**Homework 4**

**Task 1**

1) Git and GitHub

Define the following Key Git and GitHub terminology:

GIT WORKFLOW FUNDAMENTALS

* Working directory- The working tree, or working directory, consists of files that you are currently working on. You can think of a working tree as a file system where you can view and modify files.
* Staging area- The staging area is like a rough draft space, it's where you can git add the version of a file or multiple files that you want to save in your next commit (in other words in the next version of your project). Note that you can copy versions of files into the staging area and also take them out of the staging area before you make your commit which is why I referred to it as a rough draft space.
* Local repo (head)- The local repository is a Git repository that is stored on your computer.
* Remote repo (master)- The remote repository is a Git repository that is stored on some remote computer. The remote repository is usually used by teams as a central repository into which everyone pushes the changes from his local repository and from which everyone pulls changes to his local repository.

WORKING DIRECTORY STATES

* Staged- A staging step in git allows you to continue making changes to the working directory, and when you decide you want to interact with version control, it allows you to record changes in small commits. Staged means that you have marked a modified file in its current version to go into your next commit snapshot.
* Modified- Modified means that you have changed the file but have not committed it to your database yet.
* Committed- Committed means that the data is safely stored in your local database.

GIT COMMANDS

* Git add- The git add command adds a change in the working directory to the staging area. It tells Git that you want to include updates to a particular file in the next commit. However, git add doesn't really affect the repository in any significant way—changes are not actually recorded until you run git commit. In conjunction with these commands, you'll also need git status to view the state of the working directory and the staging area.
* Git commit- The "commit" command is used to save your changes to the local repository. Note that you have to explicitly tell Git which changes you want to include in a commit before running the "git commit" command. This means that a file won't be automatically included in the next commit just because it was changed.
* Git push- The git push command is used to upload local repository content to a remote repository. Pushing is how you transfer commits from your local repository to a remote repo.
* Git fetch- The git fetch command downloads commits, files, and refs from a remote repository into your local repo. Fetching is what you do when you want to see what everybody else has been working on. Git isolates fetched content from existing local content; it has absolutely no effect on your local development work.
* Git merge- Merging is Git's way of putting a forked history back together again. The git merge command lets you take the independent lines of development created by git branch and integrate them into a single branch.
* Git pull- The git pull command is used to fetch and download content from a remote repository and immediately update the local repository to match that content.

**TASK-2 ( Exception handling )**

Question 1-Simple ATM program

Tasks:

i)

user = {

'pin': 1114,

'balance': 100

}

def withdraw\_cash():

while True:

try:

amount = int(input("Enter the amount you want to withdraw: "))

except ValueError:

print("Enter valid amount")

else:

if amount > user['balance']:

raise ValueError("You don't have enough balance to make this withdrawal")

is\_quit = True

else:

user['balance'] = user['balance'] - amount

print(f"£{amount} successful withdraw your remaining balance is £{user['balance']}")

print('')

return "Thank you for using Python ATM"

finally:

print("Program executed")

def main():

count = 0

while count < 3:

try:

pin = int(input('Please enter your pin: '))

except ValueError:

print("Please enter correct pin")

count += 1

else:

if pin != user['pin']:

print("Pin does not match.. Try Again")

count += 1

else:

withdraw\_cash()

if count == 3:

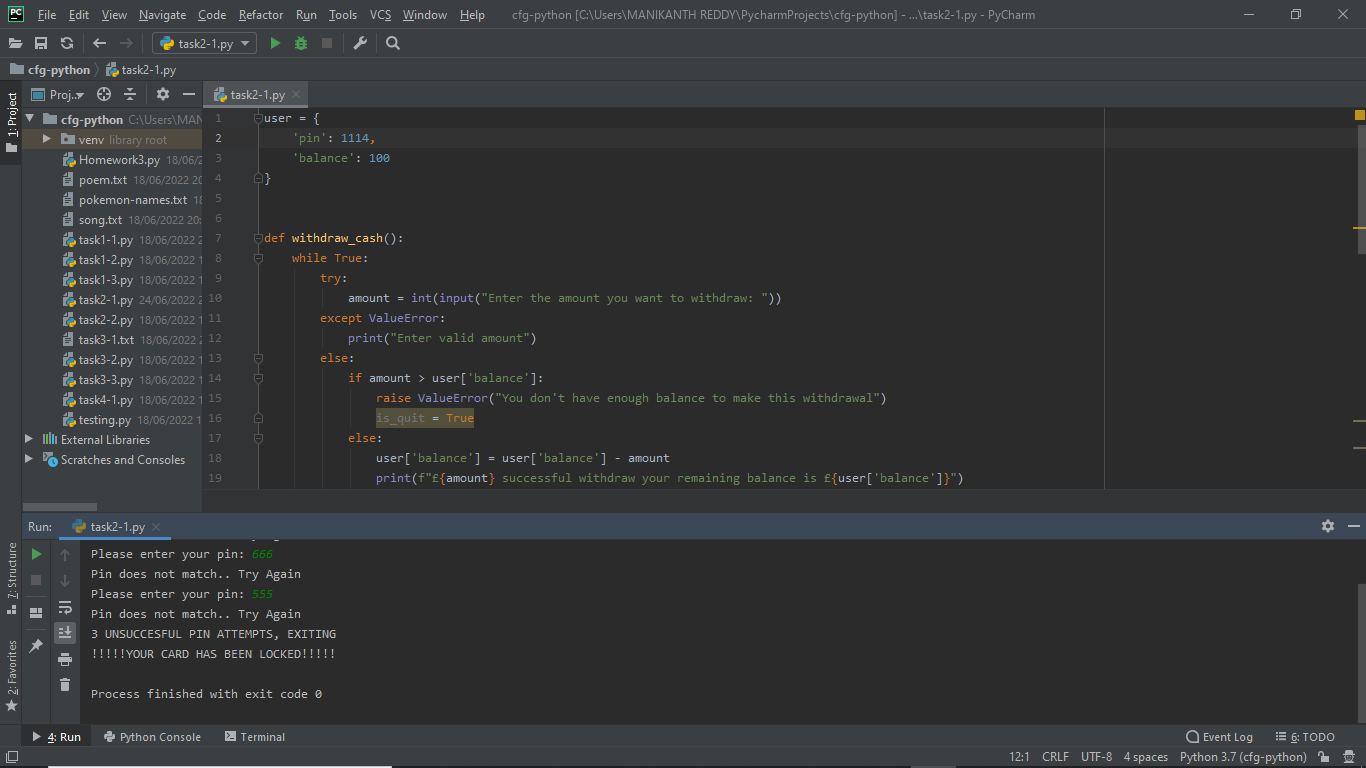
print('3 UNSUCCESFUL PIN ATTEMPTS, EXITING')

print('!!!!!YOUR CARD HAS BEEN LOCKED!!!!!')

is\_quit = True

return "Thank you for using Python ATM"

main()



**Task- 3**

from atm\_unit\_test import maketransaction

from unittest import TestCase

class TestATM\_Withdrawal(TestCase):

def test\_rightPin\_rightAmount(self):

expected = 10

result = maketransaction(amount=20, user['pin']=1042)

self.assertEqual(expected, result)

def test\_rightPin\_aboveAmount(self):

expected = 'You dont have enough balance to make this withdrawal'

result = maketransaction(amount=1000, user['pin']=1114)

self.assertEqual(expected, result)

def test\_rightPin\_negativeAmount(self):

expected = 70

result = maketransaction(amount=-80, user['pin']=1000)

self.assertEqual(expected, result)

def test\_shortPin\_rightAmount(self):

expected = 'Please enter your pin:'

result = maketransaction(amount=70, user['pin']=1117)

self.assertEqual(expected, result)

def test\_wrongPin\_rightAmount(self):

expected = 'Please enter corret pin'

result = maketransaction(amount=70, user['pin']=1114)

self.assertEqual(expected, result)