CSE 222

HOMEWORK #8

Arife YURTSEVEN

Introduction

"In this assignment, I wrote a program for social network analysis. In this assignment, I provide a range of functionalities to manage relationships and interactions between individuals in the social network. These functionalities include adding and removing people, establishing and removing friendship ties, finding the shortest path, suggesting friends, and determining the number of clusters. The program also has a menu that allows users to choose from various operations.

The classes in the program include:

Person.java

SocialNetwork.java

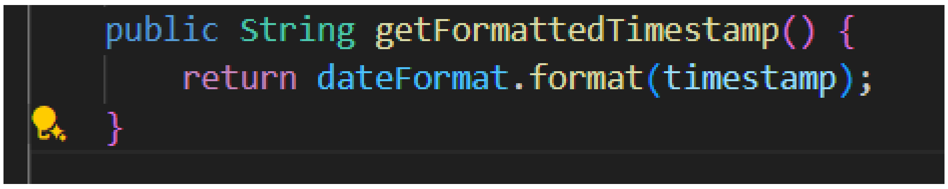
Main.java

Person.java

The Person class represents an individual in the social network. Each person has information such as name, age, a list of hobbies, and a creation timestamp. This class provides methods for managing and accessing an individual's information.

In the code, the expression private static final SimpleDateFormat dateFormat creates an instance of this class. The format template used here is "yyyy-MM-dd HH:mm:ss". This template displays the year in four digits, the month and day in two digits each, and the time in hour:minut format.

The getFormattedTimestamp() method returns the timestamp of an instance of the Person class as a formatted string. This method uses the format method of the dateFormat object to convert the timestamp object into the format specified above.



```
    public String getFormattedTimestamp() {
        return dateFormat.format(timestamp);
    }
```

SocialNetwork.java

The SocialNetwork class represents the social network itself. This class manages individuals and their relationships within the network. People and their friendships are stored and managed using nested data structures. Additionally, the class includes methods that allow for various analyses on the network, such as finding the shortest path, suggesting friends, and counting clusters.

addPerson Method

In this program, I add a new individual to the social network. The person's name, age, and hobbies are taken as parameters. A unique timestamp is assigned to each individual, and this person is registered in the people map with a unique key combining their name and timestamp. An empty list is initialized for their list of friends.

removePerson Metodu

This method removes an individual from the social network based on the specified name and timestamp. The person is removed from the people map, and all associated friendship connections are updated in the friendships map.

addFriendship Metodu

This method establishes a friendship between two individuals. The names and timestamps of both individuals are taken, and if both are found in the system, they are mutually added to each other's list of friends.

removeFriendship Metodu

This method removes the friendship relationship between two specified individuals. For each, the other is removed from their list of friends. If the process is successful, information about the removal of the friendship is printed to the console.

findShortestPath Metodu

This method finds the shortest path between two individuals using the Breadth-First Search (BFS) algorithm. Names and timestamps of the individuals are taken as parameters. If the start and end individuals are not found, an error message is given. Otherwise, the BFS algorithm is applied to calculate the shortest path, and this path is printed using the printPath method.

Bfs_short Metodu

The necessary data structures for BFS include a queue (queue), a visited set (visited), and a map (prev) that stores the predecessor of each individual. The start point is added to the queue and marked as visited. Until the queue is empty, individuals are removed from the queue (dequeue). Each removed individual is checked if it is the end point. If it is the endpoint, the printPath function prints the found path and terminates the function. The neighbors (friend list) of the removed individual are checked. If a neighbor has not been visited before:

This neighbor is added to the queue.

It is marked as visited.

Information on which individual this neighbor was reached from is added to the prev map.

printPath Metodu

This helper method prints the shortest path found by findShortestPath. The path, consisting of the names of individuals, is printed as a string to the console.

suggestFriends Metodu

This method generates friend suggestions for a specific individual. The person's name, timestamp, and maximum number of suggestions are taken. If the person is not found, an error message is given. Otherwise, potential friends are evaluated according to a scoring system, and those with the highest scores are suggested. Each suggestion includes the friend's name, score, number of mutual friends, and number of common hobbies.

Setting Up the Scoring System:

A HashMap named scores is created. This map stores the calculated scores and relevant information (FriendSuggestion) for each suggested individual. All individuals in the network (people.values()) are iterated over, and the following conditions are evaluated for each:

Score Calculation:

Mutual Friends: Each friend of the current individual (person) is compared with the suggested individual (p). If the suggested individual’s friend list includes the current individual’s friend, 1 point is added to the score, and the number of mutual friends is increased.

Common Hobbies: The hobbies of the suggested individual and the current individual are compared. For each common hobby, 0.5 points are added to the score, and the number of common hobbies is increased.

Score Recording: If the calculated score is greater than 0, the suggested individual and the corresponding FriendSuggestion object are added to the scores map.

Note: Whether there are mutual friends is not crucial in suggesting friends. If the score is high, they may be recommended.

countClusters Metodu

This method counts the number of disconnected groups (clusters) in the social network. Each disconnected group is discovered using BFS and added to a list. The total number of clusters and the members of each cluster are printed to the console.

bfs Metodu

This helper method, used by countClusters, finds all individuals accessible from a starting individual. These individuals are considered as a cluster and added to the corresponding list.

Main.java

The Main class instantiates a SocialNetwork object and uses a Scanner object to interact with the user. This class executes various social network operations in a loop that lasts until the user terminates the program. These actions include adding or removing people, managing friendships, and network analysis.

Functions and Method Definitions

1. Initializing Network and Login Method:

A Scanner instance is created for the SocialNetwork object and the input. Predefined individuals and people with hobbies are added for system demonstration.

2. Menu System:

The user is presented with a menu to add contacts, remove contacts, add friendships, remove friendships, find the shortest path, suggest friends, count clusters, and exit. The user's choice is handled with the switch-case statement.

3. Error Management:

The selection the user enters is parsed as a numerical value. If parsing fails due to invalid input, the selection is marked as invalid, the error message is issued, and the menu is displayed again

How to Run The Code?

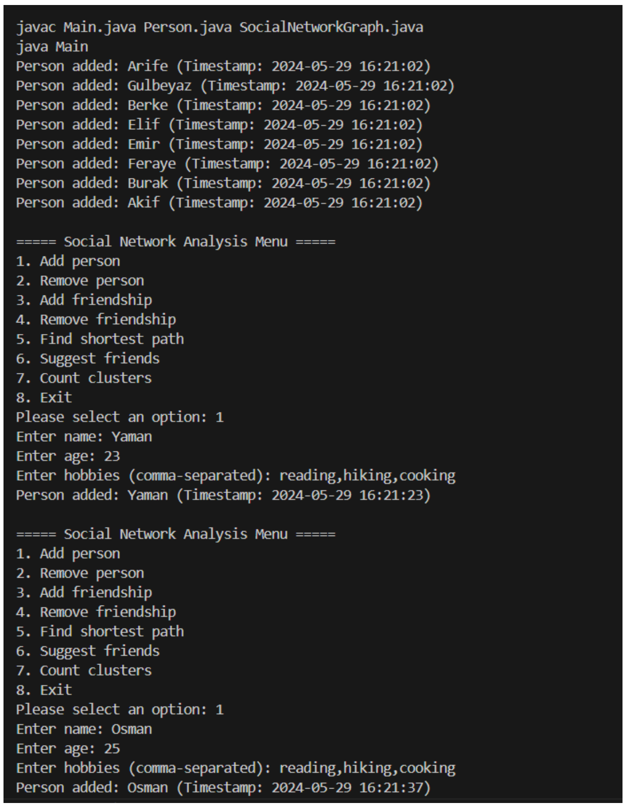
make: Type “make all” to compile the code.

make clean: Type “make clean" to clean the output files.

make doc: Type “make doc” to create Javadoc files.

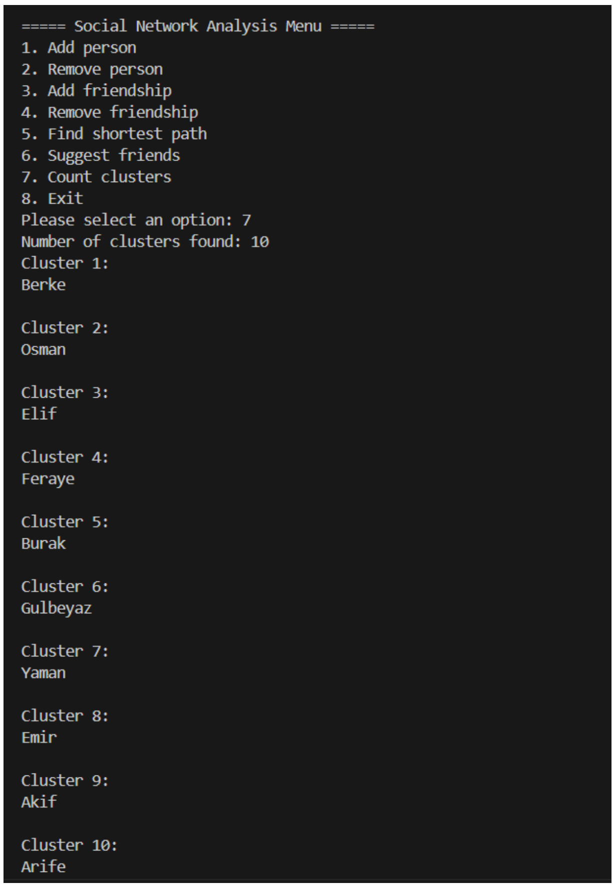
make cleandoc: Type “make cleandoc” to delete Javadoc files.

ADD (Yaman and Osman)



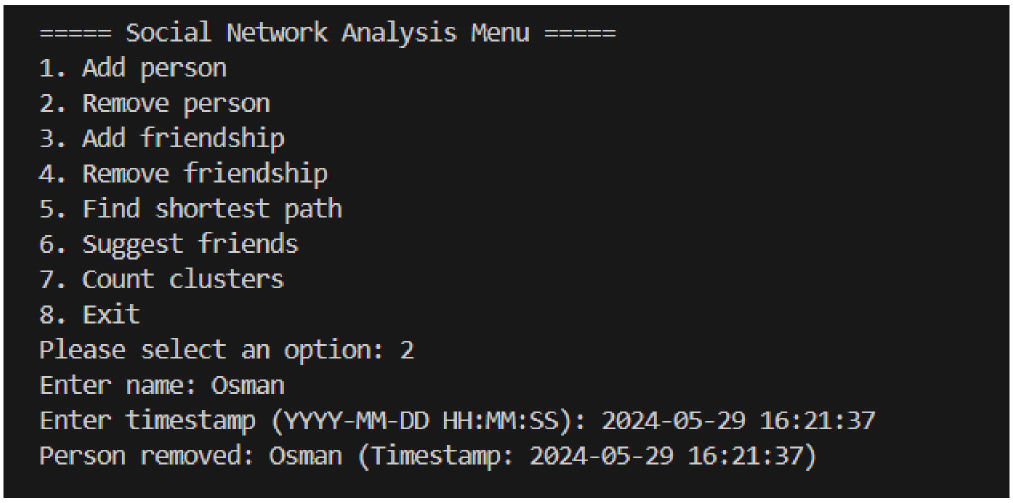
```
javac Main.java Person.java SocialNetworkGraph.java
java Main
Person added: Arife (Timestamp: 2024-05-29 16:21:02)
Person added: Gulbeyaz (Timestamp: 2024-05-29 16:21:02)
Person added: Berke (Timestamp: 2024-05-29 16:21:02)
Person added: Elif (Timestamp: 2024-05-29 16:21:02)
Person added: Emir (Timestamp: 2024-05-29 16:21:02)
Person added: Feraye (Timestamp: 2024-05-29 16:21:02)
Person added: Burak (Timestamp: 2024-05-29 16:21:02)
Person added: Akif (Timestamp: 2024-05-29 16:21:02)

===== Social Network Analysis Menu =====
1. Add person
2. Remove person
3. Add friendship
4. Remove friendship
5. Find shortest path
6. Suggest friends
7. Count clusters
8. Exit
Please select an option: 1
Enter name: Yaman
Enter age: 23
Enter hobbies (comma-separated): reading,hiking,cooking
Person added: Yaman (Timestamp: 2024-05-29 16:21:23)

===== Social Network Analysis Menu =====
1. Add person
2. Remove person
3. Add friendship
4. Remove friendship
5. Find shortest path
6. Suggest friends
7. Count clusters
8. Exit
Please select an option: 1
Enter name: Osman
Enter age: 25
Enter hobbies (comma-separated): reading,hiking,cooking
Person added: Osman (Timestamp: 2024-05-29 16:21:37)
```

Count Clusters



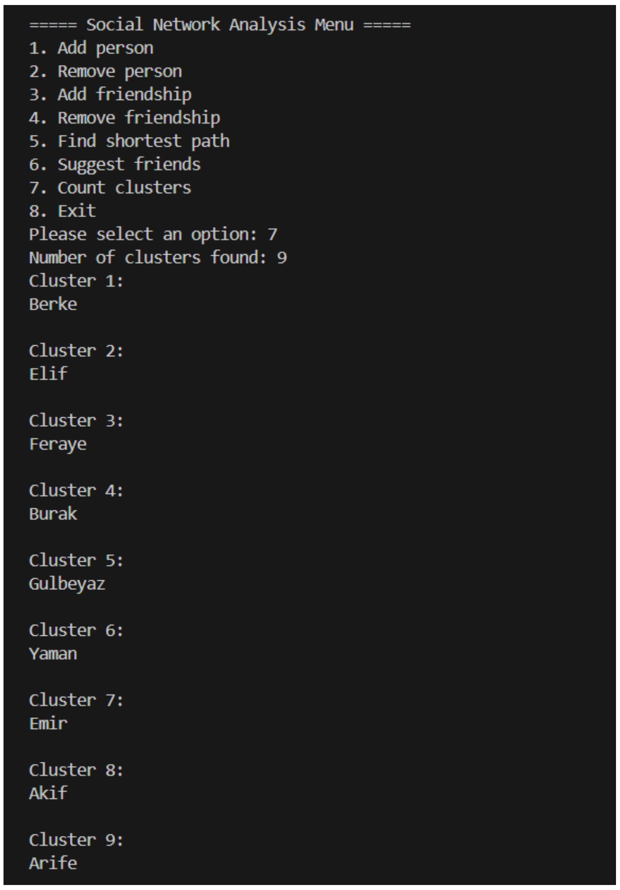
```
===== Social Network Analysis Menu =====
1. Add person
2. Remove person
3. Add friendship
4. Remove friendship
5. Find shortest path
6. Suggest friends
7. Count clusters
8. Exit
Please select an option: 7
Number of clusters found: 10
Cluster 1:
Berke

Cluster 2:
Osman

Cluster 3:
Elif

Cluster 4:
Feraye

Cluster 5:
Burak

Cluster 6:
Gulbeyaz

Cluster 7:
Yaman

Cluster 8:
Emir

Cluster 9:
Akif

Cluster 10:
Arife
```

REMOVE (osman)



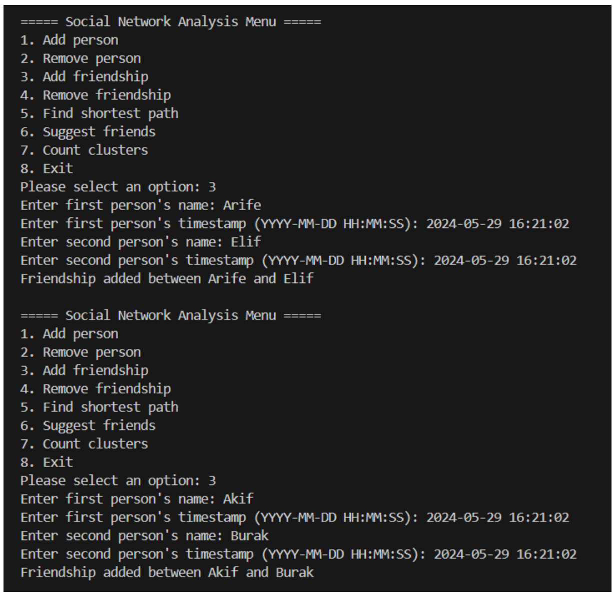
```
===== Social Network Analysis Menu =====
1. Add person
2. Remove person
3. Add friendship
4. Remove friendship
5. Find shortest path
6. Suggest friends
7. Count clusters
8. Exit
Please select an option: 2
Enter name: Osman
Enter timestamp (YYYY-MM-DD HH:MM:SS): 2024-05-29 16:21:37
Person removed: Osman (Timestamp: 2024-05-29 16:21:37)
```

COUNT CLUSTER

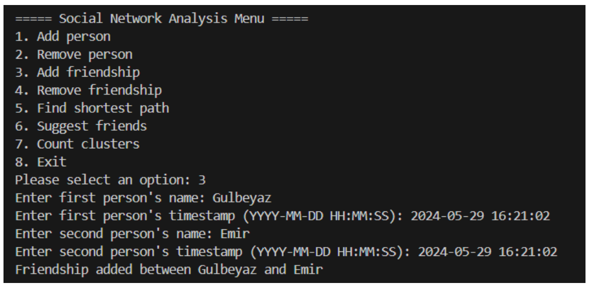


```
===== Social Network Analysis Menu =====
1. Add person
2. Remove person
3. Add friendship
4. Remove friendship
5. Find shortest path
6. Suggest friends
7. Count clusters
8. Exit
Please select an option: 7
Number of clusters found: 9
Cluster 1:
Berke

Cluster 2:
Elif

Cluster 3:
Feraye

Cluster 4:
Burak

Cluster 5:
Gulbeyaz

Cluster 6:
Yaman

Cluster 7:
Emir

Cluster 8:
Akif

Cluster 9:
Arife
```

ADD FRIENDSHIP(Arife-Elif)(Akif-Burak) (Gülbeyaz-Emir)

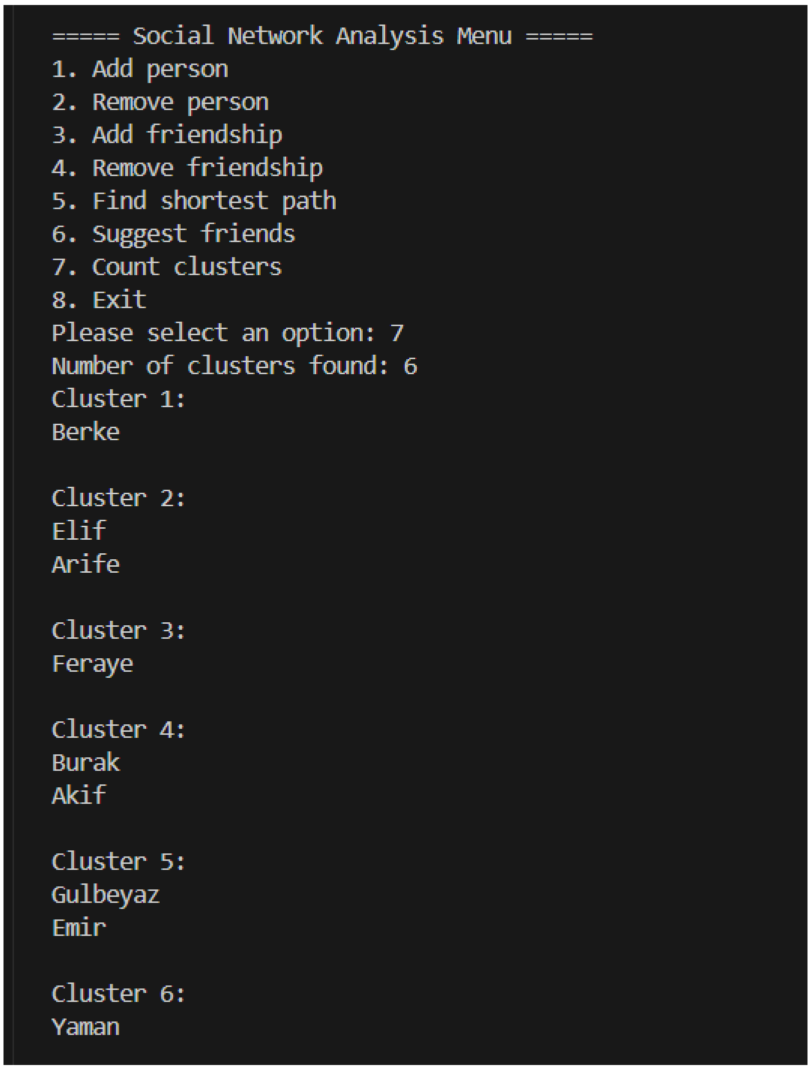


```
===== Social Network Analysis Menu =====
1. Add person
2. Remove person
3. Add friendship
4. Remove friendship
5. Find shortest path
6. Suggest friends
7. Count clusters
8. Exit
Please select an option: 3
Enter first person's name: Arife
Enter first person's timestamp (YYYY-MM-DD HH:MM:SS): 2024-05-29 16:21:02
Enter second person's name: Elif
Enter second person's timestamp (YYYY-MM-DD HH:MM:SS): 2024-05-29 16:21:02
Friendship added between Arife and Elif

===== Social Network Analysis Menu =====
1. Add person
2. Remove person
3. Add friendship
4. Remove friendship
5. Find shortest path
6. Suggest friends
7. Count clusters
8. Exit
Please select an option: 3
Enter first person's name: Akif
Enter first person's timestamp (YYYY-MM-DD HH:MM:SS): 2024-05-29 16:21:02
Enter second person's name: Burak
Enter second person's timestamp (YYYY-MM-DD HH:MM:SS): 2024-05-29 16:21:02
Friendship added between Akif and Burak
```



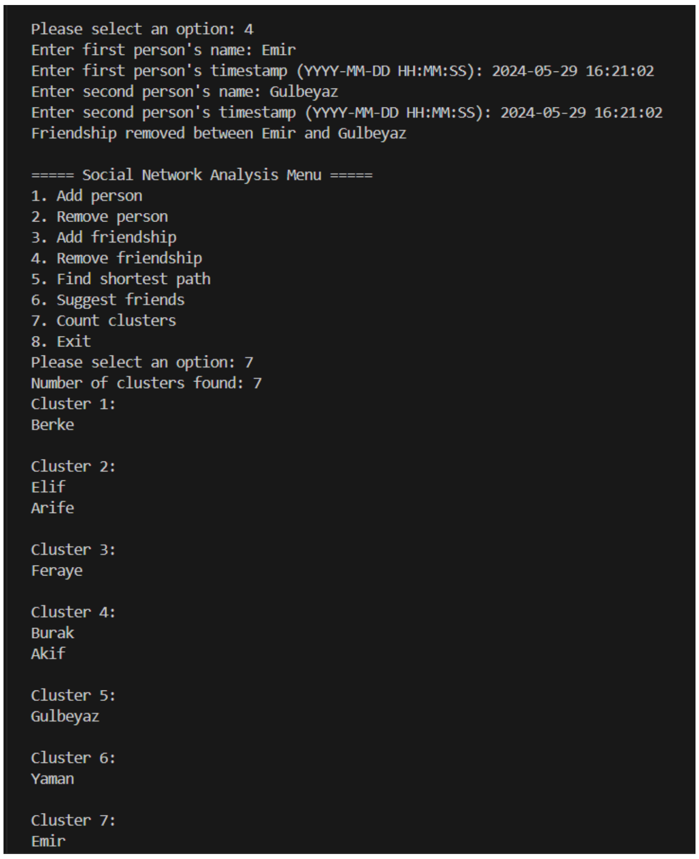
```
===== Social Network Analysis Menu =====
1. Add person
2. Remove person
3. Add friendship
4. Remove friendship
5. Find shortest path
6. Suggest friends
7. Count clusters
8. Exit
Please select an option: 3
Enter first person's name: Gulbeyaz
Enter first person's timestamp (YYYY-MM-DD HH:MM:SS): 2024-05-29 16:21:02
Enter second person's name: Emir
Enter second person's timestamp (YYYY-MM-DD HH:MM:SS): 2024-05-29 16:21:02
Friendship added between Gulbeyaz and Emir
```

COUNT CLUSTERS



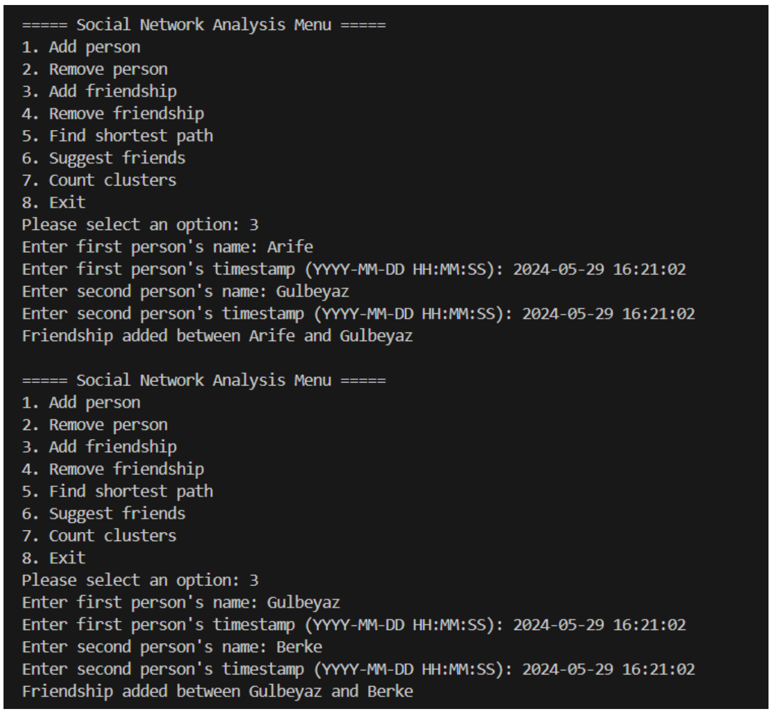
```
===== Social Network Analysis Menu =====
1. Add person
2. Remove person
3. Add friendship
4. Remove friendship
5. Find shortest path
6. Suggest friends
7. Count clusters
8. Exit
Please select an option: 7
Number of clusters found: 6
Cluster 1:
Berke

Cluster 2:
Elif
Arife

Cluster 3:
Feraye

Cluster 4:
Burak
Akif

Cluster 5:
Gulbeyaz
Emir

Cluster 6:
Yaman
```

REMOVE FRIENDSHIP(Gulbeyaz-Emir)



```
Please select an option: 4
Enter first person's name: Emir
Enter first person's timestamp (YYYY-MM-DD HH:MM:SS): 2024-05-29 16:21:02
Enter second person's name: Gulbeyaz
Enter second person's timestamp (YYYY-MM-DD HH:MM:SS): 2024-05-29 16:21:02
Friendship removed between Emir and Gulbeyaz

===== Social Network Analysis Menu =====
1. Add person
2. Remove person
3. Add friendship
4. Remove friendship
5. Find shortest path
6. Suggest friends
7. Count clusters
8. Exit
Please select an option: 7
Number of clusters found: 7
Cluster 1:
Berke

Cluster 2:
Elif
Arife

Cluster 3:
Feraye

Cluster 4:
Burak
Akif

Cluster 5:
Gulbeyaz

Cluster 6:
Yaman

Cluster 7:
Emir
```

ADD FRIENDSHIP(Gulbeyaz-Arife)(Gulbeyaz-Berke)



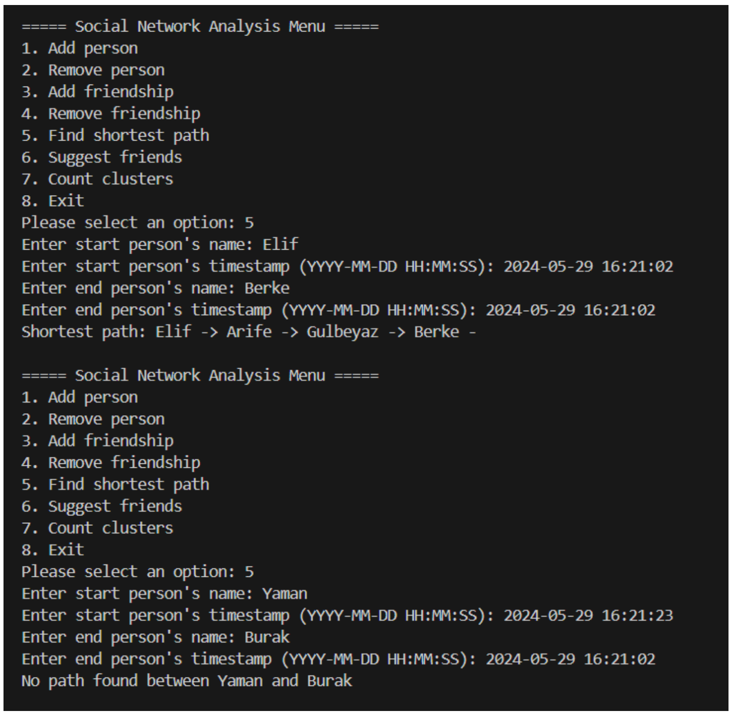
```
===== Social Network Analysis Menu =====
1. Add person
2. Remove person
3. Add friendship
4. Remove friendship
5. Find shortest path
6. Suggest friends
7. Count clusters
8. Exit
Please select an option: 3
Enter first person's name: Arife
Enter first person's timestamp (YYYY-MM-DD HH:MM:SS): 2024-05-29 16:21:02
Enter second person's name: Gulbeyaz
Enter second person's timestamp (YYYY-MM-DD HH:MM:SS): 2024-05-29 16:21:02
Friendship added between Arife and Gulbeyaz

===== Social Network Analysis Menu =====
1. Add person
2. Remove person
3. Add friendship
4. Remove friendship
5. Find shortest path
6. Suggest friends
7. Count clusters
8. Exit
Please select an option: 3
Enter first person's name: Gulbeyaz
Enter first person's timestamp (YYYY-MM-DD HH:MM:SS): 2024-05-29 16:21:02
Enter second person's name: Berke
Enter second person's timestamp (YYYY-MM-DD HH:MM:SS): 2024-05-29 16:21:02
Friendship added between Gulbeyaz and Berke
```

FIND SHORTEST PATH(Elif-Berke)(Yaman-Burak)

Elif-Arife

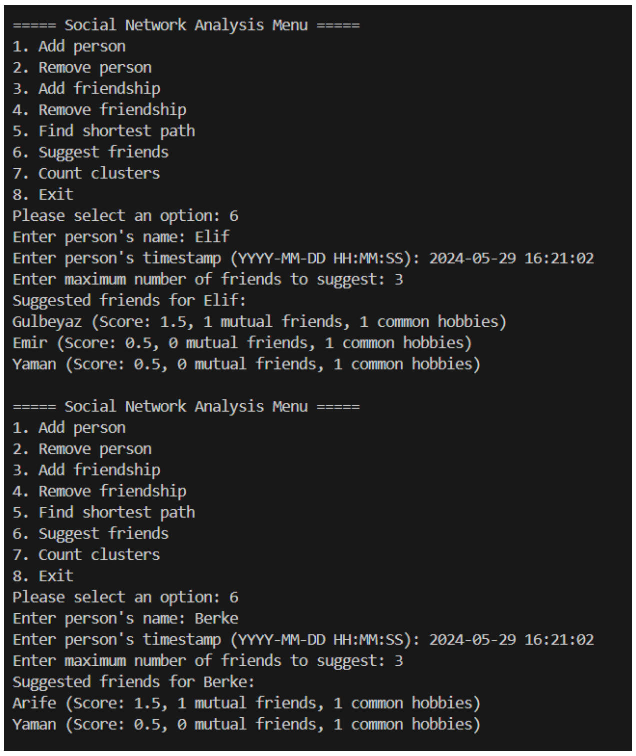
Arife-Gülbeyaz

Gülbeyaz-Berke



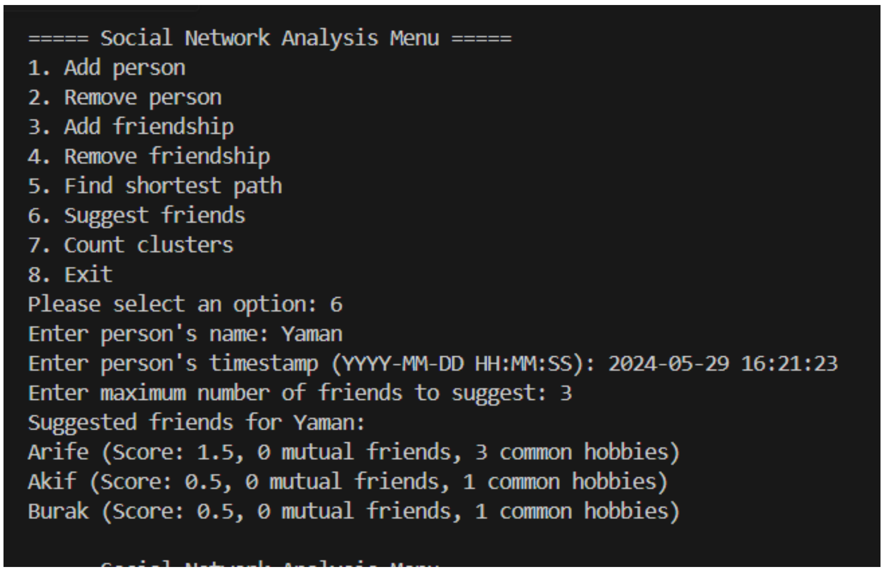
```
===== Social Network Analysis Menu =====
1. Add person
2. Remove person
3. Add friendship
4. Remove friendship
5. Find shortest path
6. Suggest friends
7. Count clusters
8. Exit
Please select an option: 5
Enter start person's name: Elif
Enter start person's timestamp (YYYY-MM-DD HH:MM:SS): 2024-05-29 16:21:02
Enter end person's name: Berke
Enter end person's timestamp (YYYY-MM-DD HH:MM:SS): 2024-05-29 16:21:02
Shortest path: Elif -> Arife -> Gulbeyaz -> Berke -

===== Social Network Analysis Menu =====
1. Add person
2. Remove person
3. Add friendship
4. Remove friendship
5. Find shortest path
6. Suggest friends
7. Count clusters
8. Exit
Please select an option: 5
Enter start person's name: Yaman
Enter start person's timestamp (YYYY-MM-DD HH:MM:SS): 2024-05-29 16:21:23
Enter end person's name: Burak
Enter end person's timestamp (YYYY-MM-DD HH:MM:SS): 2024-05-29 16:21:02
No path found between Yaman and Burak
```

SUGGEST FRIENDS(Elif)(Berke)



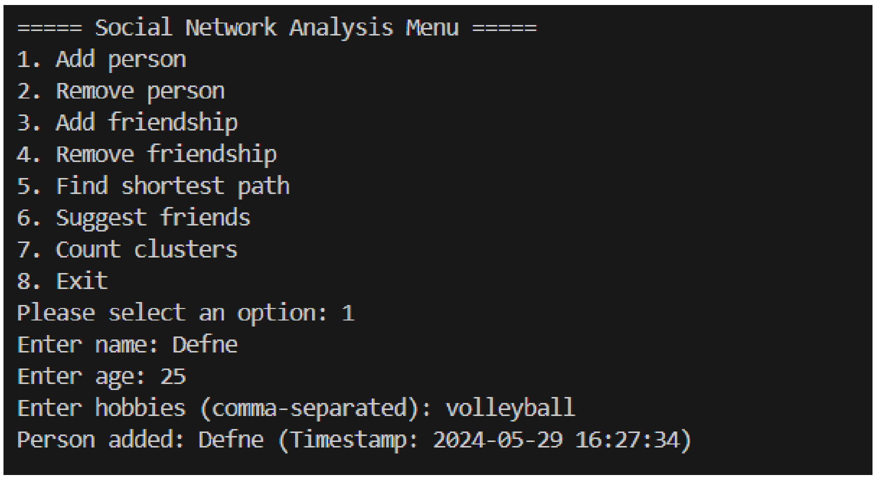
```
===== Social Network Analysis Menu =====
1. Add person
2. Remove person
3. Add friendship
4. Remove friendship
5. Find shortest path
6. Suggest friends
7. Count clusters
8. Exit
Please select an option: 6
Enter person's name: Elif
Enter person's timestamp (YYYY-MM-DD HH:MM:SS): 2024-05-29 16:21:02
Enter maximum number of friends to suggest: 3
Suggested friends for Elif:
Gulbeyaz (Score: 1.5, 1 mutual friends, 1 common hobbies)
Emir (Score: 0.5, 0 mutual friends, 1 common hobbies)
Yaman (Score: 0.5, 0 mutual friends, 1 common hobbies)

===== Social Network Analysis Menu =====
1. Add person
2. Remove person
3. Add friendship
4. Remove friendship
5. Find shortest path
6. Suggest friends
7. Count clusters
8. Exit
Please select an option: 6
Enter person's name: Berke
Enter person's timestamp (YYYY-MM-DD HH:MM:SS): 2024-05-29 16:21:02
Enter maximum number of friends to suggest: 3
Suggested friends for Berke:
Arife (Score: 1.5, 1 mutual friends, 1 common hobbies)
Yaman (Score: 0.5, 0 mutual friends, 1 common hobbies)
```

SUGGEST FRIENDS(Yaman)



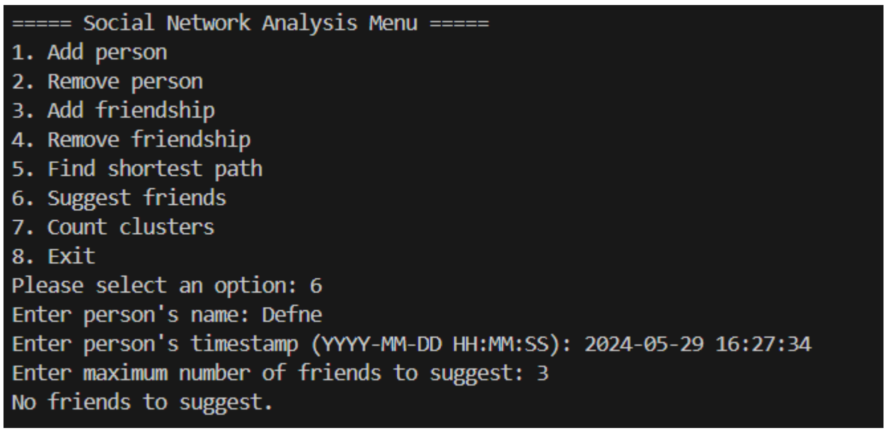
```
===== Social Network Analysis Menu =====
1. Add person
2. Remove person
3. Add friendship
4. Remove friendship
5. Find shortest path
6. Suggest friends
7. Count clusters
8. Exit
Please select an option: 6
Enter person's name: Yaman
Enter person's timestamp (YYYY-MM-DD HH:MM:SS): 2024-05-29 16:21:23
Enter maximum number of friends to suggest: 3
Suggested friends for Yaman:
Arife (Score: 1.5, 0 mutual friends, 3 common hobbies)
Akif (Score: 0.5, 0 mutual friends, 1 common hobbies)
Burak (Score: 0.5, 0 mutual friends, 1 common hobbies)
```

ADD PERSON(Defne)



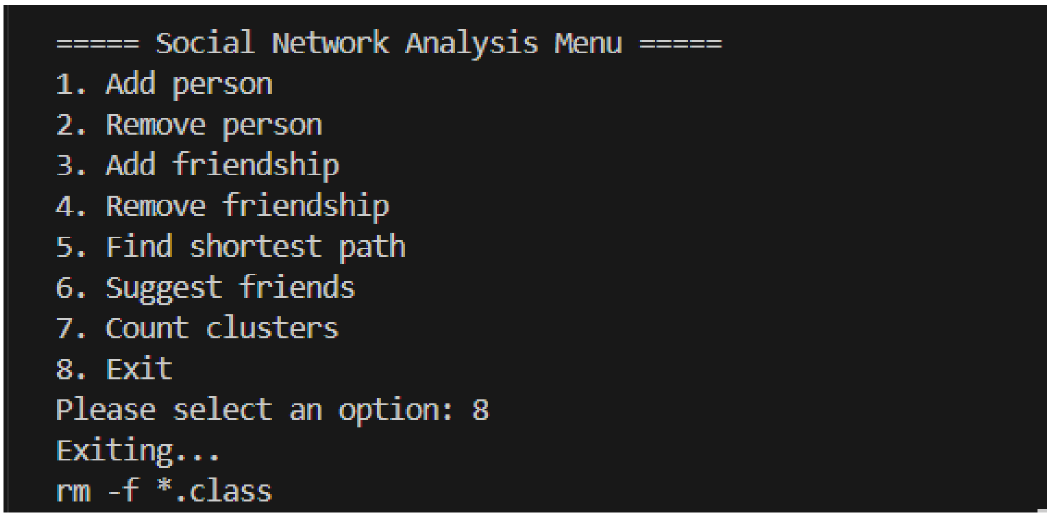
```
===== Social Network Analysis Menu =====
1. Add person
2. Remove person
3. Add friendship
4. Remove friendship
5. Find shortest path
6. Suggest friends
7. Count clusters
8. Exit
Please select an option: 1
Enter name: Defne
Enter age: 25
Enter hobbies (comma-separated): volleyball
Person added: Defne (Timestamp: 2024-05-29 16:27:34)
```

SUGGEST FRIENDS(Defne)



```
===== Social Network Analysis Menu =====
1. Add person
2. Remove person
3. Add friendship
4. Remove friendship
5. Find shortest path
6. Suggest friends
7. Count clusters
8. Exit
Please select an option: 6
Enter person's name: Defne
Enter person's timestamp (YYYY-MM-DD HH:MM:SS): 2024-05-29 16:27:34
Enter maximum number of friends to suggest: 3
No friends to suggest.
```

EXIT



```
===== Social Network Analysis Menu =====
1. Add person
2. Remove person
3. Add friendship
4. Remove friendship
5. Find shortest path
6. Suggest friends
7. Count clusters
8. Exit
Please select an option: 8
Exiting...
rm -f *.class
```