Assignment 1

Case Studies

1. Formulating the Problem

1.1 Problem Description

Design and implement a Java program for a health club to create a system that will keep track of patron's data collected from their Fitbit device.

1.2 Verbalization

What is the goal?

Store patron's data in a list array.

What are the givens?

Calories consumed

Distance walked in miles

Number of floors climbed

Number of steps taken

Average heart rate

1.3 Information Elicitation

Goal

To collect user's data using keyboard input. Store the collected data as an object. Print all the data for user's review. Add a new patron's data to the list.

Givens

Calories consumed

Distance walked in miles

Number of floors climbed

Number of steps taken

Average heart rate

Unknowns

None

Conditions

None

2. Planning the Solution

2.1 Solution Strategy

Get all values from user's input and store them in ListArray of objects. Create the architecture using the singleton design pattern. This pattern allows to create only a single class which is responsible for making sure that only single object gets created.

2.2 Goal Decomposition

Sub-goal 1

Get data from the user.

Sub-goal 2

Add data to the existing ListArray

Sub-goal 3

Display all data.

2.3 Resources

Relevant formulas
None

2.4 Data Organization and Description

Input (givens):

Name	Description	Origin	Used in Sub-goal #
Full Name	User's full name	User	1
Calories Consumed	Calories consumed by user	User	1
Distance Walked(miles)	Distance walked by user	User	1
Number of floors climbed	Number of floors climbed by user	User	1
Number of steps taken	Number of steps taken by user	User	1
Average Heart Rate	Average heard rate of user	User	1

Output (unknowns):

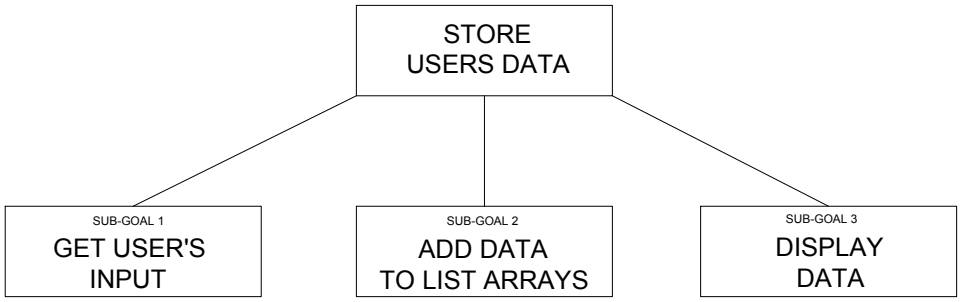
Name	Description	Origin	Used in Sub-goal #
Full Name	User's full name	Screen	3
Calories Consumed	Calories consumed by user	Screen	3
Distance Walked(miles)	Distance walked by user	Screen	3
Number of floors climbed	Number of floors climbed by user	Screen	3
Number of steps taken	Number of steps taken by user	Screen	3

`Average Heart Rate`	Average heard rate of user	Screen	3

3. Designing the Solution

3.1 Structure Chart

First Level Decomposition



The first level decomposition includes three main goals of this program.

1. Get user's input
2. Add the gathered data to the list array of existing members.
3. Display data.

Goal Refinement

Sub-goal 1

Get data from the user.

Sub-goal 1.1

Create Member class that includes all required fields.

Sub-goal 1.2

Create Club class that includes all required methods.

Sub-goal 1.3

Create MemberManager interface to manage all the objects.

Sub-goal 1.4

Create MemberMamagerImplement class that implements MemberManager interface and gathers user's data by storing it in data array.

Sub-goal 1.5

Create Test class that has main method to run the program.

Sub-goal 2

Add data to the existing ListArray of objects.

Sub-goal 2.1

Instantiate a new Club instance.

Sub-goal 2.2

Create addMembers method in the Club class that is responsible for adding a new object to the list of members.

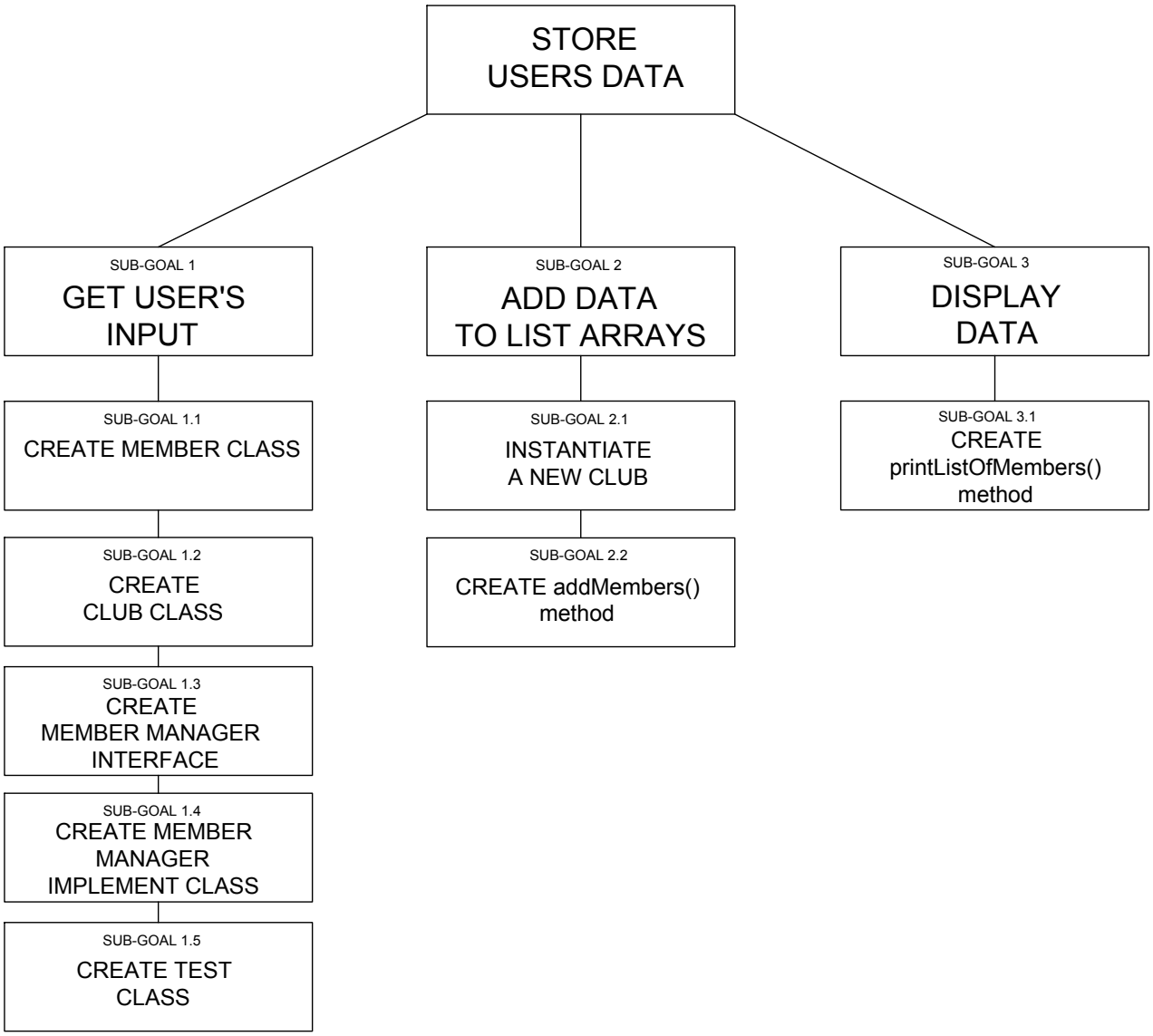
Sub-goal 3

Display results.

Sub-goal 3.1

Create the printListOfMembers() method to display all the objects in the list. In side of Test class: a new instance is being created and added, and it gets printed to the screen.

Second Level Decomposition



The second level decomposition displays more detailed process. The first sub goal consists five sub-goals: creation of four classes and one interface. The second sub-goal features instantiation a new club object and constructing method to add a new object to the existing list. The third sub-goal focuses on creating method to print all the data on the screen.

3.2 Module and Data Specifications

Name: Prompt user to enter a full name.
Input: Full name of user
Output: None
Logic: Store user's name in String name variable.

Name: Prompt user to enter data: calories consumed, distance walked, numbers of floors climbed, number of steps taken, and average heart rate.
Input: User's calories consumed, distance walked, numbers of floors climbed, number of steps taken, and average heart rate
Output: None
Logic:Store user's data in 'int [ ] data' array using for loop.

Name: Ask user if he/she wants to add a new member.
Input: Depends of users (Y/N).
Output: Depends of users (Y/N).
Logic: If user answers "yes" then the program will enter nest iteration of while loop, otherwise, the system will output the list of existing members.

3.3 Algorithm

Logic

1.0 Display user instructions
2.0 Get values from user's input.
3.0 Create an object and fill its properties with user's input
4.0 Add created object to the list of objects
5.0 Display results

Algorithm Description

Get all values from user's input and store them in ListArray of objects. Create the architecture using the singleton design pattern. This pattern allows to create only a single class which is responsible for making sure that only single object gets created. Create a new object with user's input.

Add additional object(member) if needed. Display the list of existing members.

4. Translation

4.1 Source Code

```
1 import java.util.*;
2 //=================================================
3 // Name       :  Tsagan Garyaeva
4 // SID        :  31483539
5 // Course     :  IT-114
6 // Section    :  452
7 // Instructor :  Maura Deek
8 // T.A        :
9 //=================================================
10 //=================================================
11 // Assignment # :  1
12 // Date         :  02/14/2019
13 //=================================================
14 //=================================================
15
16 public class Club {
17   // create an object
18   private static final Club INSTANCE = new Club();
19   // List to store objects
20   private List<Member> listOfMembers = new ArrayList<>();
21   // Private constructor to prevent from direct instantiation
22   private Club() {}
23
24
25
26   // Get the only object available
27   public static Club getInstance() {
28     return INSTANCE;
29   }
30   //Add new members to the list
31   public void addMembers(List<Member>newListOfmembers) {
32     listOfMembers.addAll(newListOfmembers);
33       }
34
35   // print all members
36   public String printListOfMembers() {
37     StringBuilder sb = new StringBuilder();
38     for (Member mem : listOfMembers) {
39       sb.append("\nFull Name: " + mem.getName()+
```

```
            "\nCalories Consumed: " + mem.getCal()+
            "\nDistance walked: " + mem.getDis()+
            "\nFloors climbed: " + mem.getFloors()+
            "\nSteps taken: " + mem.getSteps()+
            "\nAverage Heart Rate: " + mem.getRate()+
            "\n*******************************************" +
            "\n*******************************************"
            );
    }
    return sb.toString();
  }
}
```

```
//====================================================
// Name       :  Tsagan Garyaeva
// SID        :  31483539
// Course     :  IT-114
// Section    :  452
// Instructor :  Maura Deek
// T.A        :
//====================================================
//====================================================
// Assignment # :  1
// Date         :  02/14/2019
//====================================================
//====================================================


public class Member {
  //Fields
  private String name;
  private int cal, dis, floors, steps, rate;

  // Constructor
  public Member(String name, int cal, int dis, int floors, int steps, int rate) {
    super();
    this.name = name;
    this.cal = cal;
    this.dis = dis;
    this.floors = floors;
    this.steps = steps;
    this.rate = rate;
  }

  // Getter and setters of all the fields
  public String getName() {
```

```
34    return name;
35  }
36
37
38  public void setName(String name) {
39    this.name = name;
40  }
41
42
43  public int getCal() {
44    return cal;
45  }
46
47
48  public void setCal(int cal) {
49    this.cal = cal;
50  }
51
52
53  public int getDis() {
54    return dis;
55  }
56
57
58  public void setDis(int dis) {
59    this.dis = dis;
60  }
61
62
63  public int getFloors() {
64    return floors;
65  }
66
67
68  public void setFloors(int floors) {
69    this.floors = floors;
70  }
71
72
73  public int getSteps() {
74    return steps;
75  }
76
77
78  public void setSteps(int steps) {
79    this.steps = steps;
80  }
81
82
```

```
83    public int getRate() {
84       return rate;
85    }
86
87
88    public void setRate(int rate) {
89       this.rate = rate;
90    }
91 }
92
```

```
 1 //=================================================
 2 // Name       :  Tsagan Garyaeva
 3 // SID        :  31483539
 4 // Course     :  IT-114
 5 // Section    :  452
 6 // Instructor :  Maura Deek
 7 // T.A        :
 8 //=================================================
 9 //=================================================
10 // Assignment # :  1
11 // Date        :  02/14/2019
12 //=================================================
13 //=================================================
14
15 import java.util.List;
16 // Interface to manage all objects
17 public interface MemberManager {
18    List<Member> createListOfMembers();
19 }
```

```
=================================================
 2 // Name       :  Tsagan Garyaeva
 3 // SID        :  31483539
 4 // Course     :  IT-114
 5 // Section    :  452
 6 // Instructor :  Maura Deek
 7 // T.A        :
 8 //=================================================
 9 //=================================================
10 // Assignment # :  1
11 // Date        :  02/14/2019
```

```java
//=================================================
//=================================================


import java.util.*;

public class MemberManagerImplement implements MemberManager {

  @Override
  public List<Member> createListOfMembers() {
    // Create a new list
    List<Member> newListOfMembers = new ArrayList<>();
    // All the scanner objects to handle user's input
    Scanner scanner1 = new Scanner(System.in);
    Scanner scanner2 = new Scanner(System.in);
    Scanner input = new Scanner(System.in);
            try {
              // do while loop starts
              do {
                 System.out.println("Please enter a full name: ");
                 String name = scanner1.nextLine();

                System.out.println("Please enter the following data->");
                System.out.println("calories consumed, "
                    + "\ndistance walked, "
                    + "\nnumber of floors climbed, "
                    + "\nnumber of steps taken,"
                    + "\naverage heart rate :");
            // create an array to store data from above
            int[]data = new int[5];
            // for loop to get user's input to fill the empty
            // data array
            for(int i=0; i< data.length; i++) {
              data[i] = scanner2.nextInt();
            }
             // Instantiate a new Member object and fill it with
             // user's input
             Member item = new Member(name, data[0], data[1], data[2], data[3], data[4]);
             // Add created user to the list
             newListOfMembers.add(item);
             // ask user for the next step
             System.out.println("Would you like to add a new member?(yes/no)");
                 // If yes go to the next iteration of while loop
                 // If no, return the list
                 }while(input.nextLine().equalsIgnoreCase("y"));
            } finally {
            // Close all scanners
              scanner1.close();
              scanner2.close();
```

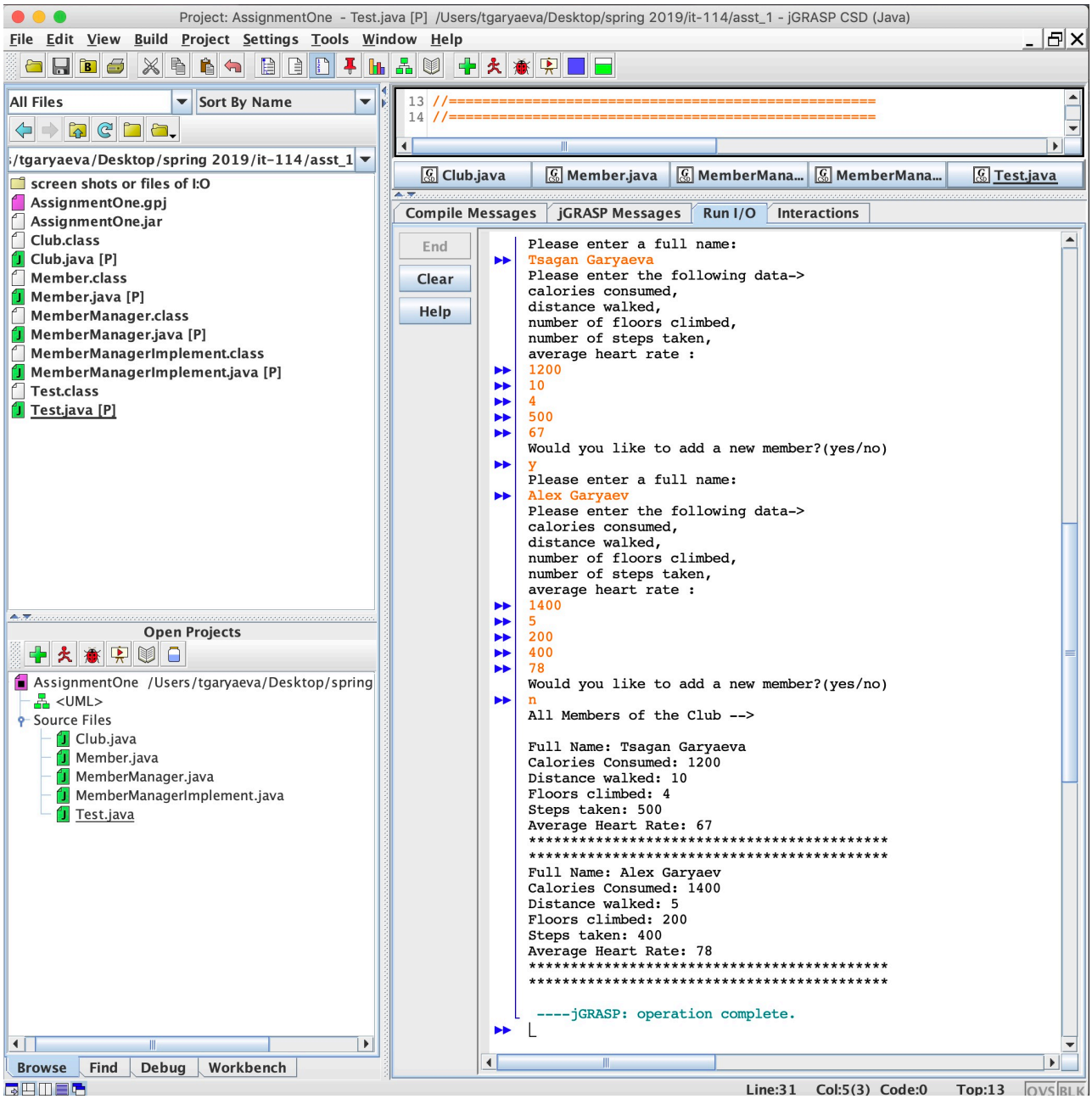
```
61                 input.close();
62
63               }
64
65           System.out.println("All Members of the Club -->");
66           // return the list
67           return newListOfMembers;
68
69    }
70 }
71
```

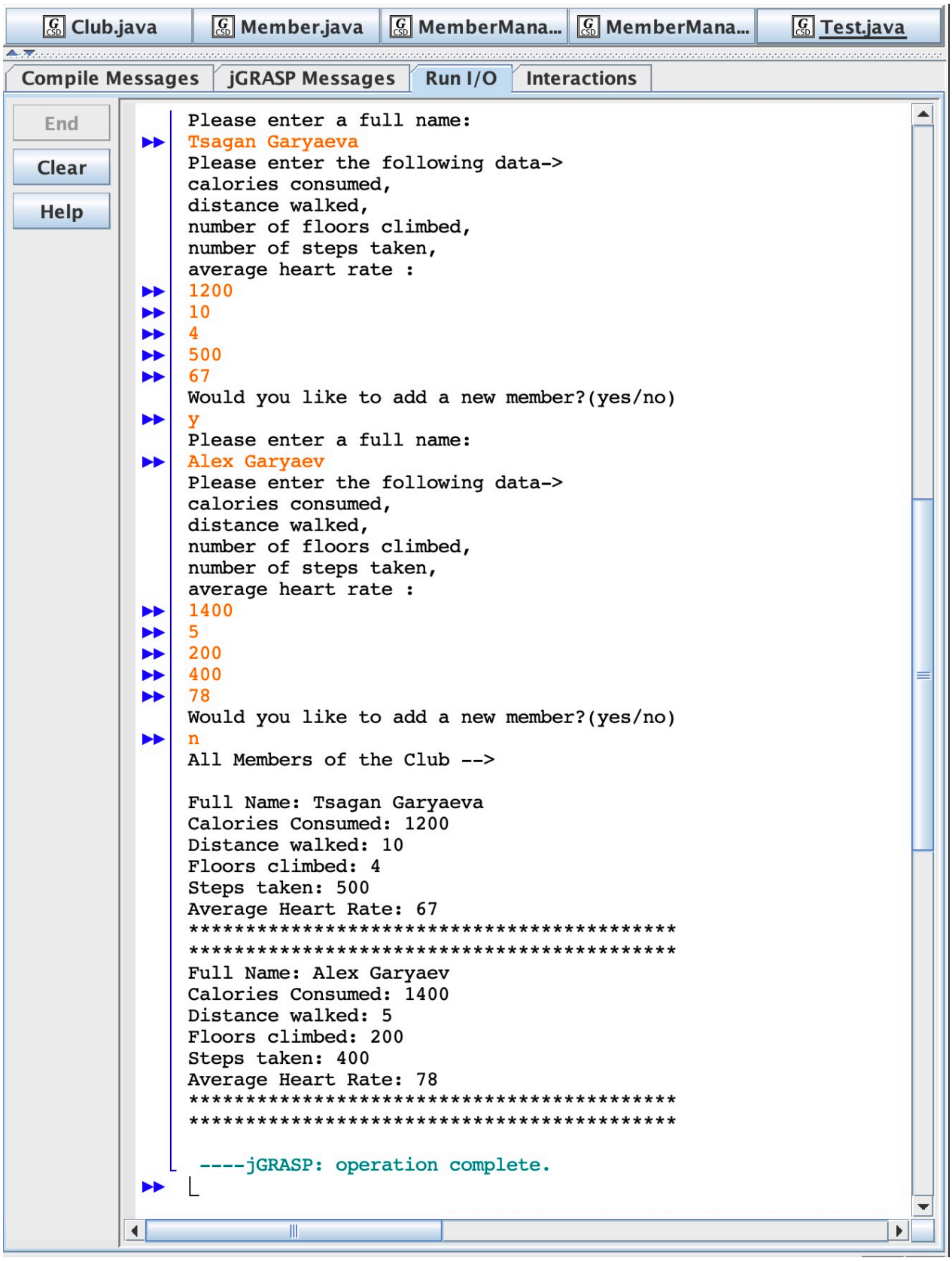
```
 1 import java.util.List;
 2 //=================================================
 3 // Name       :  Tsagan Garyaeva
 4 // SID        :  31483539
 5 // Course     :  IT-114
 6 // Section    :  452
 7 // Instructor :  Maura Deek
 8 // T.A        :
 9 //=================================================
10 //=================================================
11 // Assignment # :  1
12 // Date         :  02/14/2019
13 //=================================================
14 //=================================================
15 // Description : This program will get user's input of
16 // FitBit device and store it as object's properties
17 // using the Singleton pattern.
18 //=================================================
19 public class Test {
20 // Test class has main method to run the program
21    public static void main(String[] args) {
22       // Create a new instance
23       MemberManager itemManagerService = new MemberManagerImplement();
24         List<Member> newlistOfMembers = itemManagerService.createListOfMembers();
25          // Get the only object available and add it to the list
26         Club.getInstance().addMembers(newlistOfMembers);
27         // Print to the screen
28         System.out.println(Club.getInstance().printListOfMembers());
29     }
30
31    }
```

4. Solution Testing

Test the program with following data domain:
The domain range includes integers and String.

Test the program with following data:




Club.java
Member.java
MemberMana...
MemberMana...
Test.java
Compile Messages
jGRASP Messages
Run I/O
Interactions
End
Clear
Help
Please enter a full name:
Tsagan Garyaeva
Please enter the following data->
calories consumed,
distance walked,
number of floors climbed,
number of steps taken,
average heart rate :
1200
10
4
500
67
Would you like to add a new member?(yes/no)
y
Please enter a full name:
Alex Garyaev
Please enter the following data->
calories consumed,
distance walked,
number of floors climbed,
number of steps taken,
average heart rate :
1400
5
200
400
78
Would you like to add a new member?(yes/no)
n
All Members of the Club -->
Full Name: Tsagan Garyaeva
Calories Consumed: 1200
Distance walked: 10
Floors climbed: 4
Steps taken: 500
Average Heart Rate: 67
Full Name: Alex Garyaev
Calories Consumed: 1400
Distance walked: 5
Floors climbed: 200
Steps taken: 400
Average Heart Rate: 78
----jGRASP: operation complete.