|  |  |
| --- | --- |
| **ELECTRONIC ASSIGNMENT COVERSHEET** | Murdoch_land_RGB |

|  |  |
| --- | --- |
| Student Number | 32204274 |
| Surname | Nankivell |
| Given name | David |
| Email | [David.Nanki@gmail.com](mailto:David.Nanki@gmail.com) |
|  |  |
| Unit Code | ICT373 |
| Unit name | Software Architectures |
| Enrolment mode | Internal |
| Date | 16/04/2017 |
| Assignment number | 1 |
| Assignment name | Assignment 1 |
| Tutor | Robert Hammarstrand |

|  |
| --- |
| Student’s Declaration:   * Except where indicated, the work I am submitting in this assignment is my own work and has not been submitted for assessment in another unit. * This submission complies with Murdoch University's academic integrity commitments. I am aware that information about plagiarism and associated penalties can be found at http://www.murdoch.edu.au/teach/plagiarism/. If I have any doubts or queries about this, I am further aware that I can contact my Unit Coordinator prior to submitting the assignment. * I acknowledge that the assessor of this assignment may, for the purpose of assessing this assignment:   + reproduce this assignment and provide a copy to another academic staff member; and/or   + submit a copy of this assignment to a plagiarism-checking service. This web-based service may retain a copy of this work for the sole purpose of subsequent plagiarism checking, but has a legal agreement with the University that it will not share or reproduce it in any form. * I have retained a copy of this assignment. * I will retain a copy of the notification of receipt of this assignment. If you have not received a receipt within three days, please check with your Unit Coordinator. |
| I am aware that I am making this declaration by submitting this document electronically and by using my Murdoch ID and password it is deemed equivalent to executing this declaration with my written signature. | |
| Optional Comments to Tutor:  E.g. If this is a group assignment, list group members here | |

*If you can, please insert this completed form into the body of* ***each*** *assignment you submit. Follow the instructions in the Unit Information and Learning Guide about how to submit your file(s) and how to name them, so the Unit Coordinator knows whose work it is.*

Contents

[Title 6](#_Toc480146523)

[Requirements/Specifications 6](#_Toc480146524)

[User Guide 6](#_Toc480146525)

[Structure/Design 7](#_Toc480146526)

[Limitations 7](#_Toc480146527)

[Testing 7](#_Toc480146528)

[Advertisers 7](#_Toc480146529)

[Responders 9](#_Toc480146530)

[New Responder 11](#_Toc480146531)

[New Advertiser 14](#_Toc480146532)

[Listings 15](#_Toc480146533)

[Question2.java 15](#_Toc480146534)

[main 15](#_Toc480146535)

[PrintDetails 17](#_Toc480146536)

[InitialiseCustomers 17](#_Toc480146537)

[PrintMenu 19](#_Toc480146538)

[Login 19](#_Toc480146539)

[CheckReplies 20](#_Toc480146540)

[CheckMatches 21](#_Toc480146541)

[CreateNewUser 22](#_Toc480146542)

[AcceptPayment 23](#_Toc480146543)

[NewResponder 23](#_Toc480146544)

[NewAdvertiser 24](#_Toc480146545)

[DeleteUser 26](#_Toc480146546)

[ListAdvertisers 27](#_Toc480146547)

[ListResponders 27](#_Toc480146548)

[DatingService.java 28](#_Toc480146549)

[DatingService 28](#_Toc480146550)

[Add 28](#_Toc480146551)

[Remove 28](#_Toc480146552)

[CheckUser 29](#_Toc480146553)

[CheckPass 29](#_Toc480146554)

[CheckMatches 29](#_Toc480146555)

[Login 30](#_Toc480146556)

[ListCustomers 30](#_Toc480146557)

[ListAdvertisers 30](#_Toc480146558)

[ListResponders 31](#_Toc480146559)

[toString 31](#_Toc480146560)

[Customer.java 32](#_Toc480146561)

[Customer 32](#_Toc480146562)

[GeneratePassword 32](#_Toc480146563)

[SetGender 33](#_Toc480146564)

[SetAge 33](#_Toc480146565)

[SetIncome 33](#_Toc480146566)

[GetUsername 33](#_Toc480146567)

[GetPassword 34](#_Toc480146568)

[GetGender 34](#_Toc480146569)

[GetAge 34](#_Toc480146570)

[GetIncome 34](#_Toc480146571)

[toSafeString 34](#_Toc480146572)

[toString 35](#_Toc480146573)

[Advertiser.java 35](#_Toc480146574)

[Advertiser 36](#_Toc480146575)

[Advertise 36](#_Toc480146576)

[PartnerDescription 36](#_Toc480146577)

[SetAdvertisement 36](#_Toc480146578)

[SetPartner 37](#_Toc480146579)

[AddReply 37](#_Toc480146580)

[RemoveReply 37](#_Toc480146581)

[GetReplies 37](#_Toc480146582)

[CheckMatch 37](#_Toc480146583)

[toSafeString 38](#_Toc480146584)

[toString 38](#_Toc480146585)

[Responder.java 39](#_Toc480146586)

[Responder 39](#_Toc480146587)

[AddMatch 39](#_Toc480146588)

[RemoveMatch 39](#_Toc480146589)

[toSafeString 40](#_Toc480146590)

[toString 40](#_Toc480146591)

[PartnerSought.java 40](#_Toc480146592)

[PartnerSought 41](#_Toc480146593)

[GetGender 41](#_Toc480146594)

[GetMinAge 41](#_Toc480146595)

[GetMaxAge 41](#_Toc480146596)

[GetMinIncome 42](#_Toc480146597)

[GetMaxIncome 42](#_Toc480146598)

[SetGender 42](#_Toc480146599)

[SetMinAge 42](#_Toc480146600)

[SetMaxAge 42](#_Toc480146601)

[SetMinIncome 43](#_Toc480146602)

[SetMaxIncome 43](#_Toc480146603)

[toString 43](#_Toc480146604)

[Reply.java 44](#_Toc480146605)

[Reply 44](#_Toc480146606)

[SetReply 44](#_Toc480146607)

[SetOwner 44](#_Toc480146608)

[GetReply 45](#_Toc480146609)

[GetOwner 45](#_Toc480146610)

[toString 45](#_Toc480146611)

# Title

**Dating Service**

David Nankivell

32204274

16/04/2017

Question2.java

DatingService.java

Advertiser.java

Responder.java

Customer.java

PartnerSought.java

Reply.java

This program will manage a small database for a Dating Service, and provide functions for Customers to access, delete, and add various relevant information.

# Requirements/Specifications

The program allows the user to login as either a Responder, or an Advertiser.

When logged in as a Responder, the user will be shown a list of all Advertisers that they have matched with. Responders will then be able to send a message to any Advertiser on the list.

When logged in as an Advertiser, the user will be shown a list of all the messages that they have been sent by a Responder. The user can then choose to delete, or keep any one of these messages.

Next, the program allows the user to add a new Responder or Advertiser to the database. The user will be required to enter Customer details, and an advertisement and description of the partner they seek if they are creating an Advertiser. The user will also be required to confirm that they are willing to pay a joining fee.

Continuing, the user can delete any other user from the database, provided they know the user’s Username and Password.

The fourth option available to the user is to list all the Advertisers in the database. This also lists their passwords, presumably this would be removed from a final release!

The fifth option is similar, however it lists all the Responders instead.

Finally, the user can quit the program.

To my knowledge, all user input is validated and strictly enforced.

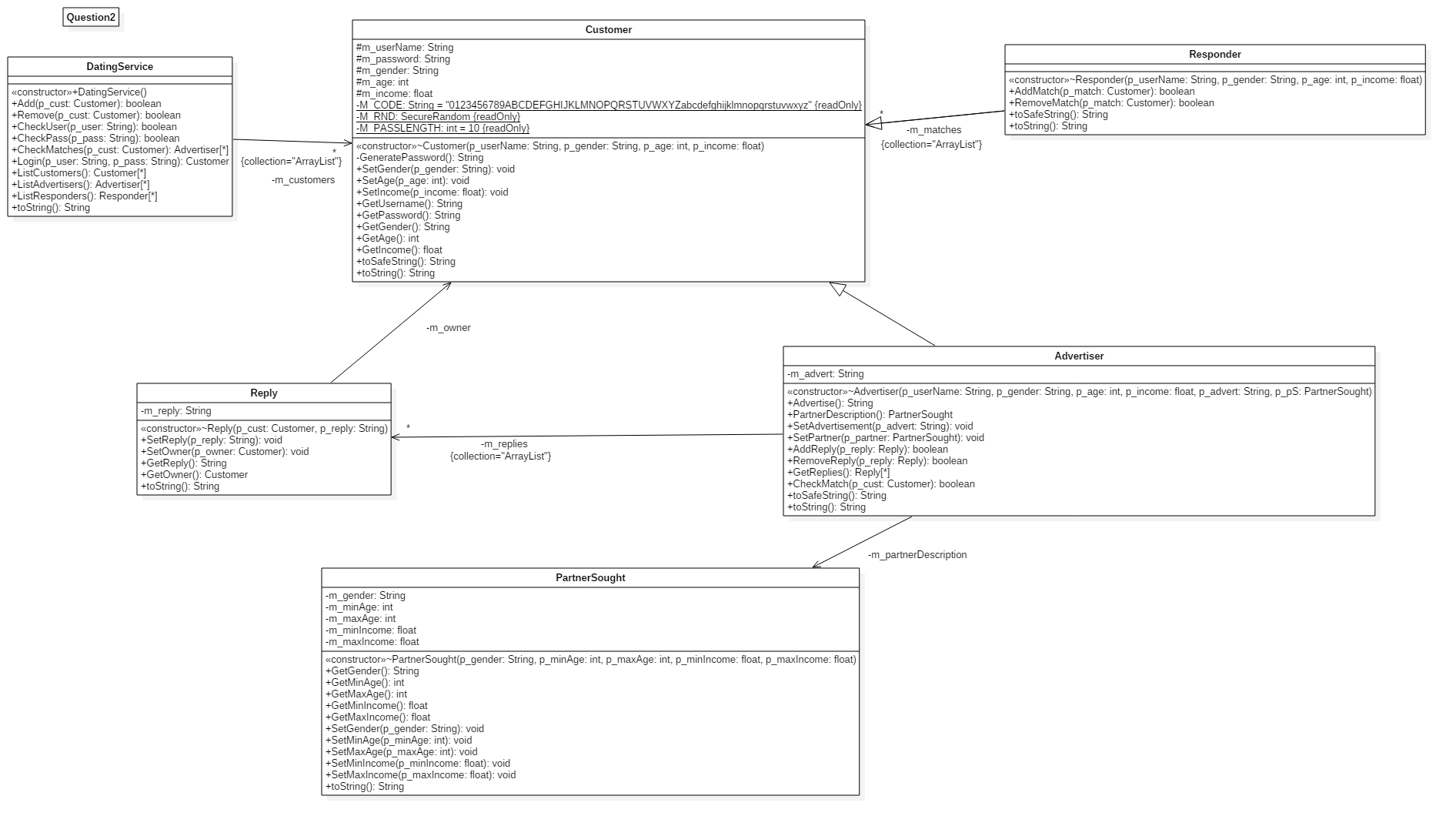
# User Guide

To compile the program, open the Netbeans project file. Then all that is required is to use the build keys at the top of the IDE. Alternately, the user can open a command prompt in the file containing the .jar file (../Question2/dist) and run the following command:

**java -jar “Question2.jar”**

The use-cases for the program are listed above, in the Requirements/Specifications.

# Structure/Design



The program is accessed through DatingService. DatingService contains an ArrayList of Customers. Customer class is extended by both Advertiser and Responder. Advertiser has a PartnerSought class, and an ArrayList of Replies. Responder has an ArrayList of Customers.

# Limitations

To the best of my knowledge, there are no limitations to the program.

# Testing

Author: David Nankivell

Student Number: 32204274

Enrolment: Internal

Tutor: Robert Hammarstrand

Tutorial: Monday 1:30 - 2:30

Welcome to the Dating Service!

Would you like to:

[1] Login

[2] Create a new user

[3] Delete an existing user

[4] List all Advertisers

[5] List all Responders

[6] Exit

4

### Advertisers

Advertisers

UserName: John; Password: JqPQRO8EZi

Gender: Male; Age: 30; Income: 50000.0

Advert: Guy looking for anyone really

Partner Description

Gender: Any; Age Range: 25-40; Income Range: $2500.0-$3500.0

UserName: Sally; Password: 8F6H7aF3wN

Gender: Female; Age: 90; Income: 4000.0

Advert: Pensioner wanting some fun

Partner Description

Gender: Female; Age Range: 30-40; Income Range: $25000.0-$35000.0

UserName: Bob; Password: Ogv5Kg1wXA

Gender: Male; Age: 21; Income: 50450.0

Advert: Dude lookin' for dudes bro

Partner Description

Gender: Male; Age Range: 55-67; Income Range: $250.0-$300.0

UserName: May; Password: wkJVCTxZ3q

Gender: Female; Age: 37; Income: 150000.0

Advert: Single mother needs some time to relax

Partner Description

Gender: Female; Age Range: 21-80; Income Range: $0.0-$3.5E8

UserName: George; Password: qZkwt5H0LA

Gender: Male; Age: 35; Income: 65000.0

Advert: Nice guy hoping for a nice girl

Partner Description

Gender: Female; Age Range: 21-120; Income Range: $2525.0-$3535.0

UserName: Sam; Password: L02R8yeWOC

Gender: Female; Age: 28; Income: 250.0

Advert: Younger woman wants to aid older people have some fun

Partner Description

Gender: Any; Age Range: 70-80; Income Range: $4000.0-$4010.0

UserName: Fred; Password: L7Rs6q9BBx

Gender: Male; Age: 45; Income: 450000.0

Advert: Single man needs some company

Partner Description

Gender: Any; Age Range: 45-46; Income Range: $2552.0-$3553.0

Enter any key to continue:

Welcome to the Dating Service!

Would you like to:

[1] Login

[2] Create a new user

[3] Delete an existing user

[4] List all Advertisers

[5] List all Responders

[6] Exit

5

### Responders

Responders

UserName: Mary; Password: Ik8BEiDbel

Gender: Female; Age: 30; Income: 3000.0

UserName: Jacob; Password: VHQWdEh2VJ

Gender: Male; Age: 40; Income: 3250.0

UserName: Eliza; Password: rUcwSQU3iy

Gender: Female; Age: 35; Income: 30000.0

UserName: Jake; Password: UwpqCB9AbM

Gender: Male; Age: 63; Income: 250.0

UserName: Jesse; Password: L4Kna416ke

Gender: Female; Age: 48; Income: 7762876.0

UserName: Lisa; Password: bYAcrnD6sF

Gender: Female; Age: 75; Income: 4005.0

UserName: Tim; Password: EIysmkJ2jZ

Gender: Male; Age: 45; Income: 3000.0

Enter any key to continue:

Welcome to the Dating Service!

Would you like to:

[1] Login

[2] Create a new user

[3] Delete an existing user

[4] List all Advertisers

[5] List all Responders

[6] Exit

1

Enter Username: Mary

Enter Password: Ik8BEiDbel

Matches

UserName: May

Gender: Female; Age: 37; Income: 150000.0

Advert: Single mother needs some time to relax

Do you wish to message this user? (Y/N): Y

Please enter message:

I would like to meet

Message added!

Continuing...

UserName: George

Gender: Male; Age: 35; Income: 65000.0

Advert: Nice guy hoping for a nice girl

Do you wish to message this user? (Y/N): Y

Please enter message:

Sure, why not?

Message added!

Continuing...

Successful Login

Welcome to the Dating Service!

Would you like to:

[1] Login

[2] Create a new user

[3] Delete an existing user

[4] List all Advertisers

[5] List all Responders

[6] Exit

1

Enter Username: George

Enter Password: qZkwt5H0LA

Replies

Owner: UserName: Mary; Password: Ik8BEiDbel

Gender: Female; Age: 30; Income: 3000.0

Reply Message: Sure, why not?

Do you wish to keep this message? (Y/N): n

Deleting message...

Message deleted

Continuing...

Successful Login

Welcome to the Dating Service!

Would you like to:

[1] Login

[2] Create a new user

[3] Delete an existing user

[4] List all Advertisers

[5] List all Responders

[6] Exit

1

Enter Username: George

Enter Password: qZkwt5H0LA

Replies

No Replies

Successful Login

Welcome to the Dating Service!

Would you like to:

[1] Login

[2] Create a new user

[3] Delete an existing user

[4] List all Advertisers

[5] List all Responders

[6] Exit

1

Enter Username: Jacob

Enter Password: VHQWdEh2VJ

Matches

No Matches

Successful Login

Welcome to the Dating Service!

Would you like to:

[1] Login

[2] Create a new user

[3] Delete an existing user

[4] List all Advertisers

[5] List all Responders

[6] Exit

1

Enter Username: May

Enter Password: wkJVCTxZ3q

Replies

Owner: UserName: Mary; Password: Ik8BEiDbel

Gender: Female; Age: 30; Income: 3000.0

Reply Message: I would like to meet

Do you wish to keep this message? (Y/N): y

Keeping message

Continuing...

Successful Login

Welcome to the Dating Service!

Would you like to:

[1] Login

[2] Create a new user

[3] Delete an existing user

[4] List all Advertisers

[5] List all Responders

[6] Exit

2

New User:

[1]Responder

[2]Advertiser

[3]Exit user creation

1

### New Responder

Will you accept the payment for user creation? Y/N

y

Thankyou for creating an account!

Please enter user name (this is final!): Freddie

Please enter gender (M/F): M

Please enter age (21-120): 68

Please enter income: 5998

New User Added

Welcome to the Dating Service!

Would you like to:

[1] Login

[2] Create a new user

[3] Delete an existing user

[4] List all Advertisers

[5] List all Responders

[6] Exit

3

Please enter login for user to be deleted: George

Enter password: qZkwt5H0LA

User Deleted

Welcome to the Dating Service!

Would you like to:

[1] Login

[2] Create a new user

[3] Delete an existing user

[4] List all Advertisers

[5] List all Responders

[6] Exit

4

Advertisers

UserName: John; Password: JqPQRO8EZi

Gender: Male; Age: 30; Income: 50000.0

Advert: Guy looking for anyone really

Partner Description

Gender: Any; Age Range: 25-40; Income Range: $2500.0-$3500.0

UserName: Sally; Password: 8F6H7aF3wN

Gender: Female; Age: 90; Income: 4000.0

Advert: Pensioner wanting some fun

Partner Description

Gender: Female; Age Range: 30-40; Income Range: $25000.0-$35000.0

UserName: Bob; Password: Ogv5Kg1wXA

Gender: Male; Age: 21; Income: 50450.0

Advert: Dude lookin' for dudes bro

Partner Description

Gender: Male; Age Range: 55-67; Income Range: $250.0-$300.0

UserName: May; Password: wkJVCTxZ3q

Gender: Female; Age: 37; Income: 150000.0

Advert: Single mother needs some time to relax

Partner Description

Gender: Female; Age Range: 21-80; Income Range: $0.0-$3.5E8

Owner: UserName: Mary; Password: Ik8BEiDbel

Gender: Female; Age: 30; Income: 3000.0

Reply Message: I would like to meet

UserName: Sam; Password: L02R8yeWOC

Gender: Female; Age: 28; Income: 250.0

Advert: Younger woman wants to aid older people have some fun

Partner Description

Gender: Any; Age Range: 70-80; Income Range: $4000.0-$4010.0

UserName: Fred; Password: L7Rs6q9BBx

Gender: Male; Age: 45; Income: 450000.0

Advert: Single man needs some company

Partner Description

Gender: Any; Age Range: 45-46; Income Range: $2552.0-$3553.0

Enter any key to continue:

Welcome to the Dating Service!

Would you like to:

[1] Login

[2] Create a new user

[3] Delete an existing user

[4] List all Advertisers

[5] List all Responders

[6] Exit

5

Responders

UserName: Mary; Password: Ik8BEiDbel

Gender: Female; Age: 30; Income: 3000.0

UserName: Jacob; Password: VHQWdEh2VJ

Gender: Male; Age: 40; Income: 3250.0

UserName: Eliza; Password: rUcwSQU3iy

Gender: Female; Age: 35; Income: 30000.0

UserName: Jake; Password: UwpqCB9AbM

Gender: Male; Age: 63; Income: 250.0

UserName: Jesse; Password: L4Kna416ke

Gender: Female; Age: 48; Income: 7762876.0

UserName: Lisa; Password: bYAcrnD6sF

Gender: Female; Age: 75; Income: 4005.0

UserName: Tim; Password: EIysmkJ2jZ

Gender: Male; Age: 45; Income: 3000.0

UserName: Freddie; Password: Y6ttcHmCzc

Gender: Male; Age: 68; Income: 5998.0

Enter any key to continue:

Welcome to the Dating Service!

Would you like to:

[1] Login

[2] Create a new user

[3] Delete an existing user

[4] List all Advertisers

[5] List all Responders

[6] Exit

2

New User:

[1]Responder

[2]Advertiser

[3]Exit user creation

2

Will you accept the payment for user creation? Y/N

n

Returning to main menu...

No User Added

Welcome to the Dating Service!

Would you like to:

[1] Login

[2] Create a new user

[3] Delete an existing user

[4] List all Advertisers

[5] List all Responders

[6] Exit

2

### New Advertiser

New User:

[1]Responder

[2]Advertiser

[3]Exit user creation

2

Will you accept the payment for user creation? Y/N

y

Thankyou for creating an account!

Please enter user name (this is final!): Joe

Please enter gender (M/F): F

Please enter age (21-120): 26

Please enter income: 4000

Please enter your description:

Hey, I'm Joe Bloggs, looking for a good time!

Please enter preferred partner gender (M/F/A): A

Please enter minimum preferred age (21-120): 30

Please enter maximum preferred age (21-120): 50

Please enter minimum preferred income: 0

Please enter maximum preferred income: 5000

New User Added

Welcome to the Dating Service!

Would you like to:

[1] Login

[2] Create a new user

[3] Delete an existing user

[4] List all Advertisers

[5] List all Responders

[6] Exit

6

Exiting Dating Service...

BUILD SUCCESSFUL (total time: 6 minutes 28 seconds)

# Listings

## Question2.java

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package question2;

import java.util.Scanner;

import java.util.ArrayList;

/\*\*

\* Main test program for Question 2

\* @author David Nankivell

\* 16/04/2017

\* Question2.java

\* Test Program for question2 package

\*/

public class Question2 {

### main

/\*\*

\* Main menu of program

\* @param args the command line arguments

\*/

public static void main(String[] args) {

PrintDetails();

DatingService o\_dServ = new DatingService();

InitialiseCustomers(o\_dServ);

int o\_menuChoice;

Scanner o\_input = new Scanner(System.in);

boolean o\_menu = true;

while (o\_menu)

{

PrintMenu();

if (o\_input.hasNextInt())

{

while (!o\_input.hasNext("[1-6]"))

{

System.out.println("Choice must be between 1 and 6");

PrintMenu();

o\_input.nextInt();

}

o\_menuChoice = o\_input.nextInt();

switch (o\_menuChoice)

{

case 1: //Login

if (Login(o\_dServ))

{

System.out.println("Successful Login");

}

else

{

System.out.println("Unsuccessful Login");

}

break;

case 2: //New User

if (CreateNewUser(o\_dServ))

{

System.out.println("New User Added");

}

else

{

System.out.println("No User Added");

}

break;

case 3: //Delete User

if (DeleteUser(o\_dServ))

{

System.out.println("User Deleted");

}

else

{

System.out.println("No User Deletion");

}

break;

case 4: //List Advertisers

ListAdvertisers(o\_dServ);

break;

case 5: //List Responders

ListResponders(o\_dServ);

break;

case 6: //Exit

System.out.println("Exiting Dating Service...");

o\_menu = false;

break;

}

}

else

{

System.out.println("Please enter a number");

o\_input.next();

}

}

}

### PrintDetails

/\*\*

\* Print student details

\*/

private static void PrintDetails()

{

System.out.print("Author: David Nankivell\n"

+ "Student Number: 32204274\n"

+ "Enrolment: Internal\n"

+ "Tutor: Robert Hammarstrand\n"

+ "Tutorial: Monday 1:30 - 2:30\n\n");

}

### InitialiseCustomers

/\*\*

\* Set initial Customers into DatingService

\* @param p\_dServ DatingService

\*/

private static void InitialiseCustomers(DatingService p\_dServ)

{

ArrayList<PartnerSought> l\_paSo = new ArrayList<>();

ArrayList<String> l\_advert = new ArrayList<>();

ArrayList<Advertiser> l\_advT = new ArrayList<>();

ArrayList<Responder> l\_resp = new ArrayList<>();

//Partner Sought -- Gender, MinAge, MaxAge, MinIncome, MaxIncome

l\_paSo.add(new PartnerSought("A", 25, 40, 2500, 3500)); //0

l\_paSo.add(new PartnerSought("F", 30, 40, 25000, 35000)); //1

l\_paSo.add(new PartnerSought("M", 55, 67, 250, 300)); //2

l\_paSo.add(new PartnerSought("F", 21, 80, 0, 350000000)); //3

l\_paSo.add(new PartnerSought("F", 21, 120, 2525, 3535)); //4

l\_paSo.add(new PartnerSought("A", 70, 80, 4000, 4010)); //5

l\_paSo.add(new PartnerSought("A", 45, 46, 2552, 3553)); //6

//Advert -- String describing Advertiser

l\_advert.add("Guy looking for anyone really"); //0

l\_advert.add("Pensioner wanting some fun"); //1

l\_advert.add("Dude lookin' for dudes bro"); //2

l\_advert.add("Single mother needs some time to relax"); //3

l\_advert.add("Nice guy hoping for a nice girl"); //4

l\_advert.add("Younger woman wants to aid older people have some fun"); //5

l\_advert.add("Single man needs some company"); //6

//Advertiser -- Name, Gender, Age, Income, Advert, Partner Sought

l\_advT.add(new Advertiser("John", "M", 30, 50000, l\_advert.get(0), l\_paSo.get(0)));

l\_advT.add(new Advertiser("Sally", "F", 90, 4000, l\_advert.get(1), l\_paSo.get(1)));

l\_advT.add(new Advertiser("Bob", "M", 21, 50450, l\_advert.get(2), l\_paSo.get(2)));

l\_advT.add(new Advertiser("May", "F", 37, 150000, l\_advert.get(3), l\_paSo.get(3)));

l\_advT.add(new Advertiser("George", "M", 35, 65000, l\_advert.get(4), l\_paSo.get(4)));

l\_advT.add(new Advertiser("Sam", "F", 28, 250, l\_advert.get(5), l\_paSo.get(5)));

l\_advT.add(new Advertiser("Fred", "M", 45, 450000, l\_advert.get(6), l\_paSo.get(6)));

//Responder -- Name, Gender, Age, Income

l\_resp.add(new Responder("Mary", "F", 30, 3000)); //Partner Sought 0

l\_resp.add(new Responder("Jacob", "M", 40, 3250)); //Partner Sought 0

l\_resp.add(new Responder("Eliza", "F", 35, 30000)); //Partner Sought 1

l\_resp.add(new Responder("Jake", "M", 63, 250)); //Partner Sought 2

l\_resp.add(new Responder("Jesse", "F", 48, 7762876)); //Partner Sought 3

l\_resp.add(new Responder("Lisa", "F", 75, 4005)); //Partner Sought 5

l\_resp.add(new Responder("Tim", "M", 45, 3000)); //Partner Sought 6

for (Advertiser i\_advT : l\_advT)

{

p\_dServ.Add(i\_advT);

}

for (Responder i\_resp : l\_resp)

{

p\_dServ.Add(i\_resp);

}

}

### PrintMenu

/\*\*

\* Print main menu

\*/

private static void PrintMenu()

{

System.out.print("Welcome to the Dating Service!\n"

+ " Would you like to: \n"

+ "[1] Login\n"

+ "[2] Create a new user\n"

+ "[3] Delete an existing user\n"

+ "[4] List all Advertisers\n"

+ "[5] List all Responders\n"

+ "[6] Exit\n");

}

### Login

/\*\*

\* Log user into DatingService

\* @param p\_dServ DatingService

\* @return Boolean regarding status of user login

\*/

private static boolean Login(DatingService p\_dServ)

{

Scanner input = new Scanner(System.in);

System.out.print("Enter Username: ");

String l\_user = input.nextLine();

if (p\_dServ.CheckUser(l\_user))

{

System.out.print("Enter Password: ");

String l\_pass = input.nextLine();

if (p\_dServ.CheckPass(l\_pass))

{

Customer l\_cust = p\_dServ.Login(l\_user, l\_pass);

if (l\_cust instanceof Advertiser)

{

CheckReplies(p\_dServ, l\_cust);

return true;

}

else if (l\_cust instanceof Responder)

{

CheckMatches(p\_dServ, l\_cust);

return true;

}

}

}

return false;

}

### CheckReplies

/\*\*

\* Check for Advertiser's replies

\* @param p\_dServ DatingService

\* @param p\_cust Customer (Advertiser)

\*/

private static void CheckReplies(DatingService p\_dServ, Customer p\_cust)

{

Scanner input = new Scanner(System.in);

Advertiser l\_advT = (Advertiser)p\_cust;

ArrayList<Reply> l\_replies = l\_advT.GetReplies();

ArrayList<Reply> l\_delReps = new ArrayList<>();

System.out.println("Replies");

if (!l\_replies.isEmpty())

{

for (Reply i\_reply : l\_replies)

{

System.out.print(i\_reply.toString());

System.out.print("Do you wish to keep this message? (Y/N): ");

if (!input.hasNext("[ynYN]"))

{

System.out.print("Please enter (Y/N): ");

}

String l\_repChoice = input.next();

input.nextLine();

switch (l\_repChoice)

{

case "y":

case "Y":

System.out.println("Keeping message");

break;

case "n":

case "N":

System.out.println("Deleting message...");

l\_delReps.add(i\_reply);

System.out.println("Message deleted");

break;

}

System.out.println("Continuing...");

}

for (Reply i\_reply : l\_delReps)

{

l\_advT.RemoveReply(i\_reply);

}

}

else

{

System.out.println("No Replies");

}

}

### CheckMatches

/\*\*

\* Check for Responder's matches

\* @param p\_dServ DatingService

\* @param p\_cust Customer (Responder)

\*/

private static void CheckMatches(DatingService p\_dServ, Customer p\_cust)

{

Scanner input = new Scanner(System.in);

ArrayList<Advertiser> l\_advT = p\_dServ.CheckMatches(p\_cust);

System.out.println("Matches");

if (!l\_advT.isEmpty())

{

for (Advertiser i\_advT : l\_advT)

{

System.out.print(i\_advT.toSafeString());

System.out.print("Do you wish to message this user? (Y/N): ");

if (!input.hasNext("[ynYN]"))

{

System.out.print("Please enter (Y/N): ");

}

String l\_mesChoice = input.next();

input.nextLine();

switch (l\_mesChoice)

{

case "y":

case "Y":

System.out.println("Please enter message: ");

String l\_message = input.nextLine();

Reply l\_msg = new Reply(p\_cust, l\_message);

i\_advT.AddReply(l\_msg);

System.out.println("Message added!");

break;

case "n":

case "N":

System.out.println("No message will be sent");

break;

}

System.out.println("Continuing...");

}

}

else

{

System.out.println("No Matches");

}

}

### CreateNewUser

/\*\*

\* Add a new Customer to the DatingService

\* @param p\_dServ DatingService

\* @return Boolean regarding success of creating a new Customer

\*/

private static boolean CreateNewUser(DatingService p\_dServ)

{

int l\_menuChoice;

Scanner input = new Scanner(System.in);

while (true)

{

System.out.print("New User: \n[1]Responder\n[2]Advertiser\n[3]Exit user creation\n");

if (input.hasNextInt())

{

l\_menuChoice = input.nextInt();

switch (l\_menuChoice)

{

case 1:

if (AcceptPayment())

{

return NewResponder(p\_dServ);

}else{

return false;

}

case 2:

if (AcceptPayment())

{

return NewAdvertiser(p\_dServ);

}else{

return false;

}

case 3:

System.out.println("Returning to main menu...");

return false;

default:

System.out.println("Please enter 1, 2 or 3!");

break;

}

}

System.out.println("Please enter 1, 2 or 3!");

input.next();

}

}

### AcceptPayment

/\*\*

\* Check to see if new Customer will accept a payment

\* @return Boolean regarding acceptance of payment

\*/

private static boolean AcceptPayment()

{

String l\_payChoice;

Scanner input = new Scanner(System.in);

while (true)

{

System.out.println("Will you accept the payment for user creation? Y/N");

if (input.hasNext())

{

l\_payChoice = input.next();

}else

{

return false;

}

switch (l\_payChoice)

{

case "y":

case "Y":

System.out.println("Thankyou for creating an account!");

return true;

case "n":

case "N":

System.out.println("Returning to main menu...");

return false;

default:

System.out.println("Please enter Y or N");

break;

}

}

}

### NewResponder

/\*\*

\* Add a new Responder to the DatingService

\* @param p\_dServ DatingService

\* @return Boolean regarding success of adding new Responder

\*/

private static boolean NewResponder(DatingService p\_dServ)

{

Scanner input = new Scanner(System.in);

String l\_userName, l\_gender;

int l\_age;

float l\_income;

System.out.print("Please enter user name (this is final!): ");

l\_userName = input.next();

System.out.print("Please enter gender (M/F): ");

while (!input.hasNext("[MF]"))

{

System.out.println("Must be M or F!");

System.out.print("Please enter gender (M/F): ");

input.next();

}

l\_gender = input.next();

System.out.print("Please enter age (21-120): ");

while (!input.hasNext("2[1-9]|3[0-9]|4[0-9]|5[0-9]|6[0-9]|7[0-9]|8[0-9]|9[0-9]|10[0-9]|11[0-9]|120"))

{

System.out.println("Age must be between 21 and 120!");

System.out.print("Please enter age (21-120): ");

input.nextInt();

}

l\_age = input.nextInt();

System.out.print("Please enter income: ");

while (!input.hasNextFloat())

{

System.out.println("Must be a number!");

System.out.print("Please enter income: ");

input.nextFloat();

}

l\_income = input.nextFloat();

Responder l\_resp = new Responder(l\_userName, l\_gender, l\_age, l\_income);

return p\_dServ.Add(l\_resp);

}

### NewAdvertiser

/\*\*

\* Add a new Advertiser to the DatingService

\* @param p\_dServ DatingService

\* @return Boolean regarding success of adding new Advertiser

\*/

private static boolean NewAdvertiser(DatingService p\_dServ)

{

Scanner input = new Scanner(System.in);

String l\_userName, l\_gender, l\_advert, l\_prefGen;

int l\_age, l\_minA, l\_maxA;

float l\_income, l\_minIn, l\_maxIn;

System.out.print("Please enter user name (this is final!): ");

l\_userName = input.next();

System.out.print("Please enter gender (M/F): ");

while (!input.hasNext("[MF]"))

{

System.out.println("Must be M or F!");

System.out.print("Please enter gender (M/F): ");

input.next();

}

l\_gender = input.next();

System.out.print("Please enter age (21-120): ");

while (!input.hasNext("2[1-9]|3[0-9]|4[0-9]|5[0-9]|6[0-9]|7[0-9]|8[0-9]|9[0-9]|10[0-9]|11[0-9]|120"))

{

System.out.println("Age must be between 21 and 120!");

System.out.print("Please enter age (21-120): ");

input.nextInt();

}

l\_age = input.nextInt();

System.out.print("Please enter income: ");

while (!input.hasNextFloat())

{

System.out.println("Must be a number!");

System.out.print("Please enter income: ");

input.nextFloat();

}

l\_income = input.nextFloat();

input.nextLine();

System.out.println("Please enter your description:");

l\_advert = input.nextLine();

System.out.print("Please enter preferred partner gender (M/F/A): ");

while (!input.hasNext("[MFA]"))

{

System.out.println("Must be M or F or A!");

System.out.print("Please enter preferred partner gender (M/F/A): ");

input.next();

}

l\_prefGen = input.next();

System.out.print("Please enter minimum preferred age (21-120): ");

while (!input.hasNext("2[1-9]|3[0-9]|4[0-9]|5[0-9]|6[0-9]|7[0-9]|8[0-9]|9[0-9]|10[0-9]|11[0-9]|120"))

{

System.out.println("Minimum Age must be between 21 and 120!");

System.out.print("Please enter minimum preferred age (21-120): ");

input.nextInt();

}

l\_minA = input.nextInt();

System.out.print("Please enter maximum preferred age (21-120): ");

while (!input.hasNext("2[1-9]|3[0-9]|4[0-9]|5[0-9]|6[0-9]|7[0-9]|8[0-9]|9[0-9]|10[0-9]|11[0-9]|120"))

{

System.out.println("Maximum Age must be between 21 and 120!");

System.out.print("Please enter maximum preferred age (21-120): ");

input.nextInt();

}

l\_maxA = input.nextInt();

System.out.print("Please enter minimum preferred income: ");

while (!input.hasNextFloat())

{

System.out.println("Must be a number!");

System.out.print("Please enter minimum income: ");

input.nextFloat();

}

l\_minIn = input.nextFloat();

System.out.print("Please enter maximum preferred income: ");

while (!input.hasNextFloat())

{

System.out.println("Must be a number!");

System.out.print("Please enter maximum income: ");

input.nextFloat();

}

l\_maxIn = input.nextFloat();

PartnerSought l\_pS = new PartnerSought(l\_prefGen, l\_minA, l\_maxA, l\_minIn, l\_maxIn);

Advertiser l\_adv = new Advertiser(l\_userName, l\_gender, l\_age, l\_income, l\_advert, l\_pS);

return p\_dServ.Add(l\_adv);

}

### DeleteUser

/\*\*

\* Method to delete a Customer from the DatingService

\* @param p\_dServ DatingService

\* @return Boolean regarding the success of removing the Customer

\*/

private static boolean DeleteUser(DatingService p\_dServ)

{

ArrayList<Customer> l\_cust = p\_dServ.ListCustomers();

Scanner input = new Scanner(System.in);

System.out.print("Please enter login for user to be deleted: ");

String l\_login = input.nextLine();

for (Customer i\_cust : l\_cust)

{

if (l\_login.equals(i\_cust.GetUsername()))

{

System.out.print("Enter password: ");

String l\_password = input.nextLine();

if (l\_password.equals(i\_cust.GetPassword()))

{

return p\_dServ.Remove(i\_cust);

}

else

{

System.out.println("Incorrect Password, returning to menu...");

}

}

}

System.out.println("No Matching User, returning to menu...");

return false;

}

### ListAdvertisers

/\*\*

\* Method to list all Advertisers in the DatingService

\* @param p\_dServ DatingService

\*/

private static void ListAdvertisers(DatingService p\_dServ)

{

Scanner input = new Scanner(System.in);

ArrayList<Advertiser> l\_advT = p\_dServ.ListAdvertisers();

String l\_string = "Advertisers\n";

for (Advertiser i\_advT : l\_advT)

{

l\_string += i\_advT.toString();

}

System.out.print(l\_string);

System.out.print("Enter any key to continue: ");

input.nextLine();

System.out.print("\n");

}

### ListResponders

/\*\*

\* Method to list all Responders in the DatingService

\* @param p\_dServ DatingService

\*/

private static void ListResponders(DatingService p\_dServ)

{

Scanner input = new Scanner(System.in);

ArrayList<Responder> l\_resp = p\_dServ.ListResponders();

String l\_string = "Responders\n";

for (Responder i\_resp : l\_resp)

{

l\_string += i\_resp.toString() + "\n\n";

}

System.out.print(l\_string);

System.out.print("Enter any key to continue: ");

input.nextLine();

System.out.print("\n");

}

}

## DatingService.java

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package question2;

import java.util.ArrayList;

/\*\*

\* Dating Service to contain all Customers

\* @author David Nankivell

\* 16/04/2017

\* DatingService.java

\* This class keeps a record of all Customers

\*/

public class DatingService {

private ArrayList<Customer> m\_customers;

### DatingService

/\*\*

\* Constructor

\*/

public DatingService()

{

m\_customers = new ArrayList<>();

}

### Add

/\*\*

\* Add a customer to the DatingService

\* @param p\_cust Customer to be added

\* @return Boolean regarding success of add operation

\*/

public boolean Add(Customer p\_cust)

{

return m\_customers.add(p\_cust);

}

### Remove

/\*\*

\* Remove a customer from the DatingService

\* @param p\_cust Customer to be removed

\* @return Boolean regarding success of remove operation

\*/

public boolean Remove(Customer p\_cust)

{

return m\_customers.remove(p\_cust);

}

### CheckUser

/\*\*

\* Check if given username exists inside DatingService

\* @param p\_user Username to be checked

\* @return Boolean regarding existence of username

\*/

public boolean CheckUser(String p\_user)

{

for (Customer i\_cust : m\_customers)

{

if (p\_user.equals(i\_cust.GetUsername()))

{

return true;

}

}

return false;

}

### CheckPass

/\*\*

\* Check if given password exists inside DatingService

\* @param p\_pass Password to be checked

\* @return Boolean regarding existence of password

\*/

public boolean CheckPass(String p\_pass)

{

for (Customer i\_cust : m\_customers)

{

if (p\_pass.equals(i\_cust.GetPassword()))

{

return true;

}

}

return false;

}

### CheckMatches

/\*\*

\* Returns list of all Advertisers matching Customer

\* @param p\_cust Customer to be matched

\* @return ArrayList of Advertisers that match Customer

\*/

public ArrayList<Advertiser> CheckMatches (Customer p\_cust)

{

ArrayList<Advertiser> al\_advT = new ArrayList<>();

for (Customer i\_cust : m\_customers)

{

if (i\_cust instanceof Advertiser)

{

Advertiser l\_advT = (Advertiser)i\_cust;

if (l\_advT.CheckMatch(p\_cust))

{

al\_advT.add(l\_advT);

}

}

}

return al\_advT;

}

### Login

/\*\*

\* Checks if given username matches given password

\* @param p\_user Username to be validated

\* @param p\_pass Password to be validated

\* @return Customer matching given username and password

\*/

public Customer Login(String p\_user, String p\_pass)

{

for (Customer i\_cust : m\_customers)

{

if ((p\_user.equals(i\_cust.GetUsername())) && (p\_pass.equals(i\_cust.GetPassword())))

{

return i\_cust;

}

}

return null;

}

### ListCustomers

/\*\*

\* Returns all Customers in DatingService

\* @return ArrayList of all Customers

\*/

public ArrayList<Customer> ListCustomers()

{

return m\_customers;

}

### ListAdvertisers

/\*\*

\* Returns all Advertisers in the DatingService

\* @return ArrayList of all Advertisers

\*/

public ArrayList<Advertiser> ListAdvertisers()

{

ArrayList<Advertiser> l\_advT = new ArrayList<>();

for (Customer i\_cust : m\_customers)

{

if (i\_cust instanceof Advertiser)

{

l\_advT.add((Advertiser)i\_cust);

}

}

return l\_advT;

}

### ListResponders

/\*\*

\* Returns all Responders in the DatingService

\* @return ArrayList of all Responders

\*/

public ArrayList<Responder> ListResponders()

{

ArrayList<Responder> l\_resp = new ArrayList<>();

for (Customer i\_cust : m\_customers)

{

if (i\_cust instanceof Responder)

{

l\_resp.add((Responder)i\_cust);

}

}

return l\_resp;

}

### toString

/\*\*

\* Puts all DatingService data into a string

\* @return String with all DatingService details

\*/

@Override

public String toString()

{

String l\_string = "Dating Service \n";

for (Customer i\_cust : m\_customers)

{

l\_string += i\_cust.toString();

}

return l\_string;

}

}

## Customer.java

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package question2;

import java.security.SecureRandom;

/\*\*

\* Customer represents each user of the Dating Service

\* @author David Nankivell

\* 16/04/2017

\* Customer.java

\* Class Customer to be extended by Responder and Advertiser

\*/

public class Customer {

protected String m\_userName;

protected String m\_password;

protected String m\_gender;

protected int m\_age;

protected float m\_income;

private static final String M\_CODE = "0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz";

private static final SecureRandom M\_RND = new SecureRandom();

private static final int M\_PASSLENGTH = 10;

### Customer

/\*\*

\* Constructor

\* @param p\_userName Username of Customer

\* @param p\_gender Gender of Customer

\* @param p\_age Age of Customer

\* @param p\_income Income of Customer

\*/

Customer (String p\_userName, String p\_gender,

int p\_age, float p\_income)

{

m\_userName = p\_userName;

m\_password = GeneratePassword();

m\_gender = p\_gender;

m\_age = p\_age;

m\_income = p\_income;

}

### GeneratePassword

/\*\*

\* Private method to generate passwords

\* @return String password

\*/

private String GeneratePassword()

{

StringBuilder l\_pass = new StringBuilder(M\_PASSLENGTH);

for (int i = 0; i < M\_PASSLENGTH; i++ )

{

l\_pass.append( M\_CODE.charAt( M\_RND.nextInt( M\_CODE.length())));

}

return l\_pass.toString();

}

### SetGender

/\*\*

\* Sets gender of Customer

\* @param p\_gender Customer gender

\*/

public void SetGender(String p\_gender)

{

m\_gender = p\_gender;

}

### SetAge

/\*\*

\* Sets age of Customer

\* @param p\_age Customer age

\*/

public void SetAge(int p\_age)

{

m\_age = p\_age;

}

### SetIncome

/\*\*

\* Sets income of Customer

\* @param p\_income Customer income

\*/

public void SetIncome(float p\_income)

{

m\_income = p\_income;

}

### GetUsername

/\*\*

\* Gets Customer username

\* @return String username

\*/

public String GetUsername ()

{

return m\_userName;

}

### GetPassword

/\*\*

\* Gets Customer password

\* @return String password

\*/

public String GetPassword()

{

return m\_password;

}

### GetGender

/\*\*

\* Gets Customer gender

\* @return String gender

\*/

public String GetGender()

{

return m\_gender;

}

### GetAge

/\*\*

\* Gets Customer age

\* @return Integer age

\*/

public int GetAge()

{

return m\_age;

}

### GetIncome

/\*\*

\* Gets Customer income

\* @return Float income

\*/

public float GetIncome()

{

return m\_income;

}

### toSafeString

/\*\*

\* Safe Customer -- no passwords

\* @return Safe Customer String

\*/

public String toSafeString()

{

String l\_toString = "UserName: " + m\_userName + "\nGender: ";

if (m\_gender.equals("M"))

{

l\_toString += "Male";

}

else if (m\_gender.equals("F"))

{

l\_toString += "Female";

}

l\_toString += "; Age: " + m\_age + "; Income: " + m\_income;

return l\_toString;

}

### toString

/\*\*

\* Customer details to string

\* @return String Customer

\*/

@Override

public String toString()

{

String l\_toString = "UserName: " + m\_userName + "; Password: " + m\_password + "\nGender: ";

if (m\_gender.equals("M"))

{

l\_toString += "Male";

}

else if (m\_gender.equals("F"))

{

l\_toString += "Female";

}

l\_toString += "; Age: " + m\_age + "; Income: " + m\_income;

return l\_toString;

}

}

## Advertiser.java

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package question2;

import java.util.ArrayList;

/\*\*

\* Extends Customer, Advertiser asks for Replies

\* @author David Nankivell

\* 16/04/2017

\* Advertiser.java

\* Advertiser extends Customer, Advertises for sought match

\*/

public class Advertiser extends Customer {

private String m\_advert;

private PartnerSought m\_partnerDescription;

private ArrayList<Reply> m\_replies = new ArrayList<>();

### Advertiser

/\*\*

\* Constructor

\* @param p\_userName Username of Advertiser

\* @param p\_gender Gender of Advertiser

\* @param p\_age Age of Advertiser

\* @param p\_income Income of Advertiser

\* @param p\_advert Advertiser's advertisement

\* @param p\_pS PartnerSought by Advertiser

\*/

Advertiser (String p\_userName, String p\_gender, int p\_age,

float p\_income, String p\_advert, PartnerSought p\_pS)

{

super(p\_userName, p\_gender, p\_age, p\_income);

m\_advert = p\_advert;

m\_partnerDescription = p\_pS;

}

### Advertise

/\*\*

\* Advertises Advertiser

\* @return String with Advertiser's advertisement

\*/

public String Advertise()

{

return m\_advert;

}

### PartnerDescription

/\*\*

\* Advertiser's PartnerSought

\* @return PartnerSought by Advertiser

\*/

public PartnerSought PartnerDescription()

{

return m\_partnerDescription;

}

### SetAdvertisement

/\*\*

\* Set Advertiser's advertisement

\* @param p\_advert Advertisement

\*/

public void SetAdvertisement(String p\_advert)

{

m\_advert = p\_advert;

}

### SetPartner

/\*\*

\* Set PartnerSought by Advertiser

\* @param p\_partner PartnerSought

\*/

public void SetPartner(PartnerSought p\_partner)

{

m\_partnerDescription = p\_partner;

}

### AddReply

/\*\*

\* Add a new Reply to the Advertiser

\* @param p\_reply Reply message

\* @return Boolean regarding success of add operation

\*/

public boolean AddReply(Reply p\_reply)

{

return m\_replies.add(p\_reply);

}

### RemoveReply

/\*\*

\* Remove a Reply from the Advertiser

\* @param p\_reply Reply message

\* @return Boolean regarding success of remove operation

\*/

public boolean RemoveReply(Reply p\_reply)

{

return m\_replies.remove(p\_reply);

}

### GetReplies

/\*\*

\* Method to return list of all Replies

\* @return ArrayList of Replies

\*/

public ArrayList<Reply> GetReplies()

{

return m\_replies;

}

### CheckMatch

/\*\*

\* Check that passed customer matches PartnerSought parameters

\* @param p\_cust Customer to be matched

\* @return Boolean regarding match success

\*/

public boolean CheckMatch(Customer p\_cust)

{

if ((p\_cust.GetAge() >= m\_partnerDescription.GetMinAge()) &&

(p\_cust.GetAge() <= m\_partnerDescription.GetMaxAge()))

{

if ((p\_cust.GetIncome() >= m\_partnerDescription.GetMinIncome()) &&

(p\_cust.GetIncome() <= m\_partnerDescription.GetMaxIncome()))

{

if ((p\_cust.GetGender().equals(m\_partnerDescription.GetGender())) ||

(m\_partnerDescription.equals('A')))

{

return true;

}

}

}

return false;

}

### toSafeString

/\*\*

\* Returns safe string -- no passwords

\* @return Safe Strin

\*/

@Override

public String toSafeString()

{

String l\_toString = super.toSafeString() + "\nAdvert: " + m\_advert;

l\_toString += "\n\n";

return l\_toString;

}

### toString

/\*\*

\* Return Advertiser details in string

\* @return Advertiser String

\*/

@Override

public String toString()

{

String l\_toString = super.toString() + "\nAdvert: " + m\_advert + "\nPartner Description" + m\_partnerDescription.toString();

for (Reply reply : m\_replies)

{

l\_toString += reply.toString();

}

l\_toString += "\n";

return l\_toString;

}

}

## Responder.java

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package question2;

import java.util.ArrayList;

/\*\*

\* Extends Customer, Responder replies to Advertisers

\* @author David Nankivell

\* 16/04/2017

\* Responder.java

\* Class extends Customer, is a Responder to sought matches

\*/

public class Responder extends Customer {

private ArrayList<Customer> m\_matches = new ArrayList<>();

### Responder

/\*\*

\* Constructor

\* @param p\_userName Username of Responder

\* @param p\_gender Password of Responder

\* @param p\_age Age of Responder

\* @param p\_income Income of Responder

\*/

Responder(String p\_userName, String p\_gender,

int p\_age, float p\_income)

{

super(p\_userName,p\_gender, p\_age, p\_income);

}

### AddMatch

/\*\*

\* Add a customer to Responder's matches

\* @param p\_match Customer to add

\* @return Boolean regarding success of add operation

\*/

public boolean AddMatch(Customer p\_match)

{

return m\_matches.add(p\_match);

}

### RemoveMatch

/\*\*

\* Remove a customer from Responder's matches

\* @param p\_match Customer to remove

\* @return Boolean regarding success of remove operation

\*/

public boolean RemoveMatch(Customer p\_match)

{

return m\_matches.remove(p\_match);

}

### toSafeString

/\*\*

\* Returns safe string -- no passwords

\* @return Safe String with Responder details

\*/

@Override

public String toSafeString()

{

String l\_toString = super.toSafeString();

l\_toString += "\n\n";

return l\_toString;

}

### toString

/\*\*

\* All Responder details to string

\* @return String with Responder details

\*/

@Override

public String toString()

{

String l\_toString = super.toString();

for (Customer i\_customer : m\_matches)

{

l\_toString += "\nMatches: " + i\_customer.toString();

}

return l\_toString;

}

}

## PartnerSought.java

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package question2;

/\*\*

\* Details of Partner Sought by Advertiser

\* @author David Nankivell

\* 16/04/2017

\* PartnerSought.java

\* This class contains details of partners sought by Advertisers

\*/

public class PartnerSought {

private String m\_gender;

private int m\_minAge;

private int m\_maxAge;

private float m\_minIncome;

private float m\_maxIncome;

### PartnerSought

/\*\*

\* Constructor

\* @param p\_gender Gender of PartnerSought

\* @param p\_minAge Minimum Age of PartnerSought

\* @param p\_maxAge Maximum Age of PartnerSought

\* @param p\_minIncome Minimum Income of PartnerSought

\* @param p\_maxIncome Maximum Income of PartnerSought

\*/

PartnerSought(String p\_gender, int p\_minAge, int p\_maxAge,

float p\_minIncome, float p\_maxIncome)

{

m\_gender = p\_gender;

m\_minAge = p\_minAge;

m\_maxAge = p\_maxAge;

m\_minIncome = p\_minIncome;

m\_maxIncome = p\_maxIncome;

}

### GetGender

/\*\*

\* Returns gender of PartnerSought

\* @return Gender of PartnerSought

\*/

public String GetGender()

{

return m\_gender;

}

### GetMinAge

/\*\*

\* Returns the minimum age of PartnerSought

\* @return Minimum Age of PartnerSought

\*/

public int GetMinAge()

{

return m\_minAge;

}

### GetMaxAge

/\*\*

\* Returns the maximum age of PartnerSought

\* @return Maximum Age of PartnerSought

\*/

public int GetMaxAge()

{

return m\_maxAge;

}

### GetMinIncome

/\*\*

\* Returns the minimum income of PartnerSought

\* @return Minimum Income of PartnerSought

\*/

public float GetMinIncome()

{

return m\_minIncome;

}

### GetMaxIncome

/\*\*

\* Returns the maximum income of PartnerSought

\* @return Maximum Income of PartnerSought

\*/

public float GetMaxIncome()

{

return m\_maxIncome;

}

### SetGender

/\*\*

\* Sets the gender of the PartnerSought

\* @param p\_gender Gender of PartnerSought

\*/

public void SetGender(String p\_gender)

{

m\_gender = p\_gender;

}

### SetMinAge

/\*\*

\* Sets the minimum age of PartnerSought

\* @param p\_minAge Minimum Age of PartnerSought

\*/

public void SetMinAge(int p\_minAge)

{

m\_minAge = p\_minAge;

}

### SetMaxAge

/\*\*

\* Sets the maximum age of PartnerSought

\* @param p\_maxAge Maximum Age of PartnerSought

\*/

public void SetMaxAge(int p\_maxAge)

{

m\_maxAge = p\_maxAge;

}

### SetMinIncome

/\*\*

\* Sets the minimum income of PartnerSought

\* @param p\_minIncome Minimum Income of PartnerSought

\*/

public void SetMinIncome(float p\_minIncome)

{

m\_minIncome = p\_minIncome;

}

### SetMaxIncome

/\*\*

\* Sets the maximum income of PartnerSought

\* @param p\_maxIncome Maximum Income of PartnerSought

\*/

public void SetMaxIncome(float p\_maxIncome)

{

m\_maxIncome = p\_maxIncome;

}

### toString

/\*\*

\* Returns a string with all details of PartnerSought

\* @return PartnerSought String

\*/

@Override

public String toString()

{

String l\_toString = "\nGender: ";

if (m\_gender.equals("A"))

{

l\_toString += "Any";

}

else if (m\_gender.equals("M"))

{

l\_toString += "Male";

}

else if (m\_gender.equals("F"))

{

l\_toString += "Female";

}

l\_toString += "; Age Range: " + m\_minAge + "-" + m\_maxAge + "; Income Range: $" + m\_minIncome + "-$" + m\_maxIncome + "\n";

return l\_toString;

}

}

## Reply.java

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package question2;

/\*\*

\* A Reply to an Advertiser

\* @author David Nankivell

\* 16/04/2017

\* Reply.java

\* Class to handle replies in Dating Service program

\*/

public class Reply {

private Customer m\_owner;

private String m\_reply;

### Reply

/\*\*

\* Constructor

\* @param p\_cust Customer owner of Reply

\* @param p\_reply String reply message

\*/

Reply(Customer p\_cust, String p\_reply)

{

m\_owner = p\_cust;

m\_reply = p\_reply;

}

### SetReply

/\*\*

\* Set the message of the reply

\* @param p\_reply String reply message

\*/

public void SetReply(String p\_reply)

{

m\_reply = p\_reply;

}

### SetOwner

/\*\*

\* Set the owner of the reply

\* @param p\_owner Customer reply owner

\*/

public void SetOwner(Customer p\_owner)

{

m\_owner = p\_owner;

}

### GetReply

/\*\*

\* Get reply message

\* @return String reply message

\*/

public String GetReply()

{

return m\_reply;

}

### GetOwner

/\*\*

\* Get reply owner

\* @return Customer reply owner

\*/

public Customer GetOwner()

{

return m\_owner;

}

### toString

/\*\*

\* Reply details to string

\* @return String Reply details

\*/

@Override

public String toString()

{

String l\_toString = "Owner: " + m\_owner + "\nReply Message: " + m\_reply + "\n";

return l\_toString;

}

}