CS 115 - Introduction to Programming in Python

Tutorial 07

Lab Objectives: Inheritance

- You should only use functionality covered in CS115 in your solution.
- Include a docstring for your functions.

1. Create a class, **`BankAccount`**, with the following data attributes and methods. Note all data attributes and class variables should be private.

   Data Attributes:

   - **account number**: stores the account number of the customer.
   - **balance**: current balance of the customer

   Methods:

   - **__init()__**: initializes the `balance` to 0 and `account number` to the parameter
   - **get_account:** returns the **`account number`**.
   - **deposit:** Adds the specified amount to the account balance
   - **withdraw :** withdraws the specified amount from the account balance if there is enough money and returns True, otherwise prints a message and returns False.
   - **transfer:** transfers the given amount from this account to the given other amount.
   - **__repr()__**: returns a string representation of a `BankAccount` object formatted as shown in the sample run.

2. Create a subclass, **`SavingsAccount`**, by extending the superclass **`BankAccount,`** with the following data attributes and methods. Note all data attributes should be private.

   Data Attributes:

   - **rate**: interest rate

   Methods:

   - **__init()__:**
     - Takes the following parameters: account number, initial balance and the interest rate as parameters.
     - Initialize the `BankAccount` data using the super class `__init__` method.
     - Initialize interest rate to the parameter values.

   - **add_interest:** calculates the interest and deposits the interest to the balance.

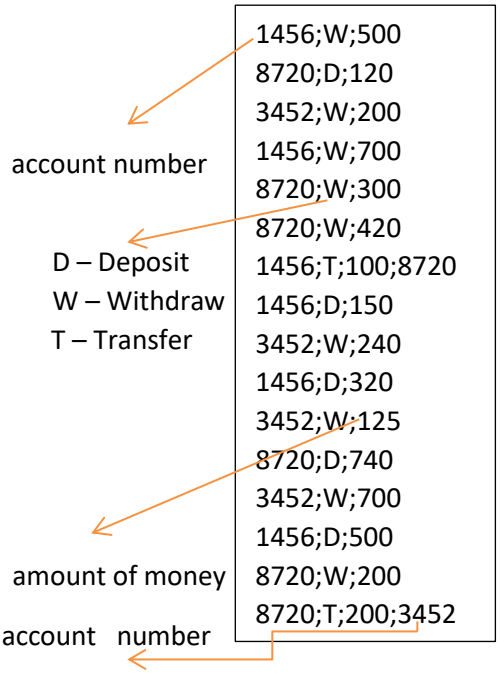
3) Write a script **`BankApp`** with the following functions:

- **readCustomers():** takes a string filename as a parameter and returns a list of **`SavingsAccounts`** objects.
- **findCustomerIndex():** takes a list of **`SavingsAccounts`** objects and an account number and returns the index of that account.
- **makeOperations():** takes a string filename as a parameter and performs the operations on the specified object.
- **The script** should do the following:
  - Creates a list of **`SavingsAccounts`** objects, **`accounts`**, by sending. `'customer.txt'` to **`readCustomers`** function.
  - Call `makeOperations` function by sending `account.txt` to make the specified operations on the specified accounts in **`accounts`**. Call **`findCustomerIndex`** function where necessary.
  - Display **`accounts`** list.

`'customer.txt'` contains:

```
1456,1000,15
8720,500,12.5
3452,2000,18
```

`'account.txt'` contains:



Sample Run:

```
There is not enough money!
700 TL cannot be withdrawn from account 1456

There is not enough money!
420 TL cannot be withdrawn from account 8720

[
Account Number: 1456
Balance: 1370 TL
,
Account Number: 8720
Balance: 760 TL
```

,
Account Number: 3452
Balance: 935 TL
]