CS 101 - Algorithms & Programming I

Fall 2022 - Lab 3

Due: Week of October 17, 2022

*Remember the **honor code** for your programming assignments.*
*For all labs, your solutions must conform to the CS101 style **guidelines**!*
All data and results should be stored in variables (or constants) with meaningful names.

The objective of this lab is to learn how to program simple and complex decisions by using if/else statements. Remember that analyzing your problems and designing them on a piece of paper *before* starting implementation/coding is always a best practice.

In this particular lab, do **not** use a switch-case statement instead of if-else statement!

0. Setup Workspace

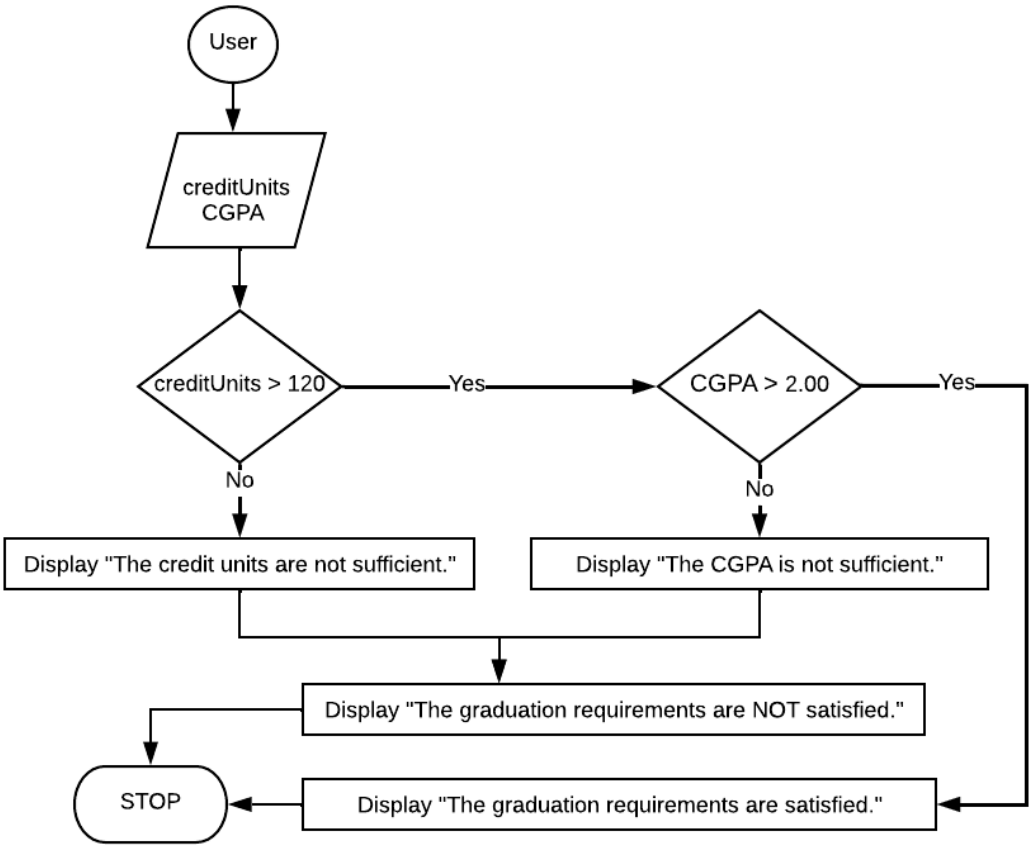
Start VSC and open the previously created workspace named `labs_ws`. Now, under the labs folder, create a new folder named `lab3`.

In this lab, you are to have three Java classes/files (under `labs/lab3` folder) as described below. A fourth Java file containing the revision should go under this folder as well. We expect you to **submit a total of 4 individual files** including the revision **without compressing** them.

The user inputs in the sample runs are shown in blue color.

1. Graduation Time

Create a new/empty file of your own under the `lab3` folder named `Lab03_Q1.java` with a class with the same name that takes an **integer** and a **double** as inputs from the user, where the integer represents the student's *credit units* and the double represents student's *cumulative grade point average (CGPA).*
Based on the input data, verify if the student meets the graduation requirements or not according to the flowchart below.



Sample runs:

```
Enter your credit units: 128
Enter your CGPA: 1.98
The CGPA is not sufficient.
Graduation requirements are NOT satisfied.
```

```
Enter your credit units: 97
Enter your CGPA: 2.10
The credit units are not sufficient.
Graduation requirements are NOT satisfied.
```

```
Enter your credit units: 135
Enter your CGPA: 3.64
Graduation requirements are satisfied.
```

2. Is My Number Palindromic?

Create a new/empty file of your own under the `lab3` folder named `Lab03_Q2.java` with a class with the same name that takes a positive **integer** as input from the user, displays its **digit number,** and verifies if it is a **palindromic number** or not.

The input integer must have a **minimum of 3 digits and a maximum of 7 digits.**
(e.g.100 and 999 are examples of 3-digit numbers, whereas 1000000 and 9999999 are 7-digit numbers.

A **palindromic number**[1] is a number that remains the same when its digits are reversed. In other words, it has reflectional symmetry across a vertical axis.

Examples of palindromic numbers: 0, 1, 44, 45554, 9870789.

Sample runs:

```
Enter a number: 33
The input is invalid! The number must be between 3 and 7 digits long.
The number has 2 digits.
```

```
Enter a number: 1234321
The number has 7 digits.
1234321 is a palindromic number.
```

```
Enter a number: 1441
The number has 4 digits.
1441 is a palindromic number.
```

```
Enter a number: 25462
The number has 5 digits.
```

[1] https://en.wikipedia.org/wiki/Palindromic_number

25462 is NOT a palindromic number.

3. Let's Create a PvP Game!

Create a new/empty file of your own under the `lab3` folder named `Lab03_Q3.java` with a class with the same name. Your program will be a small part of a PvP video game that allows the gamer to log in using a username and a password, upgrade the character's armor, add a buddy, and change his/her credentials.

The username and password must be stored as **Strings**. Assume that the username is "JavaWarrior", the password is "JavaRocks"; the character does not have a buddy and there are 600 coins available for this account.

The user must be able to perform the following operations in the game:

- Log in using a username and a password. If both are valid, display the operations shown below, otherwise display "Username not found! Bye!" if the username is wrong and "Wrong password! Bye!" if the password is wrong, then exit:

```
Enter your username: JavaWarrior
Enter your password: JavaRocks
1- Upgrade armor
2- Add a buddy
3- Change credentials
Select an operation:
```

```
Enter your username: Can
Username not found! Bye!
```

```
Enter your username: JavaWarrior
Enter your password: PythonRocks
Wrong password! Bye!
```

- To upgrade the character's armor, the user must enter 1. Each upgrade requires 150 coins. Display the chosen operation "Upgrade armor:" and the number of coins before the upgrade. If the user has the necessary coins, display "Your armor has been upgraded!" and the updated number of coins, else display "Not enough coins! Bye!" :

```
Upgrade armor:
You have 600 coins.
Your armor has been upgraded!
You have 450 coins. Bye!
```

```
Upgrade armor:
You have 90 coins.
Not enough coins! Bye!
```

- To add a buddy in the game, the user must enter 2. Display the chosen operation "Add a buddy:". If the user does not have a buddy, ask for the buddy's name. If the user already has a buddy, display "You already have a buddy!". Then, display the buddy's name and exit.

```
Add a buddy:
Enter your buddy's name: Suki
Your buddy is Suki. Bye!
```

```
Add a buddy:
You already have a buddy!
Your buddy is Suki. Bye!
```

- To change the credentials the user must enter 3 and select the username or password for this operation. Display the chosen operation "Change credentials:" and its options:

```
Change credentials:
1- Change username
2- Change password
Select an option:
```

- If the user enters 1, ask for a new username, display the new credentials, and exit:

```
Enter your new username: JavaHero
Your username is JavaHero
Your password is JavaRocks
Bye!
```

- If the user enters 2, ask for a new password (the password must have a maximum of 20 characters), display the new credentials, and exit:

```
Enter your new password: JavaIsEasy
Your username is JavaWarrior
Your password is JavaIsEasy
Bye!
```

A sample run for a user who wants to upgrade the character's armor:

```
Enter your username: JavaWarrior
Enter your password: JavaRocks
1- Upgrade armor
2- Add a buddy
3- Change credentials
Select an operation: 1
Upgrade armor:
You have 540 coins.
Your armor has been upgraded!
You have 390 coins. Bye!
```

A sample run for a user who wants to add a buddy:

```
Enter your username: JavaWarrior
Enter your password: JavaRocks
1- Upgrade armor
2- Add a buddy
3- Change credentials
Select an operation: 2
Add a buddy:
Enter your buddy's your name: Teddy
Your buddy is Teddy. Bye!
```

A sample run for a user who wants to change the credentials:

```
Enter your username: JavaWarrior
Enter your password: JavaRocks
1- Upgrade armor
2- Add a buddy
3- Change credentials
Select an operation: 3
Change credentials:
```

```
1- Change username
2- Change password
Select an option: 2
Enter your new password: LoveJava
Your username is JavaWarrior
Your password is LoveJava
Bye!
```

You can consider this as a one-session game where you run it from scratch each time. In other words, the username, password, number of coins, or buddy are not updated according to the previous run. However, you should be able to change these values and re-run your program to get different results.