PRT-452-Assingment1-TDD

Name: Wei Liu

ID: s317932

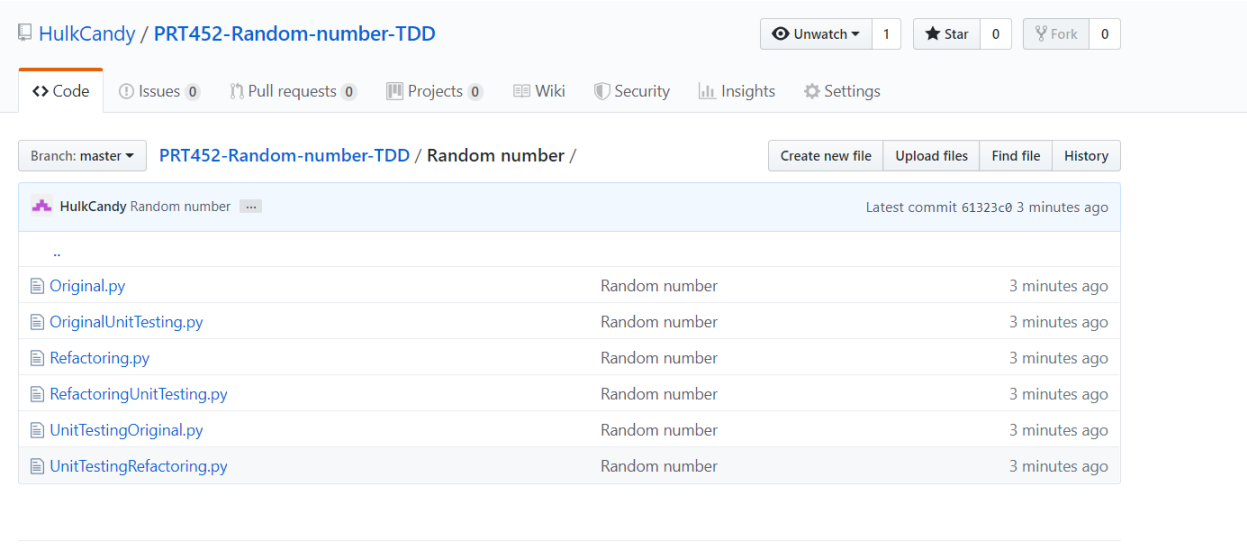
This document is the process of assignment1 which is about TDD of Random Number.

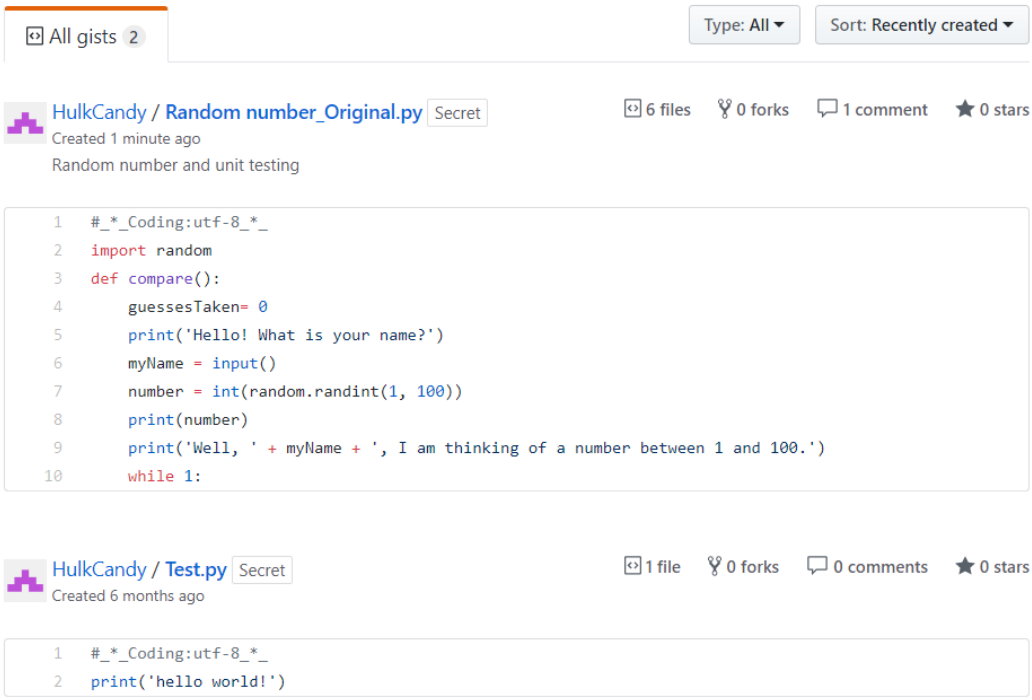
In my assignment, there are six versions of codes. Original.py, OriginalUnitTesting.py, UnitTestingOriginal.py, Refactoring.py, RefactoringUnitTesting.py, UnitTestingRefactoring.py.

Github:

https://github.com/HulkCandy/PRT452-Random-number-TDD.git

This program is developed based on Python language, I have used loops (while) and comparison to finish this program. There are multiple (while) in this program, the one is endless loop once 'input = random number', another is when this situation happened, the value of interrupter will be changed to others then loop will stop. Output the guess times. However as the arbitrary input, this program has done many regulations like '@#$%' alphabets except 'Q' in refactoring.py.





Original.py states that the original program does not have any regulation such as: signals, letters, and other conditions. Basically, it can implement the basic functions, press 'Q' to quit, guess the number and show the guessing times.

Code as below:

```
#_*_Coding:utf-8_*_
import random
def compare():
    guessesTaken= 0
    print('Hello! What is your name?')
    myName = input()
    number = int(random.randint(1, 100))
    print(number)
    print('Well, ' + myName + ', I am thinking of a number
between 1 and 100.')
    while 1:
        print('Take a guess.')
        guess = input()
        guessesTaken = guessesTaken + 1
        if guess =='q':
            print('u quit')
            quit(0);
        if int(guess) < number:
            print('Your guess is too low.')
        if int(guess) > number:
            print('Your guess is too high.')
        if int(guess) ==int(number):
```

```
            guess=int(guess)
            break
    if guess == number:
        guessesTaken = str(guessesTaken)
        print('Good job, ' + myName + '! You guessed my number
in ' + guessesTaken + ' guesses!')

compare()
```

Results:

```
C:\Users\lvv49\AppData\Local\Programs\Python\Python37\python.exe "C:/Users/lv
Hello! What is your name?
Andy
45
Well, Andy, I am thinking of a number between 1 and 100.
Take a guess.
44
Your guess is too low.
Take a guess.
46
Your guess is too high.
Take a guess.
47
Your guess is too high.
Take a guess.
45
Good job, Andy! You guessed my number in 4 guesses!

Process finished with exit code 0
```

```
2 × refactor × Original × Rerun Failed Tests ×
C:\Users\lvv49\AppData\Local\Programs\Python\Python37\python.exe "C:/Users/lvv
Hello! What is your name?
Andy
3
Well, Andy, I am thinking of a number between 1 and 100.
Take a guess.
q
u quit

Process finished with exit code 0
```

The OriginalUnitTesting.py is the code prepared to do unit testing. The mechanism is based on the “return xx”, taking advantage of the number of the program returning to carry out unit testing.

Code as below:

```
#_*_Coding:utf-8_*_
import random
def compare():
    guessesTaken= 0
    print('Hello! What is your name?')
    myName = input()
    number = int(random.randint(1, 100))
    print(number)
    print('Well, ' + myName + ', I am thinking of a number
between 1 and 100.')
    while 1:
        print('Take a guess.')
        guess = input()
        guessesTaken = guessesTaken + 1
        if guess =='q':
            print('u quit')
            return 0
            quit(0);
        if int(guess) < number:
            print('Your guess is too low.')
            return 1
        if int(guess) > number:
            print('Your guess is too high.')
            return 2
        if int(guess) ==int(number):
            guess=int(guess)
            return 3
            break
    if guess == number:
        guessesTaken = str(guessesTaken)
        print('Good job, ' + myName + '! You guessed my number
in ' + guessesTaken + ' guesses!')
```

UnitTestingOriginal.py is the unit testing code as below:

```
#_*_Coding:utf-8_*_
import unittest
from OriginalUnitTesting import compare

class TestRandom(unittest.TestCase):
    def test_guess1(self):          #testing quit
        s=compare()
        self.assertEqual(s,0)
    def test_guess2(self):          #testing less
        s = compare()
        self.assertEqual(s, 1)
    def test_guess3(self):           #testing greater
```

```
        s = compare()
        self.assertEqual(s, 2)
    def test_guess4(self):          # testing same
        s = compare()
        self.assertEqual(s, 3)



if __name__ == '__main__':
    unittest.main()
```

Testing the functions of comparison between input number and random number, for example, when it returns 0, which means 'quit' is working; when it returns 1, which means less; when it returns 2, which means greater; when I returns 3, which means both number is same.

Results:

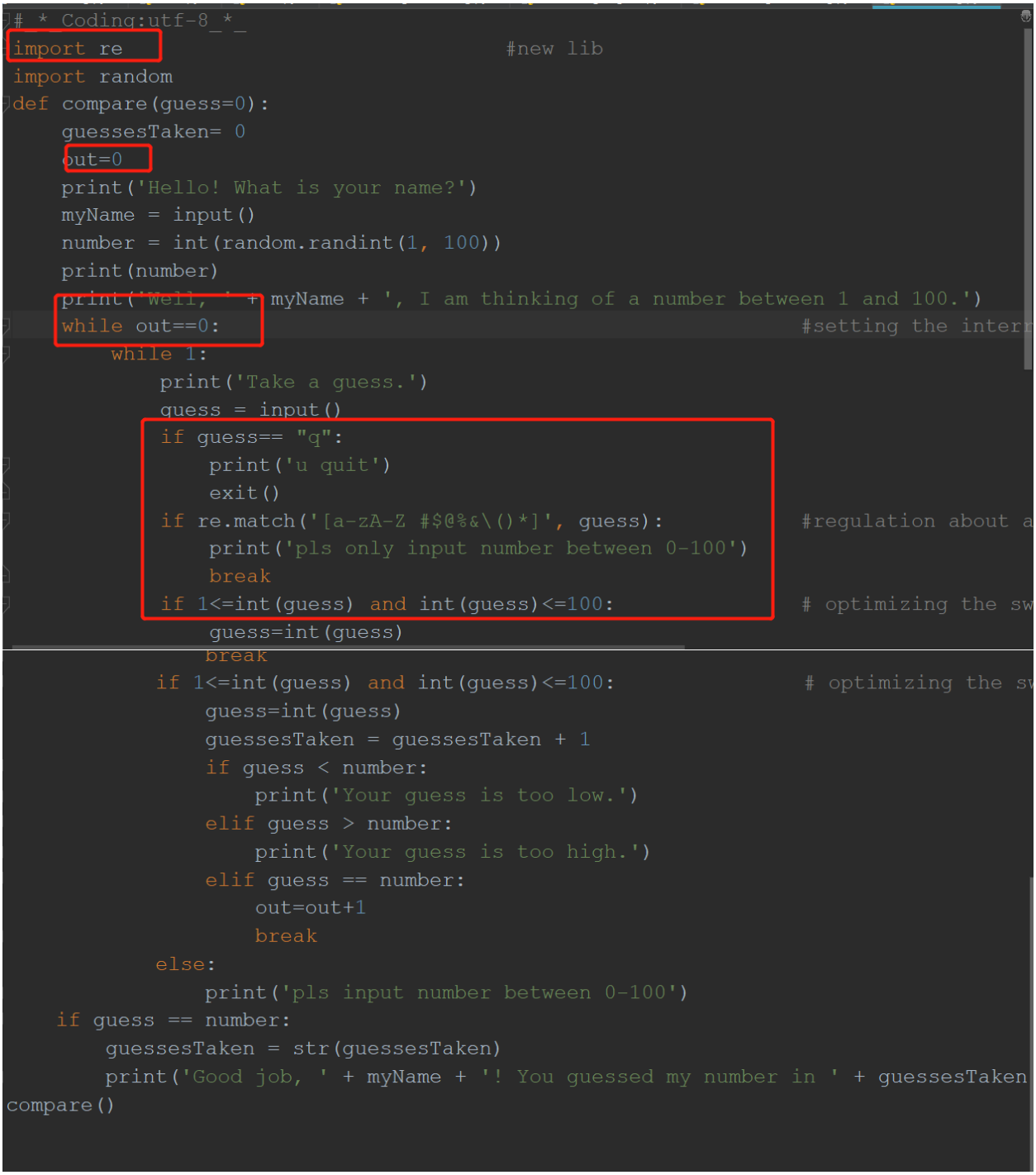
```
Testing started at 8:54 PM ...
C:\Users\lvv49\AppData\Local\Programs\Python\Python37\python.exe "C:\Program Files\JetBrains\PyCha
Launching unittests with arguments python -m unittest C:/Users/lvv49/untitled/Random number/UnitTe
Ran 4 tests in 21.796s

OK
Hello! What is your name?
96
Well, 22, I am thinking of a number between 1 and 100.
Take a guess.
u quit
Hello! What is your name?
12
Well, 22, I am thinking of a number between 1 and 100.
Take a guess.
Your guess is too low.
Hello! What is your name?
40
Well, 12, I am thinking of a number between 1 and 100.
Take a guess.
Your guess is too high.
Hello! What is your name?
87
```

Refactoring.py as below:

```
#_*_Coding:utf-8_*_
import re                               #new lib
import random
def compare(guess=0):
    guessesTaken= 0
    out=0
    print('Hello! What is your name?')
    myName = input()
    number = int(random.randint(1, 100))
    print(number)
```

```
    print('Well, ' + myName + ', I am thinking of a number
between 1 and 100.')
    while out==0:
#setting the interrupter to prevent circulating forever
        while 1:
            print('Take a guess.')
            guess = input()
            if guess== "q":
                print('u quit')
                exit()
            if re.match('[a-zA-Z #$@%&\()*]', guess):
#regulation about arbitrary input letters and signals
                print('pls only input number between 0-100')
                break
            if 1<=int(guess) and int(guess)<=100:
# optimizing the switch of 'guess' type
                guess=int(guess)
                guessesTaken = guessesTaken + 1
                if guess < number:
                    print('Your guess is too low.')
                elif guess > number:
                    print('Your guess is too high.')
                elif guess == number:
                    out=out+1
                    break
            else:
                print('pls input number between 0-100')
    if guess == number:
        guessesTaken = str(guessesTaken)
        print('Good job, ' + myName + '! You guessed my number
in ' + guessesTaken + ' guesses!')
compare()
```



```
# * Coding:utf-8 *_
import re                                   #new lib
import random
def compare(guess=0):
    guessesTaken= 0
    out=0
    print('Hello! What is your name?')
    myName = input()
    number = int(random.randint(1, 100))
    print(number)
    print('Well, ' + myName + ', I am thinking of a number between 1 and 100.')
    while out==0:                                                    #setting the interr
        while 1:
            print('Take a guess.')
            guess = input()
            if guess== "q":
                print('u quit')
                exit()
            if re.match('[a-zA-Z #$@%&\()*]', guess):                #regulation about a
                print('pls only input number between 0-100')
                break
            if 1<=int(guess) and int(guess)<=100:                    # optimizing the sw
                guess=int(guess)
                break
            if 1<=int(guess) and int(guess)<=100:                    # optimizing the sw
                guess=int(guess)
                guessesTaken = guessesTaken + 1
                if guess < number:
                    print('Your guess is too low.')
                elif guess > number:
                    print('Your guess is too high.')
                elif guess == number:
                    out=out+1
                    break
            else:
                print('pls input number between 0-100')
    if guess == number:
        guessesTaken = str(guessesTaken)
        print('Good job, ' + myName + '! You guessed my number in ' + guessesTaken
compare()
```

In this program, I have used some effective ways to make it work smoothly. Such as: regulations of input to restrict some unique letters and numbers appearing, using library “import re ” to control particular input, optimizing the process of ‘guess’ type between int and str, using the interrupter to break the loop when it matched same number.

The results:

```
C:\Users\lvv49\AppData\Local\Programs\Python\Python37\python.exe "C:/Users/lvv49/untitled/Random number/
Hello! What is your name?
Andy
71
Well, Andy, I am thinking of a number between 1 and 100.
Take a guess.
22
Your guess is too low.
Take a guess.
5
Your guess is too low.
Take a guess.
@
pls only input number between 0-100
Take a guess.
$
pls only input number between 0-100
Take a guess.
ss
pls only input number between 0-100
Take a guess.
71
Good job, Andy! You guessed my number in 3 guesses!
```

```
Run:  2 ×  refactor ×  Refactoring ×  Rerun Failed Tests ×
C:\Users\lvv49\AppData\Local\Programs\Python\Python37\python.exe "C:/Users/lvv49/untitled/Random
Hello! What is your name?
Andy
34
Well, Andy, I am thinking of a number between 1 and 100.
Take a guess.
q
u quit

Process finished with exit code 0
```

RefactoringUnitTesting.py as below

```
_*_Coding:utf-8_*_
import random
import re
def compare(guess=0):
    guessesTaken= 0
    out=0
    print('Hello! What is your name?')
    myName = input()
    number = int(random.randint(1, 100))
    print(number)
    print('Well, ' + myName + ', I am thinking of a number
between 1 and 100.')
    while out==0:
        while 1:
            print('Take a guess.') # There are four spaces in
front of print.
            guess = input()
            if guess== "q":
```

```
                return 4
                exit(0)
            if re.match('[a-zA-Z #$@%&\()*]', guess):
#filter alphbate
                print('pls only input number ssss between 0-
100')
                return 0
                break
            if 1<=int(guess) and int(guess)<=100:
                guess=int(guess)
                guessesTaken = guessesTaken + 1
                if guess < number:
                    print('Your guess is too low.')
                    return 1
                elif guess > number:
                    print('Your guess is too high.')
                    return 2
                elif guess == number:
                    out=out+1
                    return 3
                    break
            else:
                print('pls input number between 0-100')
    if guess == number:
        guessesTaken = str(guessesTaken)
        print('Good job, ' + myName + '! You guessed my number
in ' + guessesTaken + ' guesses!')
```

UnitTestingRefactoring.py. as below:

```
#_*_Coding:utf-8_*_
import unittest
from RefactoringUnitTesting import compare

class TestRandom(unittest.TestCase):
    def test_guess1(self):          #testing less
        s=compare()
        self.assertEqual(s,1)
    def test_guess2(self):          #testing greater
        s = compare()
        self.assertEqual(s, 2)
    def test_guess3(self):           #testing same
        s = compare()
        self.assertEqual(s, 3)
    def test_guess4(self):           # testing quit
        s = compare()
        self.assertEqual(s, 4)
```

```
    def test_guess5(self):           # testing signal
        s = compare()
        self.assertEqual(s, 0)


if __name__ == '__main__':
    unittest.main()
```

This unit test is based on originalUnitTest, which added one more signal test case.

Testing results:

```
Well, 77, I am thinking of a number between 1 and 100.
Take a guess.
Hello! What is your name?
92
Well, 22, I am thinking of a number between 1 and 100.
Take a guess.
Hello! What is your name?
74
Well, 22, I am thinking of a number between 1 and 100.
Take a guess.


Ran 5 tests in 30.796s

OK
pls only input number ssss between 0-100

Process finished with exit code 0
```