



LAB EXAM 2

December 19TH 2022

1PM TO 3:30PM

This exam is open book

Use of notes is permitted

Use of resources in moodle is permitted

Google, etc, is **not** permitted

There are 4 questions in this exam - answer all questions.

TOTAL MARKS: 100

Question 1: 10 marks

Question 2: 20 marks

Question 3: 25 marks

Question 4: 45 marks

PLEASE ENSURE THAT YOUR CODE COMPILES AND RUNS FOR EACH QUESTION. CODE THAT DOES NOT COMPILE WILL BE DOCKED MARKS. IF YOU HAVE CODE OR BLOCKS OF CODE THAT DOES NOT COMPILE, ENSURE THAT IT IS COMMENTED OUT PRIOR TO SUBMISSION.

ENSURE YOU UPLOAD A ZIP OF THE ENTIRE FOLDER CONTAINING ALL YOUR JAVA FILES, CLASS FILES AND ANY OTHER ASSOCIATED FILES CREATED FOR THIS ASSESSMENT.

Getting Started

- For each question create a java file called **FirstnameSurnameQuestionX.java** using your own first name and surname accordingly, where X is the number of the question. Rename the class name to match your Java program file name.

Eg:

JoeBidenQuestion1.java

JoeBidenQuestion2.java

JoeBidenQuestion3.java

JoeBidenQuestion4.java

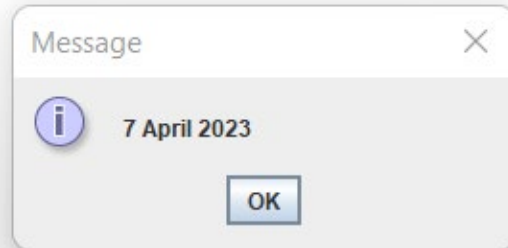
- Add your name, student ID and today's date as comments to the top of each program.

Requirements

- Ensure your code has meaningful variable names
- Ensure your code has appropriate use of space and indentation
- Ensure any non-working code is commented out prior to program submission
- Use comments throughout your program to describe the functions of blocks of code

Question 1 (10 marks)

Write a program that generates a random date in 2023, and outputs this date using JOptionPane. Your program should generate a random month, along with a random day (the day can be in the range of 1 to 28). Your output should be similar to as shown below.



Question 2 (20 marks)

Write a program that uses a **while loop** that will count upwards, starting from a number specified by the user and a finishing with a number specified by the user. The program should increment in steps of 0.05, and should output in lines of 10 numbers as shown below (for example, first line is from 10.00 to 10.45).

Your program should also include input validation so that the user cannot enter a finishing number that is lower than the starting number. If the user enters a finishing number that is lower than the starting number, a message is output stating "Finishing number must be greater than starting number", and the user will be prompted again to enter another number. This should continue until a number greater than the starting number is entered.

Sample output:

User is prompted to enter a starting and finishing number.

```
C:\Windows\system32\cmd.exe
Enter starting number: 10
Enter finishing number: 15
10.00, 10.05, 10.10, 10.15, 10.20, 10.25, 10.30, 10.35, 10.40, 10.45,
10.50, 10.55, 10.60, 10.65, 10.70, 10.75, 10.80, 10.85, 10.90, 10.95,
11.00, 11.05, 11.10, 11.15, 11.20, 11.25, 11.30, 11.35, 11.40, 11.45,
11.50, 11.55, 11.60, 11.65, 11.70, 11.75, 11.80, 11.85, 11.90, 11.95,
12.00, 12.05, 12.10, 12.15, 12.20, 12.25, 12.30, 12.35, 12.40, 12.45,
12.50, 12.55, 12.60, 12.65, 12.70, 12.75, 12.80, 12.85, 12.90, 12.95,
13.00, 13.05, 13.10, 13.15, 13.20, 13.25, 13.30, 13.35, 13.40, 13.45,
13.50, 13.55, 13.60, 13.65, 13.70, 13.75, 13.80, 13.85, 13.90, 13.95,
14.00, 14.05, 14.10, 14.15, 14.20, 14.25, 14.30, 14.35, 14.40, 14.45,
14.50, 14.55, 14.60, 14.65, 14.70, 14.75, 14.80, 14.85, 14.90, 14.95,
Press any key to continue . . .
```

Alternate example:

```
C:\Windows\system32\cmd.exe
Enter starting number: 45
Enter finishing number: 47
45.00, 45.05, 45.10, 45.15, 45.20, 45.25, 45.30, 45.35, 45.40, 45.45,
45.50, 45.55, 45.60, 45.65, 45.70, 45.75, 45.80, 45.85, 45.90, 45.95,
46.00, 46.05, 46.10, 46.15, 46.20, 46.25, 46.30, 46.35, 46.40, 46.45,
46.50, 46.55, 46.60, 46.65, 46.70, 46.75, 46.80, 46.85, 46.90, 46.95,
47.00, Press any key to continue . . .
```

Alternate example:

```
C:\Windows\system32\cmd.exe
Enter starting number: 10
Enter finishing number: 6
Finishing number must be greater than starting number!
Enter finishing number: 9
Finishing number must be greater than starting number!
Enter finishing number: 11
10.00, 10.05, 10.10, 10.15, 10.20, 10.25, 10.30, 10.35, 10.40, 10.45,
10.50, 10.55, 10.60, 10.65, 10.70, 10.75, 10.80, 10.85, 10.90, 10.95,
Press any key to continue . . .
```

Question 3 (25 Marks)

Using the random class, write a program that will output three **different** “random” names from a list, as shown below. The result can be output to the console as shown below. **Names should not be repeated**. Your solution should include an appropriate use of a **for loop**.

C:\Windows\system32\cmd.exe

```
This program will select 3 random names from the following:
Gary, Helen, Ian, Jane, Kevin, Dave, Mary, Noel, Bob, Alice

First random name is: Noel
Second random name is: Dave
Third random name is: Bob
Press any key to continue . . .
```

Alternate output:

C:\Windows\system32\cmd.exe

```
This program will select 3 random names from the following:
Gary, Helen, Ian, Jane, Kevin, Dave, Mary, Noel, Bob, Alice

First random name is: Gary
Second random name is: Alice
Third random name is: Mary
Press any key to continue . . .
```

Alternate output:

C:\Windows\system32\cmd.exe

```
This program will select 3 random names from the following:
Gary, Helen, Ian, Jane, Kevin, Dave, Mary, Noel, Bob, Alice

First random name is: Helen
Second random name is: Bob
Third random name is: Noel
Press any key to continue . . .
```

Question 4 (45 marks)

You are required to create a billing program for a gym that will allow the user to input values and generate receipts by writing to files. All monetary values should be stored as Floats.

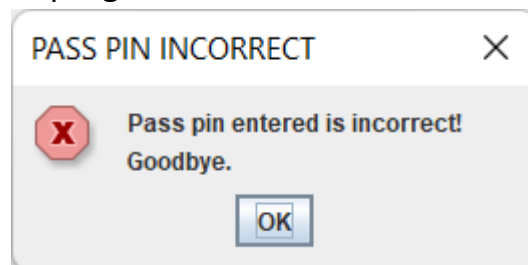
1: System login

The system should confirm login by generating a **random** pin code. The pin code should be in the form of a letter (either A, B or C) and three digits (for example, B345). This pin code should be different each time the program runs.

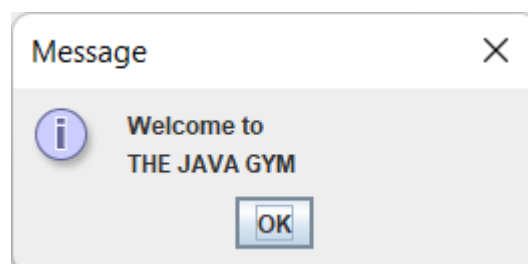
When the program starts, the user is prompted to enter the pin code:



If the passpin entered by the user matches, the user has logged on. Otherwise, a message is displayed as shown and the program ends:



Successful login shows the following message:

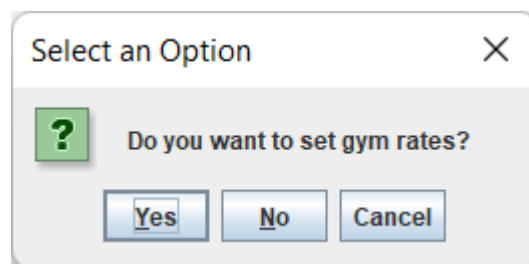


2: Setting Gym Rates

Standard default rate for using the gym facilities are as follows:

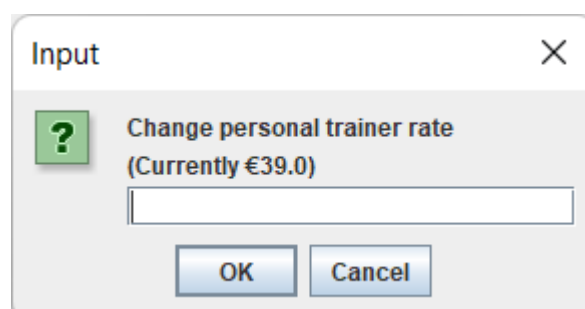
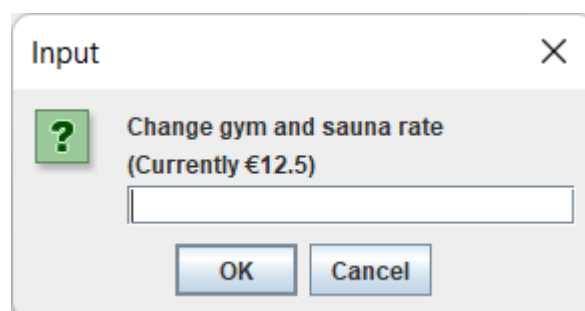
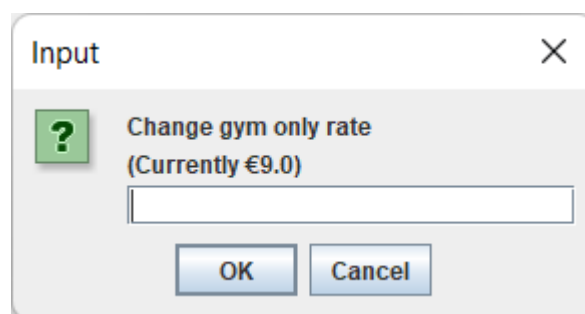
- *Gym only rate is €9.00 – Membership numbers 100 to 199*
- *Sauna rate is €12.50 – Membership numbers 200 to 299*
- *Personal Trainer rate is €39.00 – Membership numbers 300 to 399*

The user is prompted to change the standard gym rates if required:



Clicking “No” will not change the rates.

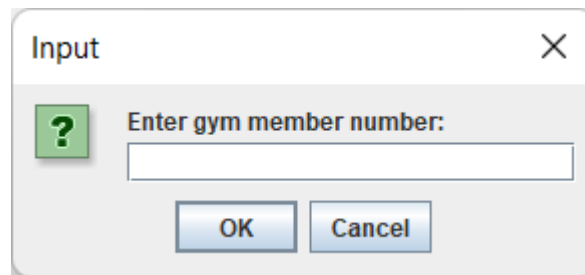
Clicking “Yes” will allow the user to enter new rates, eg:



If the above rates are changed, then these are the rates used to calculate the cost for the gym member.

3: Adding charges to gym member

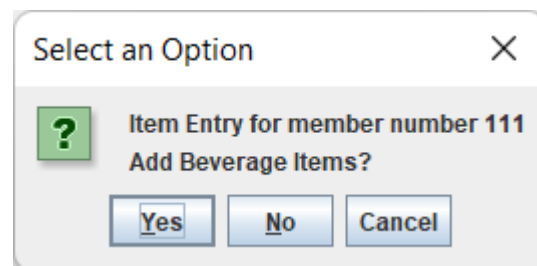
Charges for various items (beverages, snacks and gift shop) can be added to the member bill. The user is prompted to enter the gym member number:



An 'Input' dialog box with a close button (X) in the top right corner. It contains a green square icon with a white question mark on the left. To the right of the icon is the text 'Enter gym member number:' followed by a text input field. Below the input field are two buttons: 'OK' and 'Cancel'.

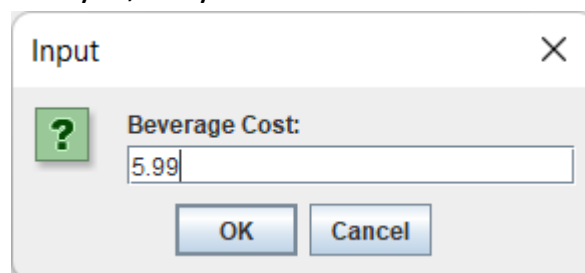
(Gym member numbers should be stored as integers)

The user is then prompted to add beverage items:



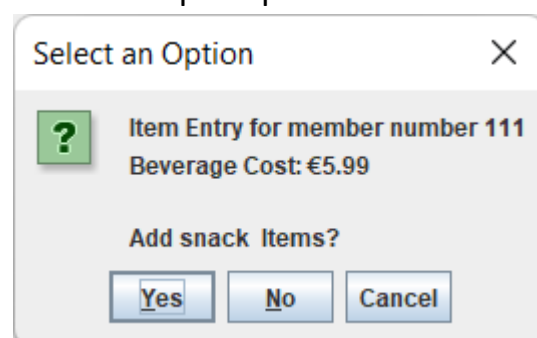
A 'Select an Option' dialog box with a close button (X) in the top right corner. It contains a green square icon with a white question mark on the left. To the right of the icon is the text 'Item Entry for member number 111' followed by 'Add Beverage Items?'. Below this text are three buttons: 'Yes', 'No', and 'Cancel'.

If the user clicks yes, they can add the cost of the bar items, eg:



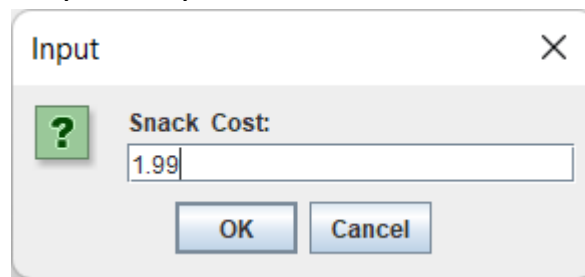
An 'Input' dialog box with a close button (X) in the top right corner. It contains a green square icon with a white question mark on the left. To the right of the icon is the text 'Beverage Cost:' followed by a text input field containing the value '5.99'. Below the input field are two buttons: 'OK' and 'Cancel'.

The user is then prompted to add snack items:



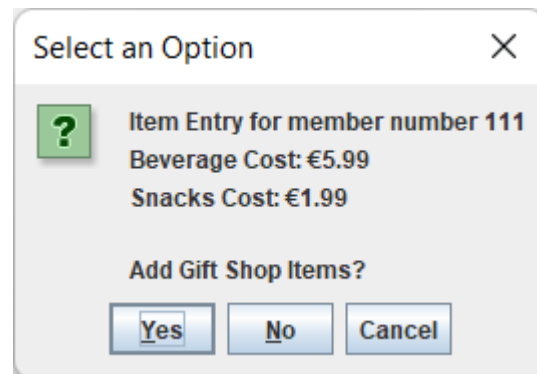
A 'Select an Option' dialog box with a close button (X) in the top right corner. It contains a green square icon with a white question mark on the left. To the right of the icon is the text 'Item Entry for member number 111' followed by 'Beverage Cost: €5.99'. Below this text is the text 'Add snack Items?'. At the bottom are three buttons: 'Yes', 'No', and 'Cancel'.

If the user clicks yes, they can add the cost of the snack items, eg:



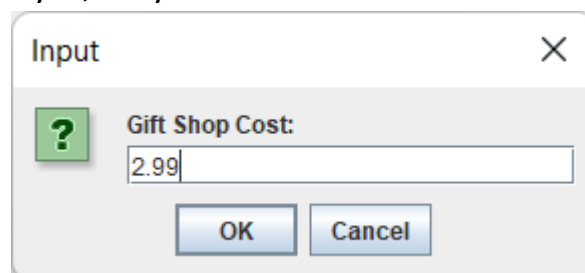
A dialog box titled "Input" with a close button (X) in the top right corner. It contains a green square icon with a white question mark. To the right of the icon is the text "Snack Cost:" followed by a text input field containing the value "1.99". At the bottom of the dialog are two buttons: "OK" and "Cancel".

The user is then prompted to add Gift Shop items:



A dialog box titled "Select an Option" with a close button (X) in the top right corner. It contains a green square icon with a white question mark. To the right of the icon is the text "Item Entry for member number 111". Below this text are two lines: "Beverage Cost: €5.99" and "Snacks Cost: €1.99". Below these lines is the text "Add Gift Shop Items?". At the bottom of the dialog are three buttons: "Yes", "No", and "Cancel".

If the user clicks yes, they can add the cost of the Gift Shop items, eg:



A dialog box titled "Input" with a close button (X) in the top right corner. It contains a green square icon with a white question mark. To the right of the icon is the text "Gift Shop Cost:" followed by a text input field containing the value "2.99". At the bottom of the dialog are two buttons: "OK" and "Cancel".

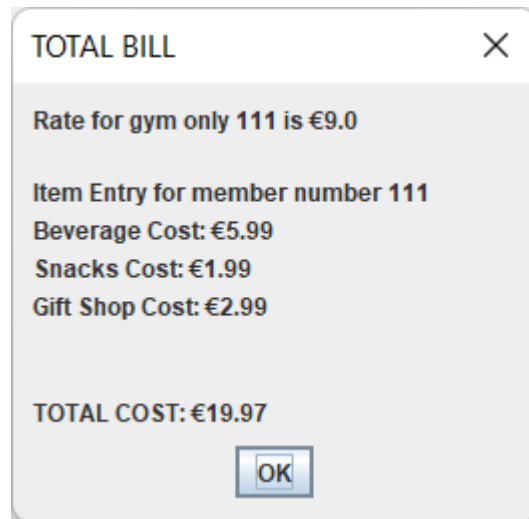
The total bill is then presented on screen, as shown.

The member rates are calculated as follows:

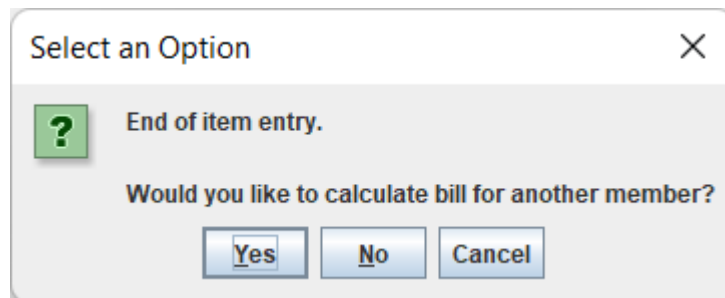
Members 100 – 199 are gym use only

Members 200 – 299 are sauna use

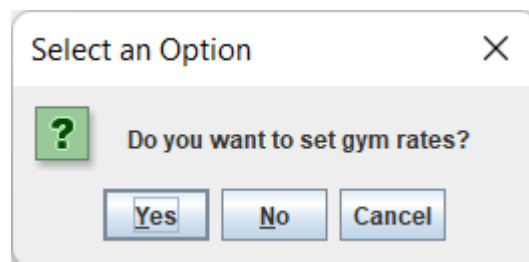
Members 300 – 399 have personal trainers



The user is then prompted on whether they would like to calculate the bill for another member:



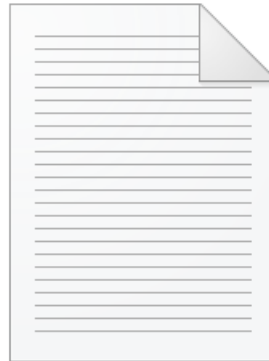
If Yes is clicked, then the user is taken back to the set gym rates prompt:



If the user clicks no, the program terminates.

4: Generating receipts

The program should also generate an individual bill as a “.txt” file format for each gym member. For example, gym member 111 would have the following receipt generated (in a file called 111.txt):



✓ 111.txt

File contents:

```
111.txt - Notepad
File Edit View
Charges for Gym Member 111
=====

Rate for gym only 111 is €9.0

Item Entry for member number 111
Beverage Cost: €5.99
Snacks Cost: €1.99
Gift Shop Cost: €2.99

TOTAL COST: €19.97

Ln 1, Col 1 | 100% | Windows (CRLF) | ANSI
```



✓ All Items.txt

The program should also generate a total record as a “.txt” file format (in a file called *All Items.txt*) for all gym member purchases. For example:

A screenshot of a Notepad window titled '*All Items.txt - Notepad'. The window has a menu bar with 'File', 'Edit', and 'View' options. The text content is as follows:

```
Member 111 - Beverage Cost: €5.99
Member 111 - snack Cost: €1.99
Member 111 - Gift Shop Cost: €2.99
RATE FOR 111: €9.0

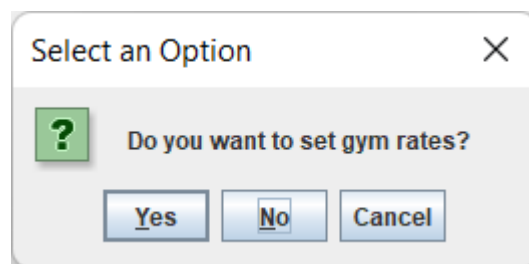
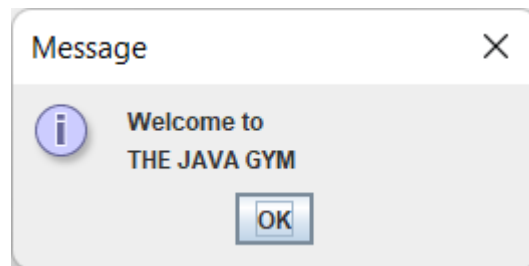
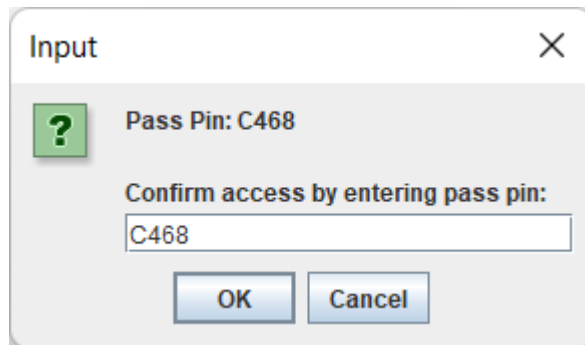
Member 222 - snack Cost: €5.0
RATE FOR 222: €12.5
|
Member 333 - Beverage Cost: €6.99
Member 333 - Gift Shop Cost: €25.5
RATE FOR 333: €39.0
```

The status bar at the bottom shows 'Ln 8, Col 1', '100%', 'Windows (CRLF)', and 'ANSI'.

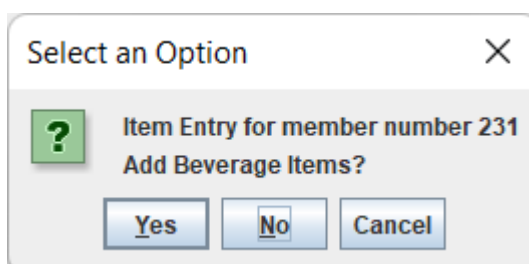
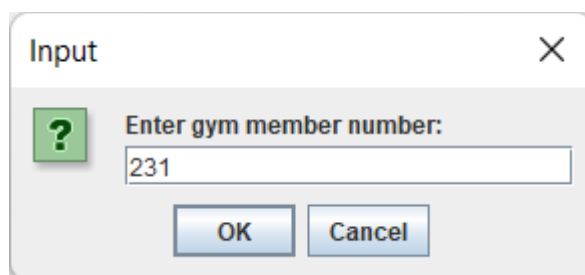
All files should be saved in a folder called bills

Alternate example of Running program:

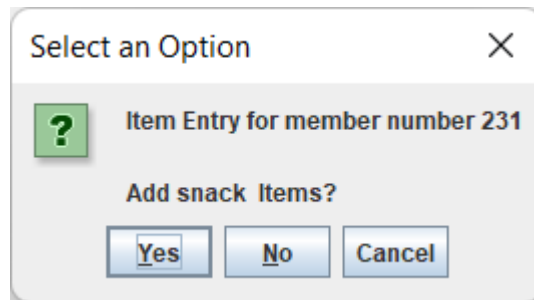
The following are screen shots of the program running with alternate inputs:



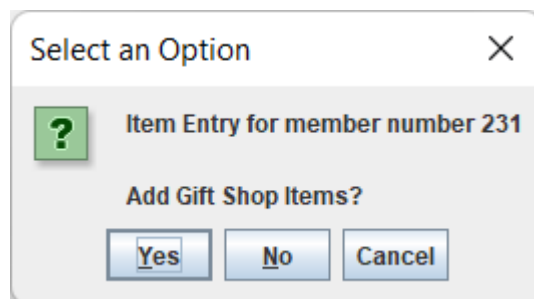
No is selected



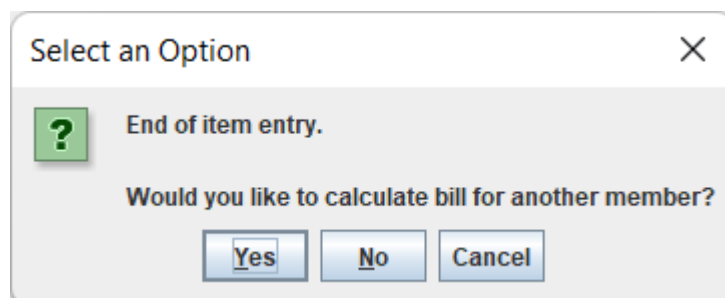
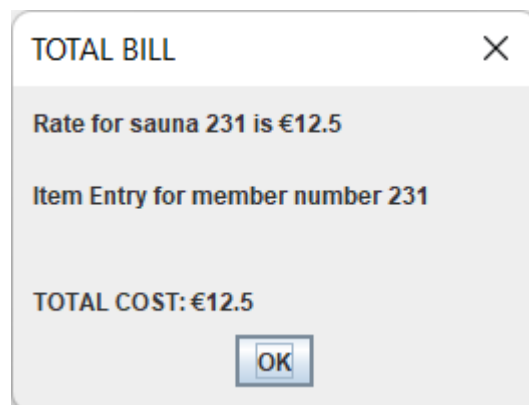
No is selected



No is selected




No is selected




Yes is selected

Select an Option ✕


 Do you want to set gym rates?

Yes is selected


Input ✕

 Change gym only rate
(Currently €9.0)


Input ✕

 Change gym and sauna rate
(Currently €12.5)

Input ✕

 Change personal trainer rate
(Currently €39.0)

Input ✕

 Enter gym member number:

Input

Beverage Cost:

1.00

OK Cancel

No is selected on the next 2 options

TOTAL BILL

Rate for personal trainer 305 is €35.0

Item Entry for member number 305
Beverage Cost: €1.0

TOTAL COST: €48.5

OK

Select an Option

End of item entry.

Would you like to calculate bill for another member?

Yes No Cancel

No is selected

Files generated:

```

All Items.txt - Notepad
File Edit View
RATE FOR 231: €12.5
Member 305 - Beverage Cost: €1.0
RATE FOR 305: €35.0
Ln 1, Col 1 | 100% | Windows (CRLF) | ANSI

```

```

305.txt - Notepad
File Edit View
Charges for Gym Member 305
-----
Rate for personal trainer 305 is €35.0
Item Entry for member number 305
Beverage Cost: €1.0
TOTAL COST: €48.5
Ln 1, Col 1 | 100% | Windows (CRLF) | ANSI

```

```

231.txt - Notepad
File Edit View
Charges for Gym Member 231
-----
Rate for sauna 231 is €12.5
Item Entry for member number 231
TOTAL COST: €12.5
Ln 1, Col 1 | 100% | Windows (CRLF) | ANSI

```


TO SUBMIT YOUR EXAM

UPLOAD LOCATION IS AT TOP OF MOODLE PAGE

Upload your work to Moodle

ENSURE YOU UPLOAD A ZIP OF THE ENTIRE FOLDER CONTAINING ALL YOUR JAVA FILES, CLASS FILES AND ANY OTHER ASSOCIATED FILES CREATED TODAY.

1. Navigate to the location of the folder where you saved all your work for today's lab.
2. Right-click on the folder, and select "Send to", then select "Compressed (zipped) folder". This will create a compressed version of any files you have worked on for the lab.
3. There should be a new compressed file created. This is the file that you will need to upload to Moodle.
4. In Moodle, navigate to the "LAB Exam 2 – Monday December 19th" section:
5. Click the upload link provided (UPLOAD YOUR JAVA LAB EXAM HERE):
6. Click "add submission" and add the ZIP file you created here. Make sure you complete the submission process to ensure that your lab work has been submitted.