Lab 12: Python

ESS 112 - Programming I

International Institute of Information Technology – Bangalore

Submission: LMS by 14 Feb 23:59:59

Submission Instructions: Submit your code in a separate file on LMS.

Exercises:

1. Considering the list of values, use the function filter to return only negative numbers.

   Sample Input:
   1 2 -3 5 -2

   Sample Output:
   -3 -2

2. Find the values that are common to the two lists using filter function

   Sample Input:
   1 2 -3 5 -2
   2 4 5 7 8

   Sample Output:
   2 5

3. Use map to print the square and cube of each number in a list

   Sample Input:
   1 2 3

   Sample Output:
   1 4 9
   1 8 27

4. Use filter function to print only the names that are greater than or equal to 4 letters in a list of string

   Sample Input:
   Man is on the moon

   Sample Output:
   moon

5. Write a Python program to find if a given string starts with a given character.

   Sample Input:
   man
   m

   Sample Output:
   True

6. Write a Python program to calculate the sum of the positive and negative numbers of a given list of numbers using lambda function.

   Sample Input:
   1 -2 2 3 -4

   Sample Output:
   6
   -6

7. Print product and sum of the list of number using reduce function

   Sample Input:
   1 2 3

   Sample Output:
   6
   6

8. Write a Python program to find the common elements in two lists using list Comprehension.

   Sample Input:
   1 2 3 4 5
   2 4 6

   Sample Output:
   2 4

9. Write a Python program to transport a matrix using list comprehension.

   Sample Input:
   1 2 3
   4 5 6
   7 8 9

   Sample Output:
   1 4 7
   2 5 8

3 6 9

10. (Bonus question – optional) Design python program to demonstrate a possible implementation of simplistic bank ATM. To check your balance, you enter your account number. The bank sends an OTP to your registered mobile, and if you enter the OTP correctly in the ATM, it prints out your current balance.
Your solution should follow the class structure described below:
1. ATM: contains the main and is responsible for taking the user input and sending it to the Bank, using the methods of Bank listed below
2. Bank: has a list of Customers and Accounts. It uses an OTPService to send OTP and also to verify the OTP. Checking balance for an account is thus a 2 step process: the ATM sends a request to the Bank to send an OTP to the registered mobile number associated with the customer for this account, and then the ATM sends a second request to the Bank to verify the OTP. Bank provides the following methods:

    - addCustomer(mobileNumber) returns a unique customer ID
    - addAccount(customerID,balance) returns a unique account ID. Balance is the initial balance for this account
    - sendOTPforBalanceCheck(accountNumber)
    - verifyOTPandGetBalance(accountNumber, OTP) returns the balance if the OTP is correct for this account number

    3. Customer: Each customer has a unique ID as well as a registered mobile number
    4. Account: An account consists of the Customer ID, the Account ID and the balance. Customers can have multiple accounts
    5. OTPService: sends an OTP to a requested mobile number, and also verifies if an OTP matches what was sent out for a particular request. Provides the following methods:

    - sendOTP(mobileNumber) - sends OTP (prints OTP to standard out)
    - verifyOTP(mobileNumber, OTP) - returns true if the OTP matches the outstanding OTP for this mobile number

    For simplicity, all unique numbers/IDs mentioned above are integers: Customer ID, Account ID, OTP, are each managed independently, and each is a set of auto-generated sequential numbers. Customer IDs start from 1, Account IDs start from 101, OTP's start from 1001.

    The ATM class can access only the Bank class and its methods. All other classes are hidden from it.

    OTPService encapsulates the mechanism of sending out OTP's and verifying them for a given mobile number. One implementation is for this class to maintain the list of pairs so that it can later verify the OTP. Note that for security, the combination of <mobileNumber, OTP> should not be accessible by any other class.

    The main reads the standard input and processes each line as follows, based on the 1st character of the input line:

- if the first character is C, then this is to add a new customer. The next field is an integer, the mobile number of the customer

- if the first character is A, then a new account is created. The next field the Customer ID of the customer owning this account, and the third field is the balance in the account

- if the first character is B, then a request to send OTP is to be sent to the bank. The next field is the account ID whose balance is being queried

- if the first character is V, then the OTP is to be verified. The next 2 fields are the account ID and the OTP to be verified

- if the first character is E, then that is the end of input

Sample Input

```
C 12345
C 23456
C 34567
C 45678
A 1 1000
A 2 2000
A 3 3000
A 2 4000
A 3 5000
A 3 6000
B 101
V 101 1001
B 103
B 102
V 103 1002
B 104
V 102 1003
V 104 1004
B 106
V 106 1002
V 106 1005
V 106 1005
E
```

Sample Output

```
OTP 1001 to 12345
Balance 1000
OTP 1002 to 34567
OTP 1003 to 23456
Invalid OTP
OTP 1004 to 23456
Invalid OTP
Balance 4000
```

OTP 1005 to 34567
Invalid OTP
Balance 3000
Invalid OTP