CS234 Computer Science II

Lab 3
Total points: 100

Read the instructions carefully.

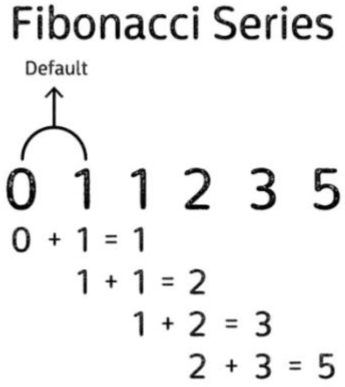
The objective of this assignment is to demonstrate your understanding of control structures in Java, specifically **conditionals and loops**, by creating a text-based menu-driven program *without* the use of any custom methods (we will talk about methods next week).

You are required to create a *Java program* that implements a **menu and its corresponding actions** using **only** conditional statements and loops. The program should provide the user with a menu of options, allow them to choose an option, and then perform the **corresponding action** based on their choice.
The menu should **keep working** until the user chooses to exit.
For this lab, you are not allowed to create methods for the different tasks.

Menu Options:
1. Get Fibonacci Sequence
2. Check Perfect Number
3. Find the Product of Digits
4. Reverse a Word
5. Exit

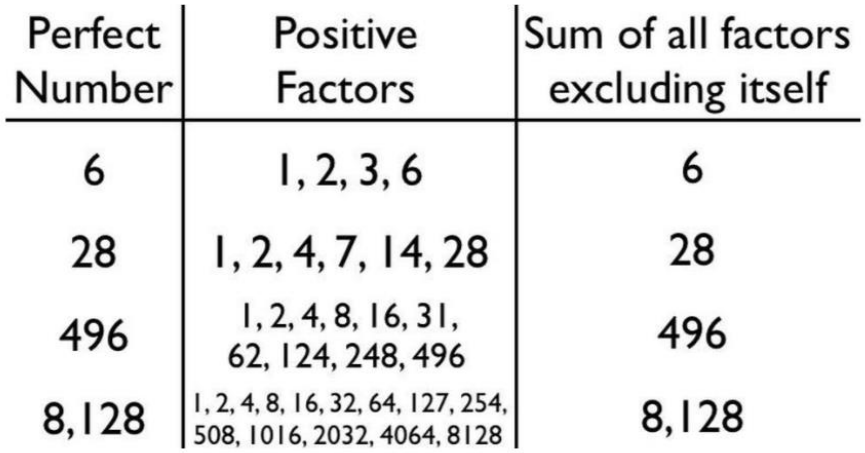
Functionality of Menu Options:

1. Get Fibonacci Series:
- Prompt the user to enter a ***positive integer***.
- Continue asking for input if the user inputs a negative number.
- Calculate and display the Fibonacci sequence for the position represented by the input number.
- Return to the main menu after displaying the result.



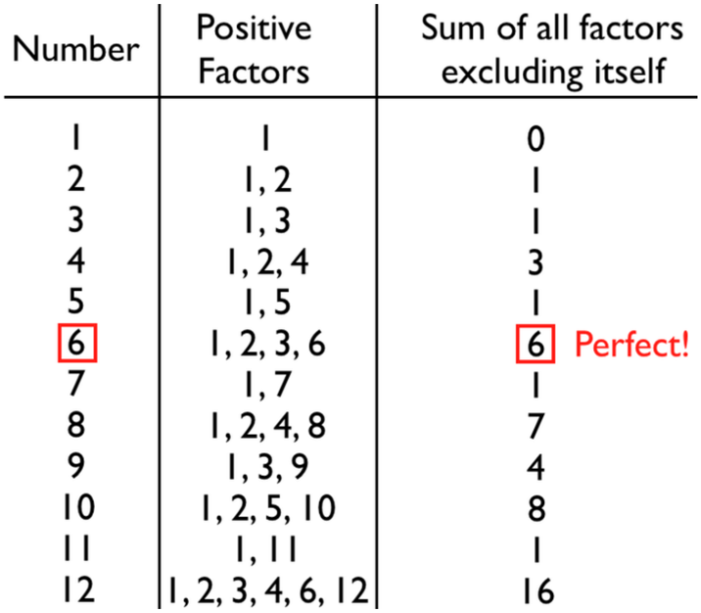
2. Check Perfect Number:
- Prompt the user to enter a *positive* ***integer greater than 1.***
- Continue asking for input until a valid positive integer greater than 1 is provided.
- Determine if the entered number is perfect or not.

- Display whether the number is perfect or not.
- Return to the main menu after displaying the result.



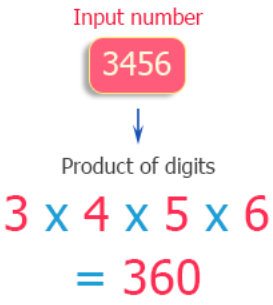
Perfect Number	Positive Factors	Sum of all factors excluding itself
6	1, 2, 3, 6	6
28	1, 2, 4, 7, 14, 28	28
496	1, 2, 4, 8, 16, 31, 62, 124, 248, 496	496
8,128	1, 2, 4, 8, 16, 32, 64, 127, 254, 508, 1016, 2032, 4064, 8128	8,128

How to determine if a number is Perfect.



Number	Positive Factors	Sum of all factors excluding itself
1	1	0
2	1, 2	1
3	1, 3	1
4	1, 2, 4	3
5	1, 5	1
6	1, 2, 3, 6	6 Perfect!
7	1, 7	1
8	1, 2, 4, 8	7
9	1, 3, 9	4
10	1, 2, 5, 10	8
11	1, 11	1
12	1, 2, 3, 4, 6, 12	16

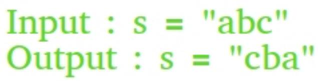
3. Find the Product of Digits:
- Prompt the user to enter a **positive integer with at least two digits**.
- Continue asking for input until a valid positive integer with at least two digits is provided.
- Calculate and display the multiplication of the digits of the entered number.
- Return to the main menu after displaying the result.



4. Reverse a Word:
- Prompt the user to enter a **string**.
- Reverse the string and display the result.
- Return to the main menu after displaying the result.

Restrictions: Do not use any built-in method to reverse strings. Do not use arrays (we haven’t talk about them!)

Tip: Use a loop with the charAt() method.


Input : s = "abc"
Output : s = "cba"

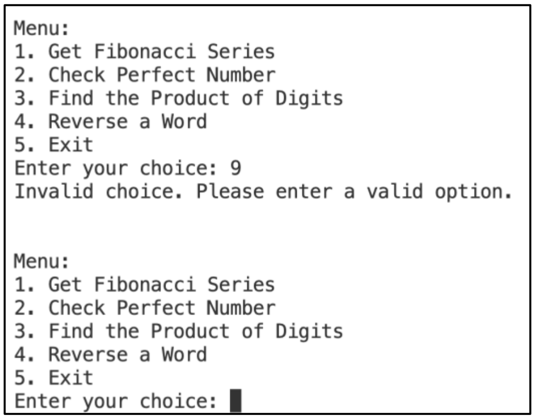
This link shows how to use the charAt() method
https://www.w3schools.com/java/ref_string_charat.asp

5. Exit:
   - Exit the program.

Additional Requirements:
- Handle invalid inputs gracefully (i.e., negative numbers, less than two digits) and display appropriate error messages.
- Ensure that the **program continues running until** the user chooses to exit.
- Use only conditional statements for menu option selection.
- Use loops to control the flow of the program.
- **Do not create methods or functions** for this assignment (we haven’t talked about that yet!).
- You can use the *Scanner* and *Math* classes.
- Use a **Switch** for your **menu** options.

Examples of its use:

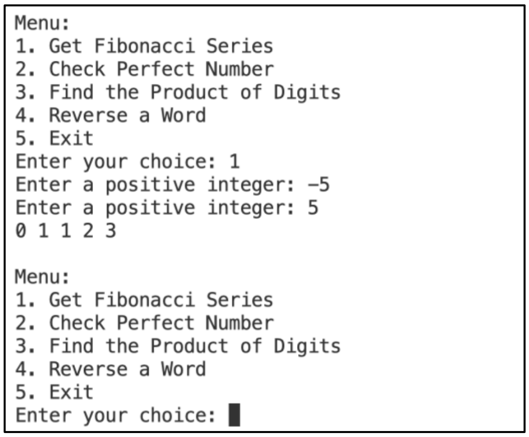
- Invalid option



```
Menu:
1. Get Fibonacci Series
2. Check Perfect Number
3. Find the Product of Digits
4. Reverse a Word
5. Exit
Enter your choice: 9
Invalid choice. Please enter a valid option.


Menu:
1. Get Fibonacci Series
2. Check Perfect Number
3. Find the Product of Digits
4. Reverse a Word
5. Exit
Enter your choice: █
```

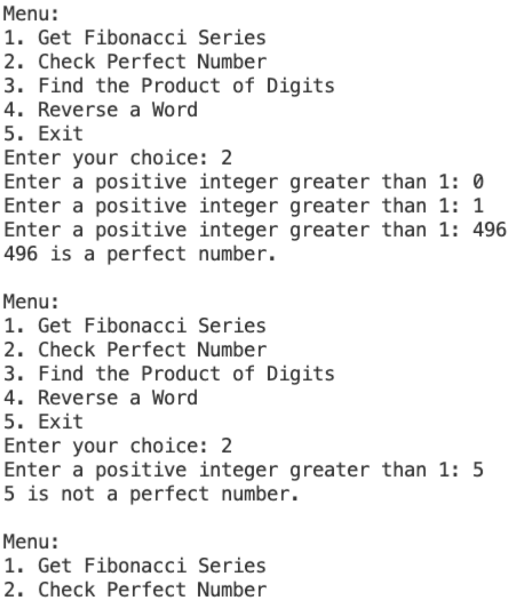
- Option 1



```
Menu:
1. Get Fibonacci Series
2. Check Perfect Number
3. Find the Product of Digits
4. Reverse a Word
5. Exit
Enter your choice: 1
Enter a positive integer: -5
Enter a positive integer: 5
0 1 1 2 3

Menu:
1. Get Fibonacci Series
2. Check Perfect Number
3. Find the Product of Digits
4. Reverse a Word
5. Exit
Enter your choice: █
```

- Option 2

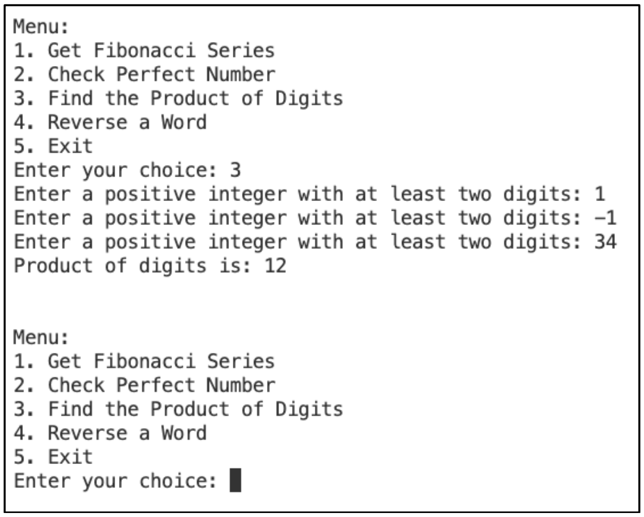


```
Menu:
1. Get Fibonacci Series
2. Check Perfect Number
3. Find the Product of Digits
4. Reverse a Word
5. Exit
Enter your choice: 2
Enter a positive integer greater than 1: 0
Enter a positive integer greater than 1: 1
Enter a positive integer greater than 1: 496
496 is a perfect number.

Menu:
1. Get Fibonacci Series
2. Check Perfect Number
3. Find the Product of Digits
4. Reverse a Word
5. Exit
Enter your choice: 2
Enter a positive integer greater than 1: 5
5 is not a perfect number.

Menu:
1. Get Fibonacci Series
2. Check Perfect Number
```

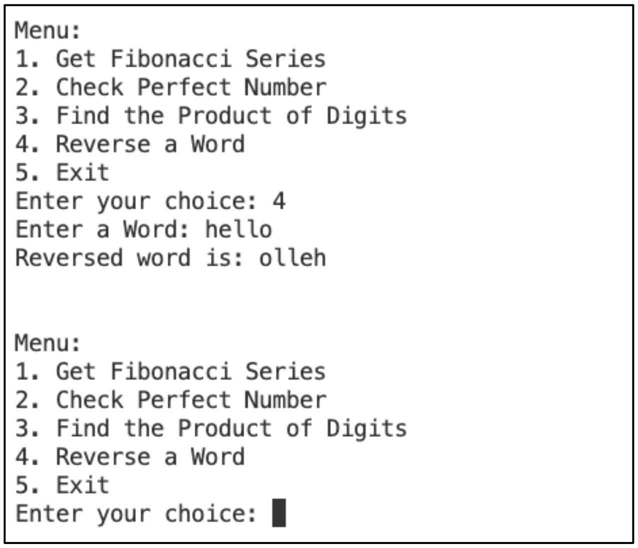
- Option 3



```
Menu:
1. Get Fibonacci Series
2. Check Perfect Number
3. Find the Product of Digits
4. Reverse a Word
5. Exit
Enter your choice: 3
Enter a positive integer with at least two digits: 1
Enter a positive integer with at least two digits: -1
Enter a positive integer with at least two digits: 34
Product of digits is: 12


Menu:
1. Get Fibonacci Series
2. Check Perfect Number
3. Find the Product of Digits
4. Reverse a Word
5. Exit
Enter your choice: █
```

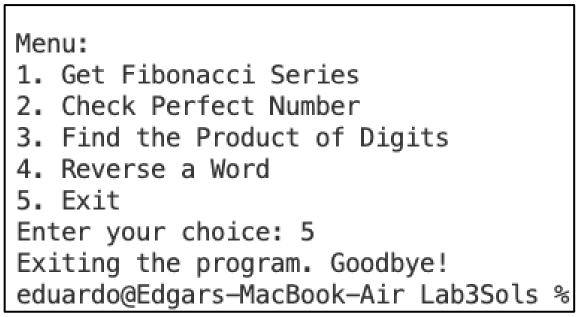
- Option 4



```
Menu:
1. Get Fibonacci Series
2. Check Perfect Number
3. Find the Product of Digits
4. Reverse a Word
5. Exit
Enter your choice: 4
Enter a Word: hello
Reversed word is: olleh


Menu:
1. Get Fibonacci Series
2. Check Perfect Number
3. Find the Product of Digits
4. Reverse a Word
5. Exit
Enter your choice: █
```

- Option 5



```
Menu:
1. Get Fibonacci Series
2. Check Perfect Number
3. Find the Product of Digits
4. Reverse a Word
5. Exit
Enter your choice: 5
Exiting the program. Goodbye!
eduardo@Edgars-MacBook-Air Lab3Sols %
```

If your program terminates after each option, then it is not correct!

Submission details:

Upload a **single ZIP** file.
Name your file as follows: **Lab3_Lastname_Firstname.zip**

Your **.zip** file must contain the following:

1. Your **.java** source file (no .class).
2. A .txt file (readme.txt) with simple instruction on how to **compile** and **execute** your programs (I reviewed in class how to compile your .java files)
3. A **SINGLE PDF** with **screenshots** from your program running. Show the use of the different menu options. Do not send .jpg files.

In each .java file, write as a *multiline* comment at the beginning of the file the following: Your name

The **zip** file must be uploaded to Canvas. I do not accept answers via email or as comments on Canvas.
I do not accept image files; it must be a PDF file with the screenshots.

The **zip** file must be **uploaded to Canvas**. I **do not accept answers via email.**
I do not accept answers in the comments section.
I do not accept image files; it must be a PDF file.

Make sure to check the **due date** for this activity on Canvas. Try to submit it before the due date so you can have time to check for improvements.
Make sure you are **submitting the correct files**. I will grade the files uploaded to Canvas.
Use the ***javac* and *java*** commands **before** submitting your solution to test if they work outside any IDE.

Make sure to review the **grading rubric.**

Read all the instructions carefully.

Late submissions are not allowed.

Do not assume requirements, contact me if you have questions!

Ensure that your code is original and developed by you.