Introduction to Programming – Autumn 2019
Assignment 4: Bank Application

Submission Deadline: Monday, 7th October 2019, 10:00 pm

Problem Statement

In this assignment, you have to develop a Banking application using the **"Switch Case"** and **"For Loop"** construct.

The first input to the program will be a **positive integer** number N, represents the number of operations a person wants to perform.

A person can perform one of the following operations on the application:

- WithdrawAmount
- AddAmount
- GetBalance

Based on the entered operation program will print the respective operation name, and perform the operation.

In case of any invalid input, your program should output "Error". For example, see the sample test cases.

For printing the operations, you have to use the **Switch Case** construct, and for the number of operations a person performs, you have to use the **For Loop** construct.

Note: - You have to use a **for loop** and **switch-case construct** for this application.

Input Format

First Input: First input to the program is an integer **N.**

Please note that **N** should be a positive integer and in case of any invalid input for **N,** your program should ask the user to enter the correct value again.

Second Input (I): Initial Balance of the person.

Next N lines: Based on selected first input there will be N inputs belong to one of these four operations (here X is an integer, and w,a, p are characters):

1. **w X** : Withdraw **X** (numeric value) from the users account
2. **a X**: Add **X** (numeric value) to the users account
3. **p**: GetBalance - Display the balance of the user account

In case of any invalid input (i.e., if a user enters any invalid input other then these 3 program should print “Error”.

Output Format

For each of the **N** inputs, print the corresponding operation name, and “Success” if the operation was successful, and “Error” if the operation fails (you can ignore the effects of that operation on the account balance). For GetBalance, instead of success you can print the current balance of the user.

Sample Test Cases

a)

INPUT 3 1000 w 500 a 200 p
Output *WithdrawAmount* *Success* *AddAmount* *Success* *GetBalance* *700*

b)

INPUT 3 500 w 1000 a 100 p
Output *WithdrawAmount* *Error* *AddAmount* *Success* *GetBalance* *600*

c)

INPUT -1
Output *Please enter a valid value for N.*

Submission Details:
Please submit the following information:

- **Source Code:** Your source program. The name of your file should be in this format: **Bank-roll no.c** where you replace “roll no” with your roll number.

- **Readme.txt:** In this file, you should explain how to compile and run your program. The name of your file should be in this format: **Bank_Readme-roll no.txt** where you replace “roll no” with your roll number.

- **Design.txt:** In this file, you explain the design of your program (control flow of your program). Your objective should be such that the TA reading this file should easily understand the working of your program. Please add details about how you have handled corner cases - i.e. for what inputs you have printed “Error”.
  The name of your file should be in this format: **Bank_Design-roll no.txt** where you replace “roll no” with your roll number.

Zip all these files and name it as **Bank-roll no.zip**.
Please follow the naming convention strictly. Otherwise, your program will not be evaluated.
Then, submit it on google classroom for this assignment by the above-mentioned deadline.

Evaluation Policy:
The TAs will use the following evaluation policy:

- Design: 30%
- Execution: 60%
- Indentation and Documentation (with comments): 10%

Late Submission Penalty:
For each day after the deadline, your submission will be penalized by 10 marks.