Problem Set 1

Published: Wednesday, June 4, 2025.

Due: Friday, June 13, 2025.

This problem set will introduce you to using control flow in Python and formulating a computational solution to a problem. You should save your code for the first problem as **firstname_lastname_teamXX_ps1.py**.

Collaboration

You may collaborate with your Team members only. However, each student should write up and hand in his or her assignment separately. *Be sure to indicate with whom you have worked in the comments of your submission.*

Before You Start: Read the Style Guide
Read the style guide sections 1, 2, and 3. You should follow the conventions laid out in the **PEP 8 – Style Guide for Python Code**

You are a software engineering intern at Telestar, a newly opened telecommunication company in Ghana. The company has decided to add a mobile money service to their list of services offered and you are part of the team working on this project. The service is required to replicate the MTN Mobile Money service. The application should allow customers to **Transfer Money, buy Airtime and Bundles, Allow Cash Out, and check/top up Wallet**.

The customers, with an initial amount of **GHS 1000** and a 4-digit **MOMO PIN "7209"**, should be given the following options on the main menu when they dial the hashcode ***150#**:

(Ensure that for an incorrectly entered hashcode, the user is prompted 3 times before the entire program terminates)

Welcome to Telestar Mobile Money! Please select an option:
1. Transfer Money
2. Airtime and Bundles
3. Allow Cash Out
4. My Wallet

You are going to explore these options for everyday transactions.

Part A: Transfer Money

In Part A, we are going to implement the money transfer functionality of the application given the following assumptions:

1. Call the phone number of the recipient, recipient's **phone_number**. This should follow the required format (e.g., 10 digit and starting with "059"" for Telestar network users and "050, 026, 023" for Other Networks)
2. Call the amount you wish to transfer, **transfer_amount**
3. A transfer made to any network will attract an E-levy charge of 1% of the transfer amount.
4. Transfers made between Telestar networks attract network service charge of 0.5% and transfers made between Telestar and Other networks attract a service charge of 0.75%

Write a program to implement this functionality. You will want your variables to be in the right format, so you should cast user inputs in the appropriate format. The program should allow the user to start a new transaction after completing a previous one, with options from the main display menu again. If no other transaction is started after, the program should terminate.

Your program should ask the user to enter the following variables:

1. The phone number of whomever you're sending the money to, (recipient's_phone_number)
2. The amount to be transferred, (transfer_amount)

Hints

To help you get started, here is a rough outline of the stages you should probably follow in writing your code:

- Retrieve user input. Look at input() if you need help with getting user input. For this problem set, you can assume that users will enter valid input (e.g., they won't enter a string when you expect a float)
- Initialize some state variables. You should decide what information you need. Be careful that the sending amount is not greater than the current balance.
- Inform the user whether the transaction was successful or not.

Try different inputs and see how the charges and new balances change. **Please make your program print results in the format shown in the test cases below.** Blue is the output of the program, black is the input of the user.

Test Case 1

```
>>>
Enter your choice: 1
Transfer Money:
1. TeleStar Network
2. Other Networks
Enter your transfer choice: 1
Enter the recipient's TeleStar phone number (should start with 059): 0597568294
Enter the amount to transfer: 30
GHS 30.0 has been sent successfully with an E-levy charge of GHS 0.3.
The service charge is GHS 0.1.
New balance: GHS 969.5.
Do you want to perform another operation? (yes/no):
```

Test Case 2

```
>>>
 Enter your choice: 1
Transfer Money:
1. TeleStar Network
2. Other Networks
Enter your transfer choice: 2
Enter the recipient's phone number (should start with 050/026/023): 0501035132
Enter the amount to transfer: 60
GHS 60.0 has been sent successfully with an E-levy charge of GHS 0.6.
The service charge is GHS 0.4.
New balance: GHS 908.5.
Do you want to perform another operation? (yes/no):
```

Part B: Airtime and Bundles

In Part B, we are going to implement a functionality that allows customers to buy either airtime or data with option 2, "**2. Airtime and Bundles**", given the following assumptions :

1. Call the amount of airtime to be bought, **airtime_amount**
2. Call the type of bundle the user wants to buy from this list of data bundles :
    - GHC 5 (280 MB)
    - GHC 10 (667 MB)
    - GHC 100 (10 GB)

Write a program that can buy airtime or data bundle when needed, from option 2.

Test Case 3

```
>>>
Enter your choice: 2
Airtime and Bundles:
1. Buy Airtime
2. Buy Bundles
Enter your choice: 1
Enter the amount to purchase: 50
GHS 50.0 airtime has been successfully purchased.
New balance: GHS 766.5
Do you want to perform another operation? (yes/no):
```

Test Case 4

```
>>>
Enter your choice: 2
Airtime and Bundles:
1. Buy Airtime
2. Buy Bundles
Enter your choice: 2
Bundles:
1. GHC 5 (280 MB)
2. GHC 10 (667 MB)
3. GHC 100 (10 GB)
Choose your bundle type: 2
667 MB Data Bundle successfully purchased.
```

New balance: GHS 756.5
Do you want to perform another operation? (yes/no):

Part C: Allow CashOut

Now your customers know they can save their money with the service. But how do they withdraw it when they need it? This is where option 3, **"3. Allow CashOut"** , comes in. This option allows the customers to withdraw cash from their wallet. This will be done given the following assumptions :

- Call the amount you wish to withdraw, **cashout_amount**
- Ensure the customer enters the right PIN before the transaction goes through

Write a program that allows cashout when option 3 is chosen and asks the user to enter the amount to be taken out (cashout_amount).

Test Case 5

```
>>>
Enter your choice: 3
Allow CashOut:
1. Yes
2. No
Enter your choice: 1
Enter the amount to allow for Cash Out: 100
Enter your 4-digit PIN: 7209
Cash Out successfully allowed
New balance: GHS 651.5
Do you want to perform another operation? (yes/no):
```

Part D: My Wallet

In your customers managing their finances, they have to be apprised of the amount in their virtual wallet and also be able to top up the wallet when needed. They are going to be able to do just that with option 4, given the following assumptions:

- Call the amount that is added to the current balance by user, **topUp_amount**

Write a program that allows them to top up their current balance and/or shows them their current balance when you select option 4.

Test Case 6

```
>>>
Enter your choice: 4

My Wallet:
1. Top Up Balance
2. Check Balance
```

Enter your choice: 1

Enter amount to top up your balance: 200

Balance top up is successful. New balance: GHS 1200.0

Do you want to perform another operation? (yes/no):

Test Case 7

>>>

Enter your choice: 4

My Wallet:

1. Top Up Balance

2. Check Balance

Enter your choice: 2

Your current balance is: GHS 1200.0

Do you want to perform another operation? (yes/no):