In-Class Activity # 3

This in-class activity is worth **10% of your final grade**.

You can work in teams of **two**. If you work in team **only one of the team members must submit the activity on Omnivox**.

With this instruction file you were given two **.py** files which you have to fill in according to the instructions on next page.

Write your names and student ids at the top of the the tests.py file.

You have to submit **two files** in a **zip file**.

1. **tests.py** filled in;
2. **pytransact.py** filled in.

You **do not have to submit this file**, answers questions by filling in the Python files.

Do not change the name of the Python files.
Do not submit each file individually.
Only the last submission will be graded.

Context

You are working for a company which develops an e-commerce platform. One of the project of the company is a Python library called **pytransact** which provides various functions meant to help handle users credit cards.

You have been charged to test and implement three functions in **pytransact**.

Read the features below, then read the instructions at the end of the document.

Features

Here are the expected behaviour of each function you have to test and develop.

is_mastercard(cc)

This function takes a credit card number (string) as argument and returns a boolean. It returns **True** if it is a valid MasterCard number and **False** otherwise.

MasterCard credit card numbers have the following features:

- It is composed only of digits;
- It must contain 16 characters;
- It must start with a 5;
- The second digit must be a digit from 1 to 5

Any credit card number which does not comply with all of the above is not a valid MasterCard number.

is_valid_expiration(date)

This function takes an expiration date (string) and validates whether is it valid or not. It returns **True** if it is valid and **False** otherwise.

An expiration date takes the format *mm/yy*, *mm-yy* or *mm yy* where the two first characters are the month of expiration and the two last characters are the year of expiration. By example *02/23* means February 2023.

A valid expiration date has the following features

- The month and year must be exactly two characters each;
- The month and year must be separated by a slash (/), a dash (-) or a space, only those three characters should be accepted as separators;
- The month must be from 01 to 12;
- The year must be from 00 to 99.

Any expiration date which does not come in this exact format is invalid.

random_mastercard()

Generating random credit card number can be useful for many reasons. By example, they can be used for testing purpose or for security by hiding real credit card numbers among fake ones.

This function takes no argument and must return a random, valid MasterCard credit card number.

The randomly generated number must be valid according to the definition given in the **is_mastercard** function documentation.

Instructions

Task #1 (20 points)

In the **tests.py** file, write a unit test for the **pytransact.is_mastercard** function. To get full marks, your choice of input and expected outputs must test all the features described in the previous section.

Hint: the more input-expected output you write, the better.

If you find a bug while testing the **pytransact.is_mastercard** function, fix it.

Task #2 (20 points)

In the **tests.py** file, write a unit test for the **pytransact.is_valid_expiration** function. To get full marks, your choice of input and expected outputs must test all the features described in the previous section.

Hint: the more input-expected output you write, the better.

Task #3 (20 points)

Once your wrote the test in Task 2, implement the **is_valid_expiration** function in the **pytransact.py** file.

Task #4 (20 points)

In the **tests.py** file, write a unit test for the **pytransact.random_mastercard** function. Since this is a random function, there is no expected output. Your test must assert that all the generated numbers are valid MasterCard numbers. It should also assert that the function does not always generate the same number, otherwise the function is not valid.

Hint: now that it is tested, you can use **pytransact.is_mastercard** in this test to help yourself.

Task #5 (20 points)

Once your wrote the test in Task 5, implement the **random_mastercard** function in the **pytransact.py** file.