse through the adj vector and printed all the neighbours.

## 2.2 Part2: Degree Centrality

In order to find the degree centrality, I needed to find how many connection a particular node has. For that purpose , I created a vector called centrality that stores the mid and degree in pair. I traverse through the graph map and take the degree as number of elements of the adj size then I pushed the pair to my vector-centrality. Then I sorted the vector in descending order because in this part we were expected to print the top 10 nodes with highest degree centrality. and by traversing the first 10 element of vector, I printed the nodes that have the highest degree centrality with their degrees.

## 2.3 Part3: Shortest Distance

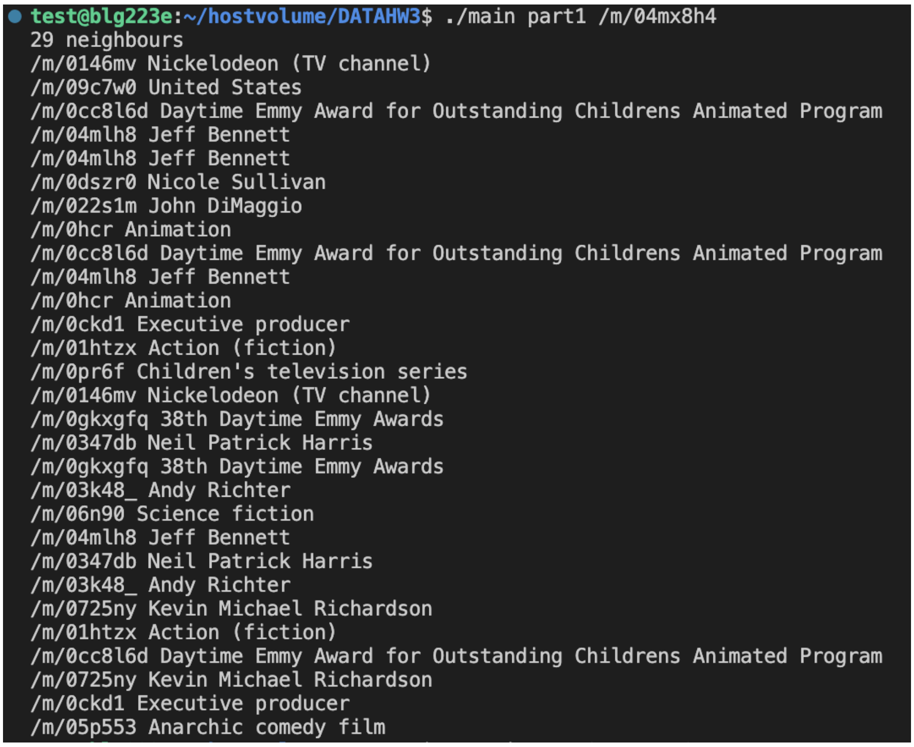
In this part, firstly I checked whether the entered nodes are available in the graph map or not. Then I used the BFS algorithm to find the shortest path, because when I use BFS , it traverses level by level because of this property when it first faced with the searched MID , it finds the shortest path. I push the started node to my queue and make it s visited true initially. Then while my queue is not empty I take current as the queue's front then I pop front. I start to search for the second MID that entered by user also with an if statement. If I find the second MID, I print the shortest path to the terminal.

Else, I check visited nodes and push mid of neighbor of the graphmap[current]-¿adj and sign as visited[neighbor-¿MID]=true.

# 3 Results

## 3.1 Part 1

Finding neighbors of /m/04mx8h4 can be seen in the below picture. Ashieved by running ./main part1 /m/04mx8h4.

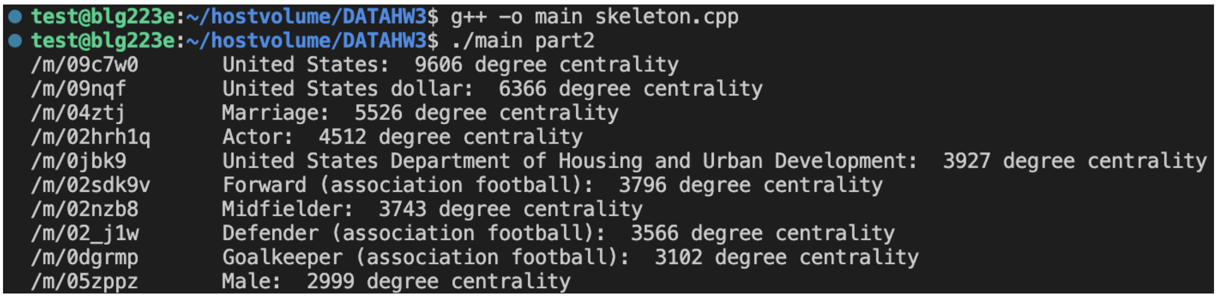


```
● test@blg223e:~/hostvolume/DATAHW3$ ./main part1 /m/04mx8h4
  29 neighbours
  /m/0146mv Nickelodeon (TV channel)
  /m/09c7w0 United States
  /m/0cc8l6d Daytime Emmy Award for Outstanding Childrens Animated Program
  /m/04mlh8 Jeff Bennett
  /m/04mlh8 Jeff Bennett
  /m/0dszr0 Nicole Sullivan
  /m/022s1m John DiMaggio
  /m/0hcr Animation
  /m/0cc8l6d Daytime Emmy Award for Outstanding Childrens Animated Program
  /m/04mlh8 Jeff Bennett
  /m/0hcr Animation
  /m/0ckd1 Executive producer
  /m/01htzx Action (fiction)
  /m/0pr6f Children's television series
  /m/0146mv Nickelodeon (TV channel)
  /m/0gkxgfq 38th Daytime Emmy Awards
  /m/0347db Neil Patrick Harris
  /m/0gkxgfq 38th Daytime Emmy Awards
  /m/03k48_ Andy Richter
  /m/06n90 Science fiction
  /m/04mlh8 Jeff Bennett
  /m/0347db Neil Patrick Harris
  /m/03k48_ Andy Richter
  /m/0725ny Kevin Michael Richardson
  /m/01htzx Action (fiction)
  /m/0cc8l6d Daytime Emmy Award for Outstanding Childrens Animated Program
  /m/0725ny Kevin Michael Richardson
  /m/0ckd1 Executive producer
  /m/05p553 Anarchic comedy film
```

Figure 1: Finding neighbors

### 3.2 Part 2

I found the most central 10 nodes as follows with their degree centralities.

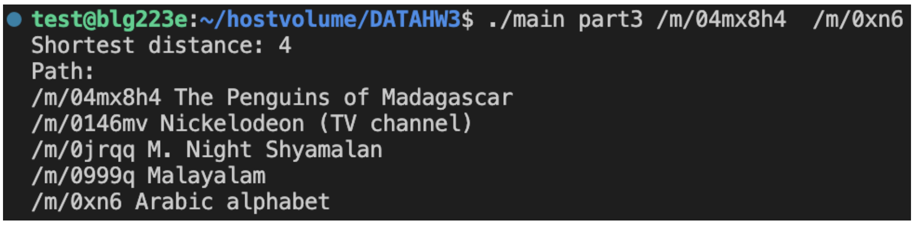


```
● test@blg223e:~/hostvolume/DATAHW3$ g++ -o main skeleton.cpp
● test@blg223e:~/hostvolume/DATAHW3$ ./main part2
  /m/09c7w0       United States:  9606 degree centrality
  /m/09nqf        United States dollar:  6366 degree centrality
  /m/04ztj        Marriage:  5526 degree centrality
  /m/02hrh1q      Actor:  4512 degree centrality
  /m/0jbk9        United States Department of Housing and Urban Development:  3927 degree centrality
  /m/02sdk9v      Forward (association football):  3796 degree centrality
  /m/02nzb8       Midfielder:  3743 degree centrality
  /m/02_j1w       Defender (association football):  3566 degree centrality
  /m/0dgrmp       Goalkeeper (association football):  3102 degree centrality
  /m/05zppz       Male:  2999 degree centrality
```

Figure 2: Top 10 highest node centrality

### 3.3 Part 3

Here I found shortest distance between the nodes: /m/04mx8h4 and /m/0xn6 and write down the path between them in the terminal.



```
● test@blg223e:~/hostvolume/DATAHW3$ ./main part3 /m/04mx8h4  /m/0xn6
  Shortest distance: 4
  Path:
  /m/04mx8h4 The Penguins of Madagascar
  /m/0146mv Nickelodeon (TV channel)
  /m/0jrqq M. Night Shyamalan
  /m/0999q Malayalam
  /m/0xn6 Arabic alphabet
```

Figure 3: Shortest distance between 04mx8h4 and 0xn6

## 4 Conclusion

All in all , I could be able to perform the wanted operations for each parts. At first I did not understand the structure well. However later on, I examined the provided code and investigated further inside the tsv file then noticed the patterns that the given code has. Then I started to write my functions easily. In the first part, I printed the number of neighbors and names on terminal. In the second part, printed the most central 10 nodes and lastly in the third part of the homework, I printed the shortest distance between two nodes and printed the path between them.