UPI Wallet - Requirement 1

Our country is going towards a cashless economy. To increase cashless transactions, the government has announed to help startups that use UPI platform to create cashless wallets. The best applications in an upcoming Hackathon will be funded. You, being an aspiring programmer, want to start a wallet platform keeping in mind the funding from the initiative. Let's create an wallet application and add features to it in each requirement.

Requirement 1:

Let's start off by creating two **User** objects and check whether they are equal.

1. Create a **User** Class with the following attributes:

Member Field Name	**Type**
_name	string
_mobileNumber	string
_email	string
_dob	DateTime

2. Mark all the attributes as private
3. Create / Generate appropriate properties
4. Add a default constructor and a parameterized constructor to take in all attributes in the given order:
   User(string _name, string _mobileNumber, string _email, DateTime _dob)
5. When the "User" object is printed, it should display the following details: **[Override the ToString method]**
   Print format:
   Name: "_name"
   MobileNumber: "_mobileNumber"
   Email: "_email"
   DOB: "_dob"
6. Two Users are considered the same if they have the same _name and _mobileNumber. Implement the logic in the appropriate function. (Case – Insensitive) **[Override the Equals method]**

   The input format consists of User details separated by a comma in the below order, (_name,_mobileNumber, _email, _dob)

The Input to your program would be details of two Users, you need to display their details as given in "5th point(refer above)" and compare the two Users and display if the Users are same or different.

Note: There is an empty line between display statements. Print the empty lines in the Main

function.
Create a class program with the Main method to implement the above features

Sample Input/Output 1:

Enter user 1 detail:
Emma,9659659790,emma@gmail.com,15/03/1993
Enter user 2 detail:
Leslie,9595878580,leslie@gmail.com,26/09/1989

User 1
Name: Emma
MobileNumber: 9659659790
Email: emma@gmail.com
Dob: 15/03/1993

User 2
Name: Leslie
MobileNumber: 9595878580
Email: leslie@gmail.com
Dob: 26/09/1989

User 1 and User 2 are different

Sample Input/Output 2:

Enter user 1 detail:
Mona,9898798652,mona@gmail.com,27/02/1987
Enter user 2 detail:
MONA,9898798652,mona@gmail.com,27/02/1987

User 1
Name: Mona
MobileNumber: 9898798652
Email: mona@gmail.com
Dob: 27/02/1987

User 2
Name: MONA
MobileNumber: 9898798652
Email: mona@gmail.com
Dob: 27/02/1987

User 1 is same as User 2

UPI - Requirement 2

Requirement 2:

In this requirement, you need to validate the email id of the users.

a)Create a class **Program** with Main method and the following static methods:

Method Name	Description
static Boolean ValidateEmailId(string email)	Validate the email id based on the rules given below. Returns **true**if email id is valid else return **false**

b) While validating email id follow the below rules. The format of the email id is given below

username@domain.TLD

where, TLD - Top Level Domain

1. The email id should start only with alphabets(either uppercase or lowercase).
2. The email username can contain alphabets(either uppercase or lowercase), numbers and the special characters (. and _).
3. The email username should not contain any special characters other than **" . "** and **" _ "**.
4. After the username special character @ should present.
5. The email domain should contain only alphabets(either uppercase or lowercase).
6. After email domain, a value dot (.) should present.
7. The email Top Level Domain should contain only alphabets(both uppercase and lowercase) and it should have only 2 to 6 characters.

Example: **alpha_Beta.01@google.com** is a valid email id.
Since the username contain only alphabets, numbers and a special character (_ and .), then the @ symbol is present. The domain name contains only alphabets and the symbol dot(.) and the Top Level Domain has only 3 characters.

Note: Print "**Email is valid**" if email is valid else print "**Email is invalid**".

All the above print statements are present in the Main method.

Sample Input and Output 1:

Enter the email to be validated:
alpha_Beta.02@mail.com
Email is valid

Sample Input and Output 2:

Enter the email to be validated:
0alpf&sk@mail.in
Email is invalid

UPI Wallet - Requirement 3

Requirement 3:

In this requirement, let's find the total amount that has been credited in a specified month.

Create a class **Transaction** with the following private attributes

Attribute Name	Datatype
_type	string
_upi	UPI
_amount	double
_date	DateTime

Create / Generate appropriate properties
Add a default constructor and a parameterized constructor to take in all attributes in the given order:

public Transaction(string _type, UPI _upi, double _amount, DateTime _date)

Create a class **UPI** with the following private attributes

Attribute Name	Datatype
_number	string
_accountNumber	string
_transactionList	List<Transaction>

Create / Generate appropriate properties
Add a default constructor and a parameterized constructor to take in all attributes in the given order:

public UPI(string _number, string _accountNumber, List<Transaction> _transactionList)

The following methods are present in the UPI class

Method Name	Description
static List<UPI> Prefill()	This method returns a List of prefilled UPI objects.
static double GetCreditedAmount(List<UPI> upiList, string month)	This method returns the total amount that has been credited in the given month.

Obtain transaction details from the user and store it in appropriate UPI objects. Finally, print the total credited amount by calling the required methods.

The Transaction details are given in comma separated format in the below order
type, upi, amount, date

Note : Use "dd/MM/yyyy" for formatting date attribute.
While printing doubledatatype, print one digit after the decimal point.

Sample Input/Output 1:

Enter the number of transactions:
5
credit,joe@okicici,5000,27/01/2018
debit,joe@okicici,1000,28/01/2018
credit,jack@okboi,4000,22/01/2018
credit,monica@okiob,6000,01/02/2018
credit,emma@oksbi,5000,03/02/2018
Enter the month:
January
The total credited amount in the month of January is 9000.0

Sample Input/Output 2:

Enter the number of transactions:
6
credit,mona@ib,20000,04/05/2018
credit,frank@okhdfc,15000,06/07/2018
debit,leslie@oksbi,1000,06/05/2018
credit,jack@okboi,14000,27/05/2018
debit,emma@oksbi,15000,13/05/2018
credit,frank@okhdfc,1500,29/03/2018
Enter the month:
May
The total credited amount in the month of May is 34000.0

UPI - Requirement 4

In this requirement, you need to sort the debit type transactions in ascending order of amount of transaction.

a) Create a class **User** with the following private attributes,

Member Field Name	Type
_name	string
_mobileNumber	string
_email	string
_dob	DateTime
_upi	UPI
_transactionList	List<Transaction>

Mark all the attributes as private,
Create / Generate appropriate properties,
Add a default constructor and a parameterized constructor to take in all attributes in the given order:
public User(string _name, string _mobileNumber, string _email, DateTime _dob, UPI _upi, List<Transaction> _transactionList)

b) Create a class **Transaction** with the following provate attributes,

Member Field Nae	Type
_type	string
_upi	UPI
_amount	double
_date	DateTime

Mark all the attributes as private,
Create / Generate appropriate properties,
Add a default constructor and a parameterized constructor to take in all attributes in the given order:

public Transaction(string _type, UPI _upi, double _amount, Date _date) The following methods are present in the UPI class

c) Create a class **UPI** with the following private attributes,

Member Field Name	Type
_number	string
_accountNumber	string
_user	User

Mark all the attributes as private,
Create / Generate appropriate properties,
Add a default constructor and a parameterized constructor to take in all attributes in the given

order:

public UPI(string _number, string _accountNumber, User _user)

d) Create the following static methods in UPI method,

Method Name	Description
List<UPI> Prefill()	This method returns a list of upi available for transaction (given in the template)
List<Transaction>FilterAndSort(List<UPI> upiList)	This method takes upi list as parameter and returns a transaction list (which are of "debit" type and sorted in ascending order of amount)

The transaction deatils are given in comma separated format,
_type,_upiNumber,_amount,_date
Date Format : **dd/MM/yyyy**

When the “Transaction” object is printed, it should display the following format
Print format:
Console.Write("{0,-10}{1,-8}{2,-10}{3}\n","UserName","Type","Amount","Date");
The amount should be printed with 1 decimal digits.

Sample Input and Output:

Enter the number of transactions:
6
credit,mona@ib,20000,04/05/2018
credit,frank@okhdfc,15000,06/07/2018
debit,leslie@oksbi,1000,06/05/2018
credit,jack@okboi,14000,27/05/2018
debit,emma@oksbi,15000,13/05/2018
credit,frank@okhdfc,1500,29/03/2018
UserName Type Amount Date
Leslie debit 1000.0 06/05/2018
Emma debit 15000.0 13/05/2018

UPI Wallet - Requirement 5

Requirement 5:

In this requirement, let's find the total count of transactions per day.

a) Create a class **User** with the following private attributes,

Member Field Name	Type
_name	string
_mobileNumber	string
_email	string
_dob	DateTime
_upi	UPI
_transactionList	List<Transaction>

Mark all the attributes as private,
Create / Generate appropriate properties,
Add a default constructor and a parameterized constructor to take in all attributes in the given order:
public User(string _name, string _mobileNumber, string _email, DateTime _dob, UPI _upi, List<Transaction> _transactionList)

b) Create a class **Transaction** with the following provate attributes,

Member Field Nae	Type
_type	string
_upi	UPI
_amount	double
_date	DateTime

Mark all the attributes as private,
Create / Generate appropriate properties,
Add a default constructor and a parameterized constructor to take in all attributes in the given order:

public Transaction(string _type, UPI _upi, double _amount, Date _date) The following methods are present in the UPI class

c) Create a class **UPI** with the following private attributes,

Member Field Name	Type
_number	string
_accountNumber	string
_user	User

Mark all the attributes as private,
Create / Generate appropriate properties,

Add a default constructor and a parameterized constructor to take in all attributes in the given order:

public UPI(string _number, string _accountNumber, User _user)

The following methods are present in the UPI class

Method Name	Description
static List<UPI> Prefill()	This method returns a List of prefilled UPI objects.
static SortedDictionary<DateTime,int> GetPerDateTransaction(List<UPI> upiList)	This method returns aSortedDictionary that has date as key and number of transactions in that date as value.

Obtain transaction details from the user and store it in appropriate User objects. Finally, print the number of transactions in each day in forward chronological order.
The Transaction details are given in comma separated format in the below order
_type, _upi, _amount, _date

Note : Use "dd/MM/yyyy" for formatting date property.
Use **Console.Write("{0,-15} {1}\n","Date","Number of Transactions");** while printing the details
Use **SortedDictionary** to store the per day count.

Sample Input/Output 1:

Enter the number of transactions:
6
credit,mona@ib,20000,20/07/2018
credit,frank@okhdfc,15000,21/07/2018
debit,leslie@oksbi,1000,21/07/2018
credit,jack@okboi,14000,22/07/2018
credit,frank@okhdfc,4500,15/07/2018
debit,emma@oksbi,15000,13/07/2018

Date	Number of Transactions
13/07/2018	1
15/07/2018	1
20/07/2018	1
21/07/2018	2
22/07/2018	1

Sample Input/Output 2:

Enter the number of transactions:

7
credit,monica@okiob,6000,01/02/2018
credit,emma@oksbi,5000,01/02/2018
debit,mona@ib,1500,01/02/2018
credit,emma@oksbi,1000,02/02/2018
debit,leslie@oksbi,1000,05/02/2018
credit,frank@okhdfc,1500,05/02/2018
debit,leslie@oksbi,1000,02/02/2018

Date	Number of Transactions
01/02/2018	3
02/02/2018	2
05/02/2018	2

UPI - Requirement 6

In this requirement, you need to find the user to whom highest amount is debited.

a) Create a class **User** with the following private attributes,

Member Field Name	Type
_name	string
_mobileNumber	string
_email	string
_dob	DateTime
_upi	UPI
_transactionList	List<Transaction>

Mark all the attributes as private,
Create / Generate appropriate properties,
Add a default constructor and a parameterized constructor to take in all attributes in the given order:
public User(string _name, string _mobileNumber, string _email, DateTime _dob, UPI _upi, List<Transaction> _transactionList)

b) Create a class **UPI** with the following private attributes,

Member Field Name	Type
_number	string
_accountNumber	string
_user	User

Mark all the attributes as private,
Create / Generate appropriate properties,
Add a default constructor and a parameterized constructor to take in all attributes in the given order:
public UPI(string _number, string _accountNumber, User _user)

c) Create a class **Transaction** with the following provate attributes,

Member Field Nae	Type
_type	string
_upi	UPI
_amount	double
_date	DateTime

Mark all the attributes as private,
Create / Generate appropriate properties,
Add a default constructor and a parameterized constructor to take in all attributes in the given order:
public Transaction(string _type, UPI _upi, double _amount, Date _date)

d) Create the following static methods in UPI method,

Method Name	Description
List<UPI> Prefill()	This method returns a list of upi available for transaction (given in the template)
List<Transaction> FavouriteUPI(List<UPI> upiList)	This method takes upi list as parameter and returns a upi object for whom highest amount is debited

The transaction deatils are given in comma separated format,
_type,_upiNumber,_amount,_date
Date Format: **dd/MM/yyyy**

Sample Input and Output:

Enter the number of transactions:
5
debit,joe@okicici,5000,27/01/2018
debit,joe@okicici,1000,28/01/2018
debit,jack@okboi,4000,22/01/2018
debit,monica@okiob,5500,01/02/2018
debit,emma@oksbi,5001,03/02/2018
Maximum amount is transferred to Joe