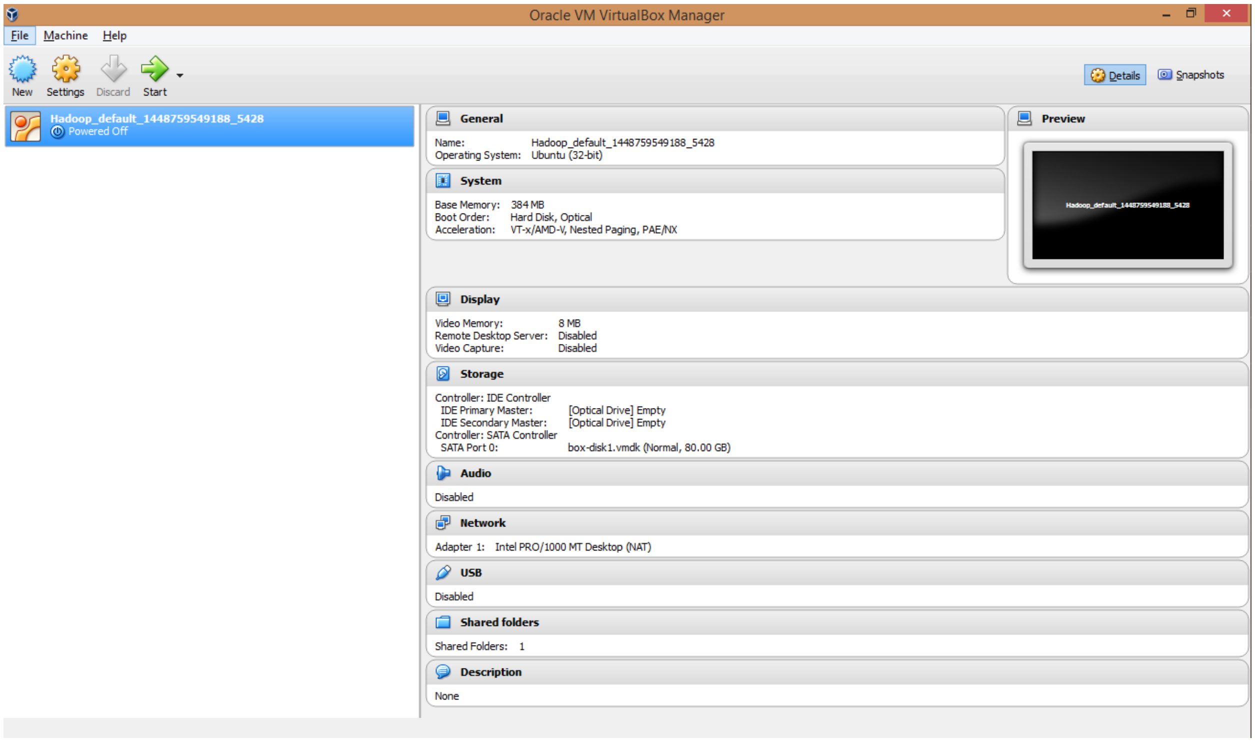
SOFTWARE TESTING METHODOLOGIES

Homework YARN

a) Getting a remote computer running

Vagrant computer -


Oracle VM VirtualBox Manager
File
Machine
Help
New
Settings
Discard
Start
Details
Snapshots
Preview
Hadoop_default_1448759549188_5428

Hadoop_default_1448759549188_5428
Powered Off

General

Name: Hadoop_default_1448759549188_5428
Operating System: Ubuntu (32-bit)

System

Base Memory: 384 MB
Boot Order: Hard Disk, Optical
Acceleration: VT-x/AMD-V, Nested Paging, PAE/NX

Display

Video Memory: 8 MB
Remote Desktop Server: Disabled
Video Capture: Disabled

Storage

Controller: IDE Controller
 IDE Primary Master: [Optical Drive] Empty
 IDE Secondary Master: [Optical Drive] Empty
Controller: SATA Controller
 SATA Port 0: box-disk1.vmdk (Normal, 80.00 GB)

Audio

Disabled

Network

Adapter 1: Intel PRO/1000 MT Desktop (NAT)

USB

Disabled

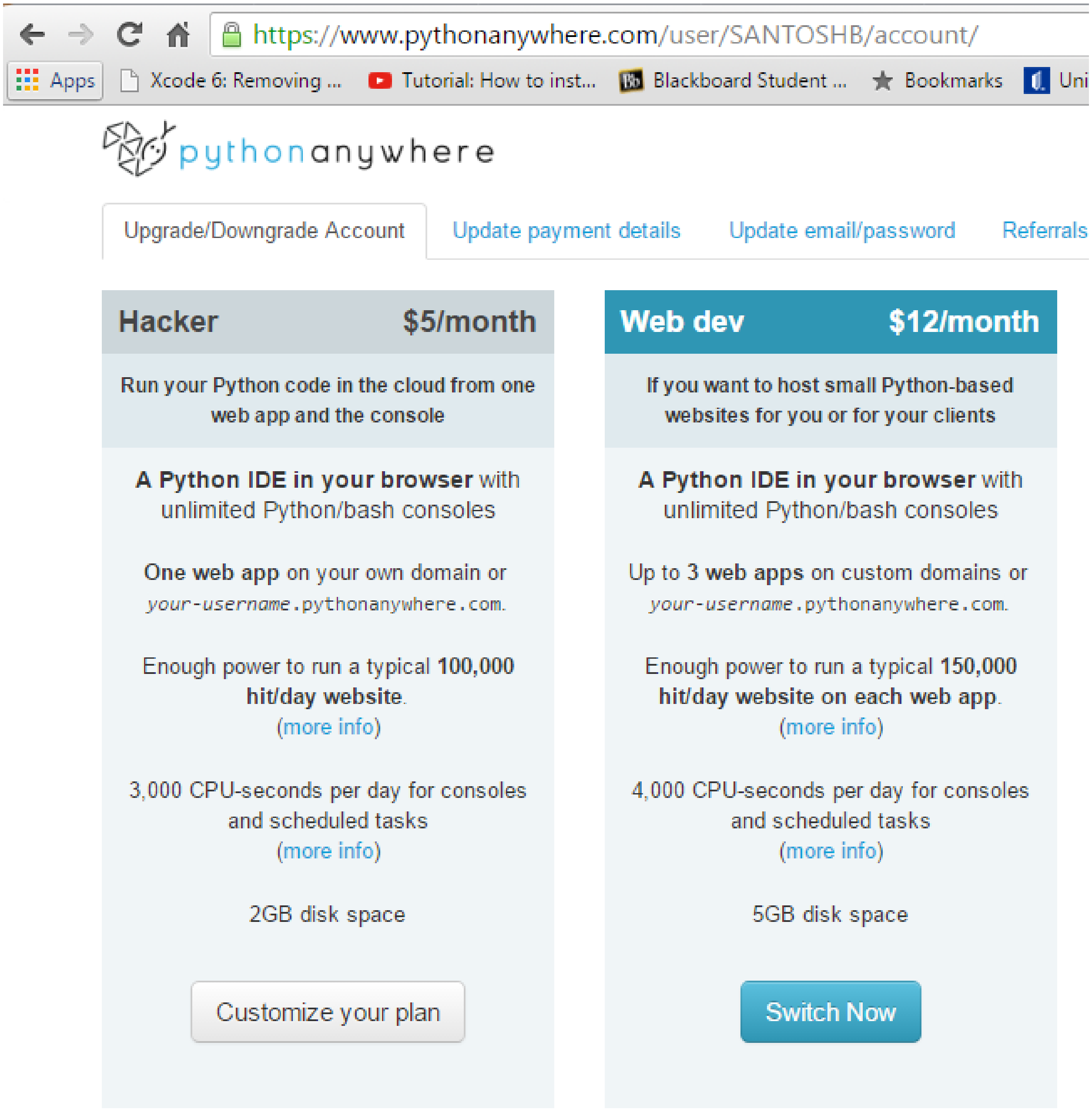
Shared folders

Shared Folders: 1

Description

None

Registered for a Hacker account In PythonAnyWhere.com from which I will be getting SSH access.



Hacker **$5/month**

Run your Python code in the cloud from one web app and the console

A Python IDE in your browser with unlimited Python/bash consoles

One web app on your own domain or *your-username*.pythonanywhere.com.

Enough power to run a typical **100,000 hit/day website**.
(more info)

3,000 CPU-seconds per day for consoles and scheduled tasks
(more info)

2GB disk space

Web dev **$12/month**

If you want to host small Python-based websites for you or for your clients

A Python IDE in your browser with unlimited Python/bash consoles

Up to **3 web apps** on custom domains or *your-username*.pythonanywhere.com.

Enough power to run a typical **150,000 hit/day website on each web app**.
(more info)

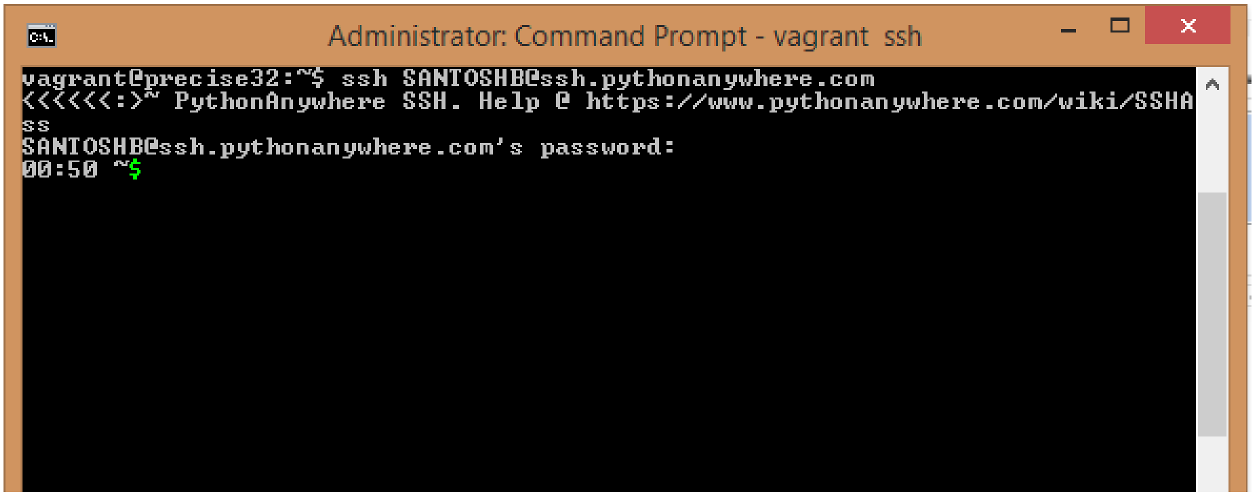
4,000 CPU-seconds per day for consoles and scheduled tasks
(more info)

5GB disk space

My ssh can be accessed through any command line in which ssh is installed properly with the following address of mine of PythonAnyWhere account

$ ssh SANTOSHB@ssh.pythonanywhere.com

Password will be the my PythonAnyWhere.com account in order to access the ssh through Vagrant

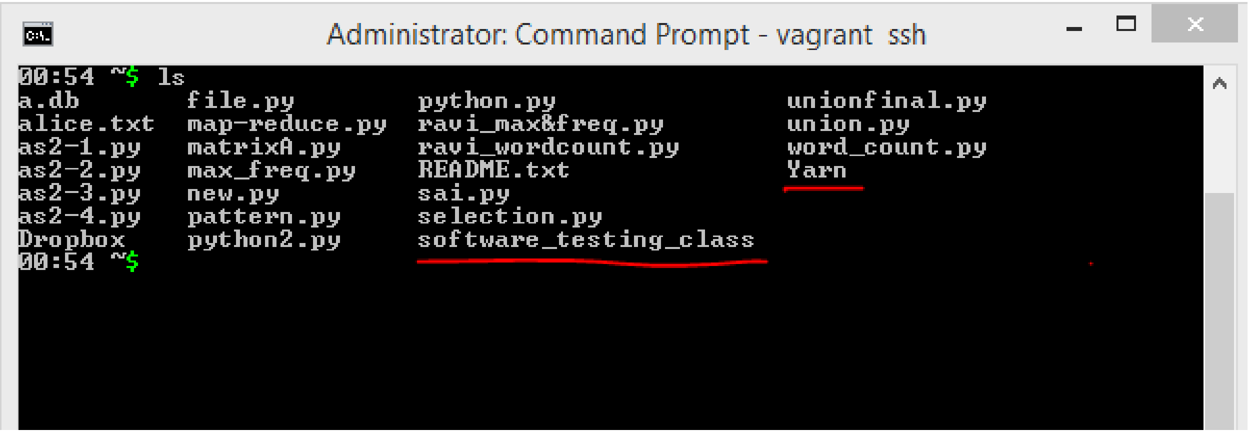


```
vagrant@precise32:~$ ssh SANTOSHB@ssh.pythonanywhere.com
<<<<<<:>~ PythonAnywhere SSH. Help @ https://www.pythonanywhere.com/wiki/SSHA
ss
SANTOSHB@ssh.pythonanywhere.com's password:
00:50 ~$
```

We can observe that my ssh of PythonAnyWhere.com has been successfully getting accessed through the vagrant box.

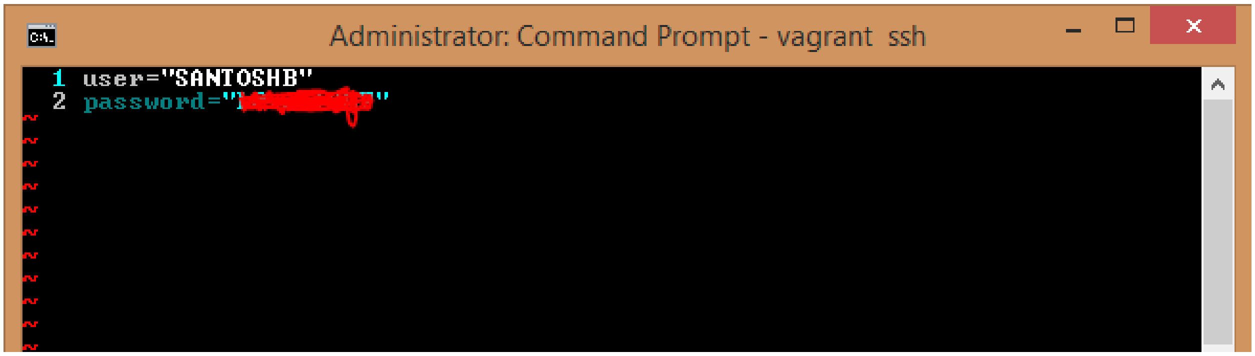
I've cloned git directories of https://github.com/gregdelozier/software_testing_class and https://github.com/Python-Yarn/Yarn/

We can observe the 2 directories in the below screenshot which is showing up 'Yarn' and 'software_testing_class'



```
00:54 ~$ ls
a.db        file.py        python.py              unionfinal.py
alice.txt   map-reduce.py  ravi_max&freq.py       union.py
as2-1.py    matrixA.py     ravi_wordcount.py      word_count.py
as2-2.py    max_freq.py    README.txt             Yarn
as2-3.py    new.py         sai.py
as2-4.py    pattern.py     selection.py
Dropbox     python2.py     software_testing_class
00:54 ~$
```

Editing the ***private.py*** in directory “*software_testing_class >> lecture_9 >> private.py*” with the user-name and password of my PythonAnyWhere account



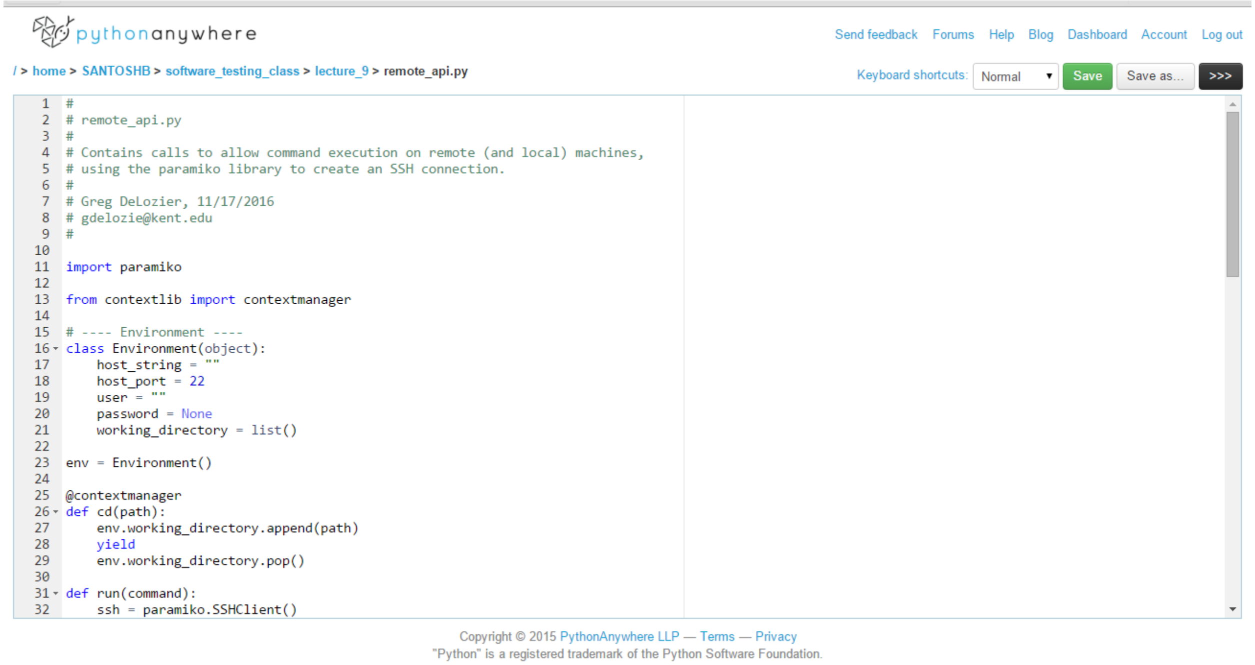
```
user="SANTOSHB"
password="[redacted]"
```

REMOTE_API.PY

Running remote_api.py by making few changes, such that all the test cases will pass.

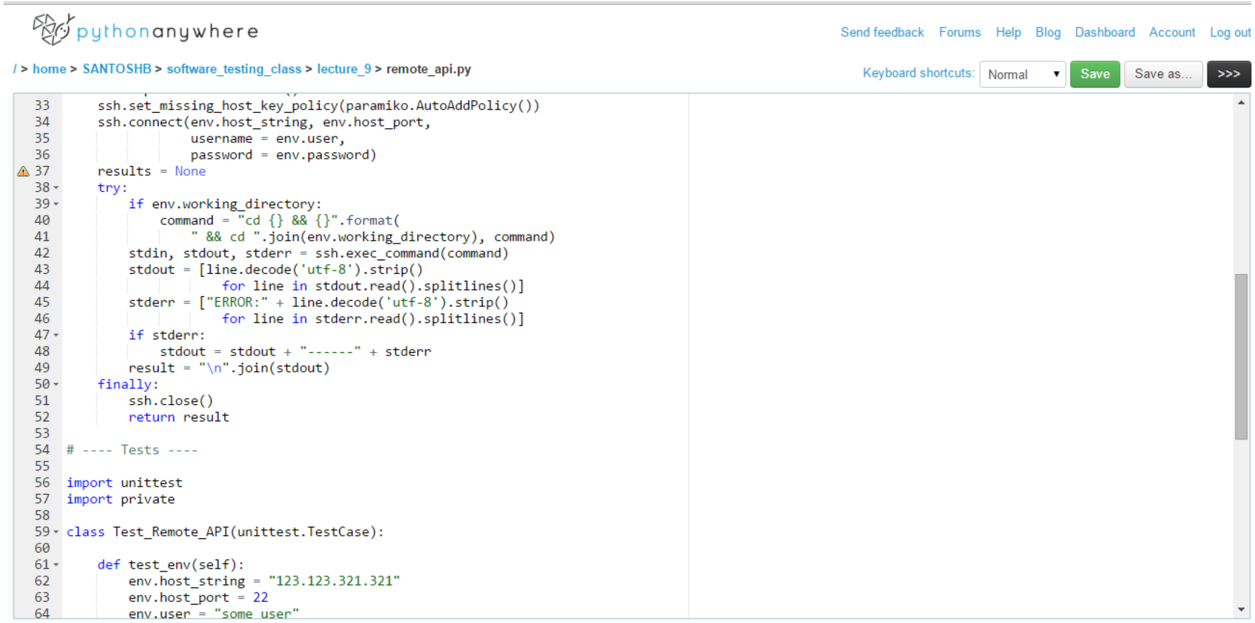
Vi remote_api.py

Code:

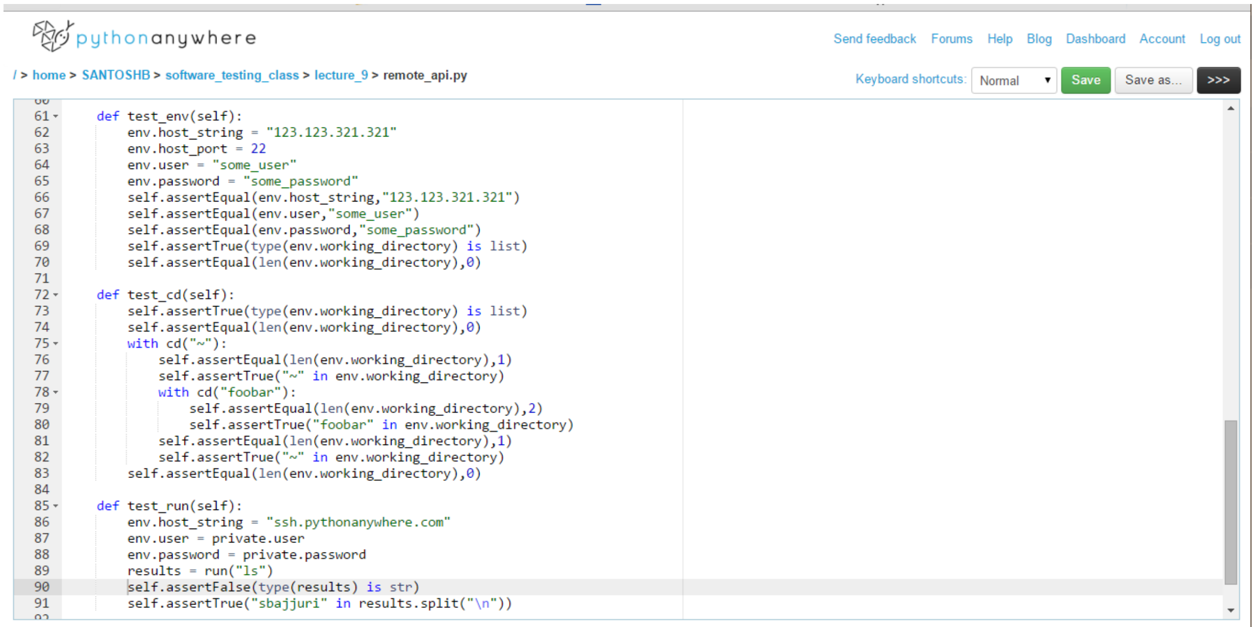


```
#
# remote_api.py
#
# Contains calls to allow command execution on remote (and local) machines,
# using the paramiko library to create an SSH connection.
#
# Greg DeLozier, 11/17/2016
# gdelozie@kent.edu
#

import paramiko

from contextlib import contextmanager

# ---- Environment ----
class Environment(object):
    host_string = ""
    host_port = 22
    user = ""
    password = None
    working_directory = list()

env = Environment()

@contextmanager
def cd(path):
    env.working_directory.append(path)
    yield
    env.working_directory.pop()

def run(command):
    ssh = paramiko.SSHClient()
```

Copyright © 2015 PythonAnywhere LLP — Terms — Privacy
"Python" is a registered trademark of the Python Software Foundation.



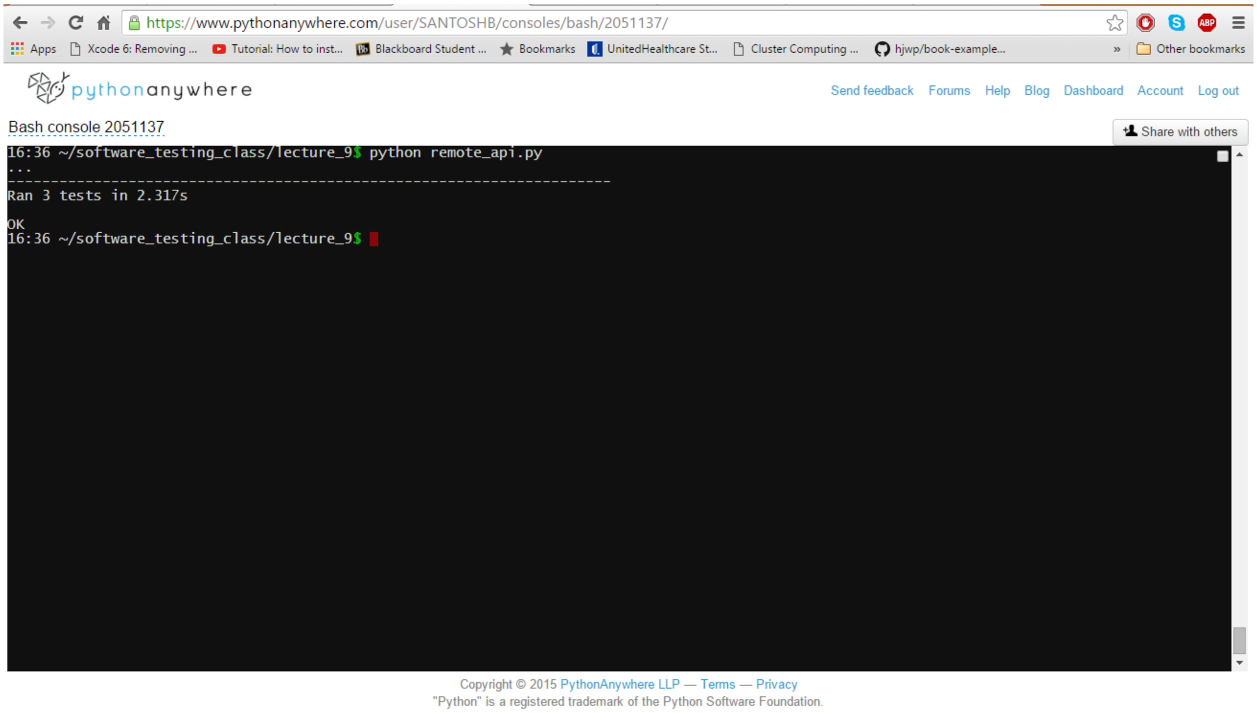
```python
    ssh.set_missing_host_key_policy(paramiko.AutoAddPolicy())
    ssh.connect(env.host_string, env.host_port,
                username = env.user,
                password = env.password)
    results = None
    try:
        if env.working_directory:
            command = "cd {} && {}".format(
                " && cd ".join(env.working_directory), command)
        stdin, stdout, stderr = ssh.exec_command(command)
        stdout = [line.decode('utf-8').strip()
                    for line in stdout.read().splitlines()]
        stderr = ["ERROR:" + line.decode('utf-8').strip()
                    for line in stderr.read().splitlines()]
        if stderr:
            stdout = stdout + "------" + stderr
        result = "\n".join(stdout)
    finally:
        ssh.close()
        return result

# ---- Tests ----

import unittest
import private

class Test_Remote_API(unittest.TestCase):

    def test_env(self):
        env.host_string = "123.123.321.321"
        env.host_port = 22
        env.user = "some_user"
```



```python
    def test_env(self):
        env.host_string = "123.123.321.321"
        env.host_port = 22
        env.user = "some_user"
        env.password = "some_password"
        self.assertEqual(env.host_string,"123.123.321.321")
        self.assertEqual(env.user,"some_user")
        self.assertEqual(env.password,"some_password")
        self.assertTrue(type(env.working_directory) is list)
        self.assertEqual(len(env.working_directory),0)

    def test_cd(self):
        self.assertTrue(type(env.working_directory) is list)
        self.assertEqual(len(env.working_directory),0)
        with cd("~"):
            self.assertEqual(len(env.working_directory),1)
            self.assertTrue("~" in env.working_directory)
            with cd("foobar"):
                self.assertEqual(len(env.working_directory),2)
                self.assertTrue("foobar" in env.working_directory)
            self.assertEqual(len(env.working_directory),1)
            self.assertTrue("~" in env.working_directory)
        self.assertEqual(len(env.working_directory),0)

    def test_run(self):
        env.host_string = "ssh.pythonanywhere.com"
        env.user = private.user
        env.password = private.password
        results = run("ls")
        self.assertFalse(type(results) is str)
        self.assertTrue("sbajjuri" in results.split("\n"))
```

Running the remote_api.py using

Output: *Python remote_api.py*



```
16:36 ~/software_testing_class/lecture_9$ python remote_api.py
...
----------------------------------------------------------------------
Ran 3 tests in 2.317s

OK
16:36 ~/software_testing_class/lecture_9$
```

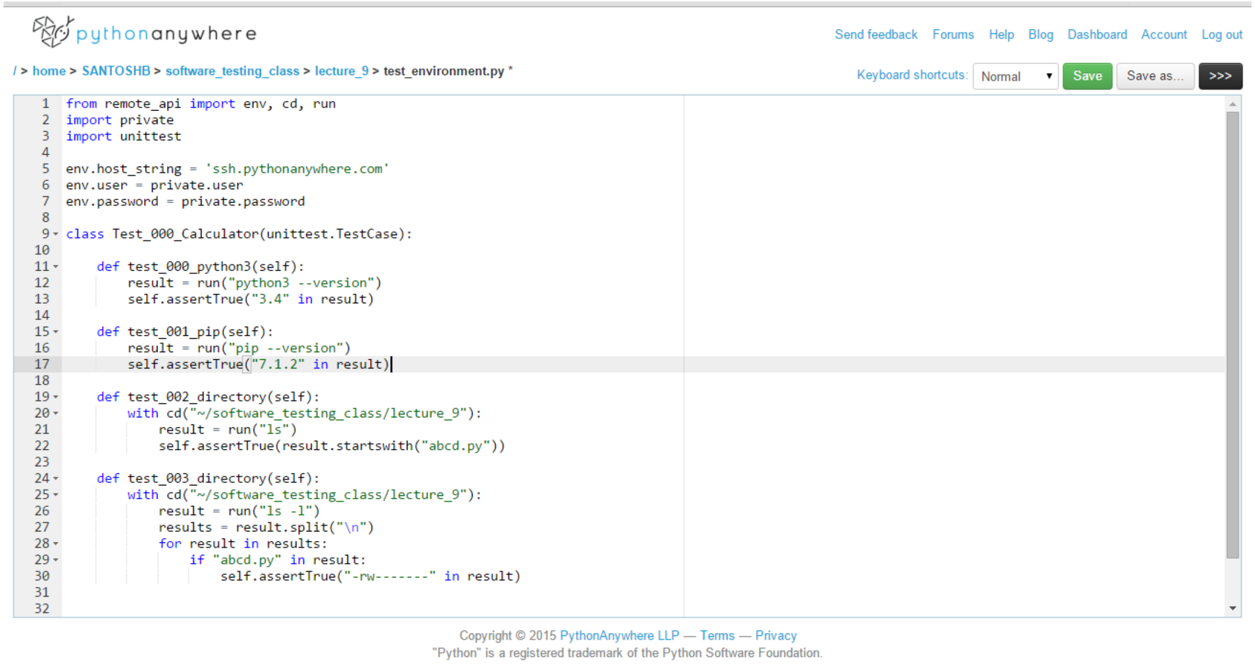
Copyright © 2015 PythonAnywhere LLP — Terms — Privacy
"Python" is a registered trademark of the Python Software Foundation.

TEST_ENVIRONMENT.PY

Executing the test_environment.py by editing the file taken from the repo.

vi test_environment.py

Code: *test_environment.py*

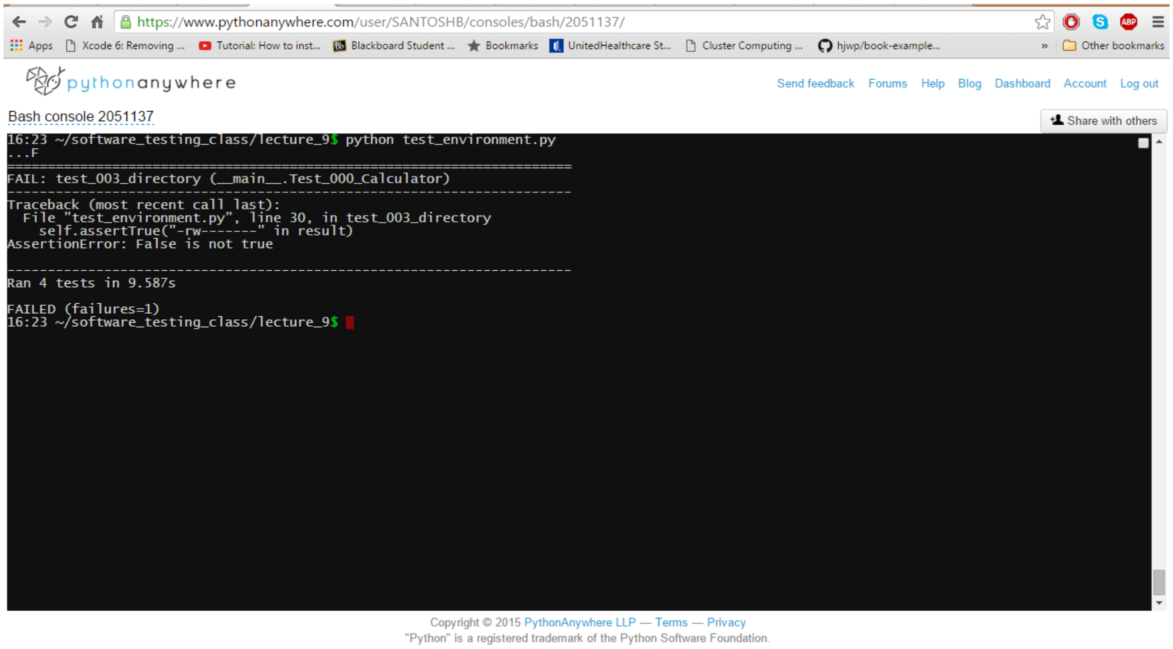


pythonanywhere
Send feedback Forums Help Blog Dashboard Account Log out
/ > home > SANTOSHB > software_testing_class > lecture_9 > test_environment.py *
Keyboard shortcuts: Normal
Save
Save as...
>>>

```python
from remote_api import env, cd, run
import private
import unittest

env.host_string = 'ssh.pythonanywhere.com'
env.user = private.user
env.password = private.password

class Test_000_Calculator(unittest.TestCase):

    def test_000_python3(self):
        result = run("python3 --version")
        self.assertTrue("3.4" in result)

    def test_001_pip(self):
        result = run("pip --version")
        self.assertTrue("7.1.2" in result)

    def test_002_directory(self):
        with cd("~/software_testing_class/lecture_9"):
            result = run("ls")
            self.assertTrue(result.startswith("abcd.py"))

    def test_003_directory(self):
        with cd("~/software_testing_class/lecture_9"):
            result = run("ls -l")
            results = result.split("\n")
            for result in results:
                if "abcd.py" in result:
                    self.assertTrue("-rw-------" in result)
```

Running the output using:

Output: *python test_environment.py*



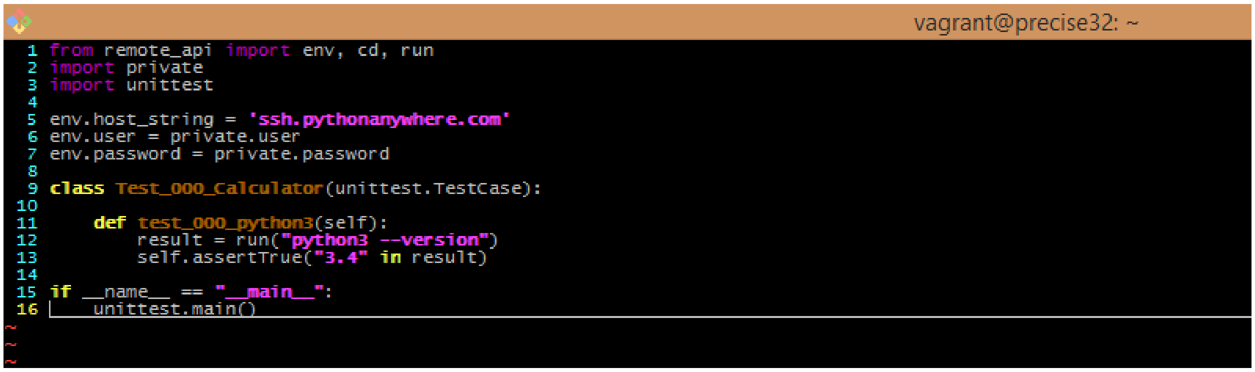
```
16:23 ~/software_testing_class/lecture_9$ python test_environment.py
...F
======================================================================
FAIL: test_003_directory (__main__.Test_000_Calculator)
----------------------------------------------------------------------
Traceback (most recent call last):
  File "test_environment.py", line 30, in test_003_directory
    self.assertTrue("-rw-------" in result)
AssertionError: False is not true

----------------------------------------------------------------------
Ran 4 tests in 9.587s

FAILED (failures=1)
16:23 ~/software_testing_class/lecture_9$
```

TEST CASE 1:

Running the sun.py using ***"vi version.py".***

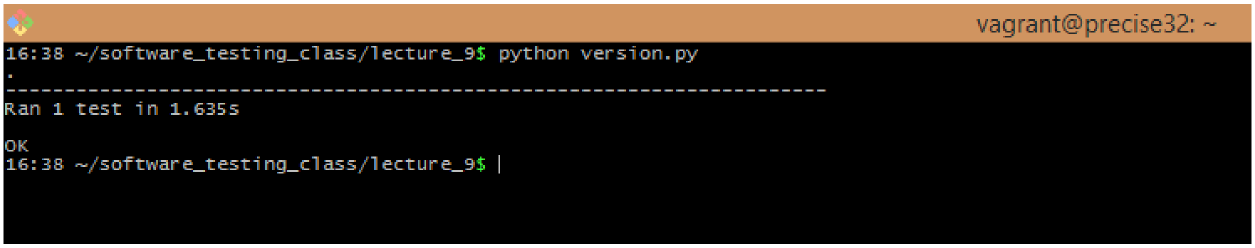
This program will determine whether there is a python version of "3.4" in the PythonAnyWhere.com. If there is a python version of 3.4, then the test will pass with the time stamp. Otherwise, the test will be failed.

Code



```
from remote_api import env, cd, run
import private
import unittest

env.host_string = 'ssh.pythonanywhere.com'
env.user = private.user
env.password = private.password

class Test_000_Calculator(unittest.TestCase):

    def test_000_python3(self):
        result = run("python3 --version")
        self.assertTrue("3.4" in result)

if __name__ == "__main__":
    unittest.main()
~
~
~
```

Output



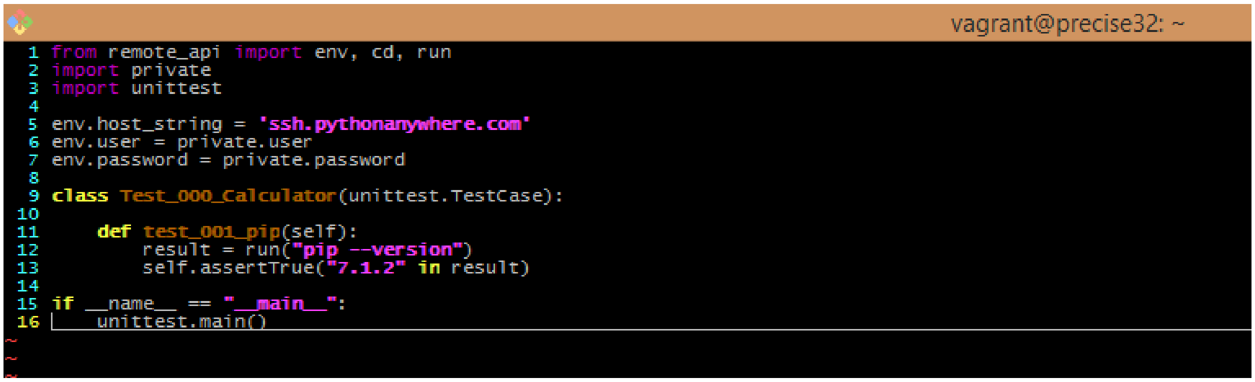
```
16:38 ~/software_testing_class/lecture_9$ python version.py
.
----------------------------------------------------------------------
Ran 1 test in 1.635s

OK
16:38 ~/software_testing_class/lecture_9$ |
```

TEST CASE 2:

In this test case, we have written an automation case where it finds for “pip –version” to be installed of 7.1.2

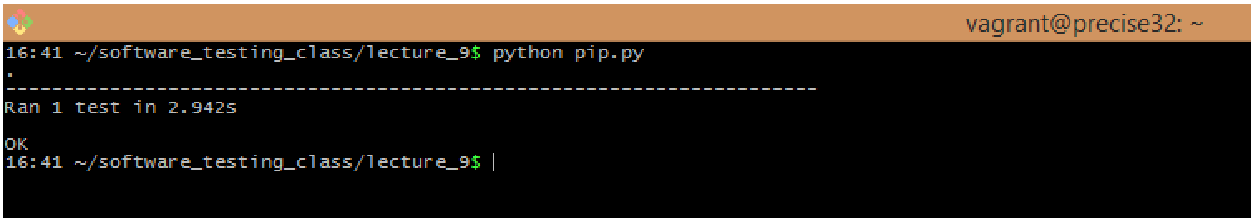
If that version is installed, then the test case will pass or else the test will fail.

Code



```
from remote_api import env, cd, run
import private
import unittest

env.host_string = 'ssh.pythonanywhere.com'
env.user = private.user
env.password = private.password

class Test_000_Calculator(unittest.TestCase):

    def test_001_pip(self):
        result = run("pip --version")
        self.assertTrue("7.1.2" in result)

if __name__ == "__main__":
    unittest.main()
~
~
~
```

Output



```
16:41 ~/software_testing_class/lecture_9$ python pip.py
.
----------------------------------------------------------------------
Ran 1 test in 2.942s

OK
16:41 ~/software_testing_class/lecture_9$
```

TEST CASE 3:

In this test case, I've written the test cases in such a way that the test case will find "Yarn" directory followed by its "yarn" sub-directory and then checks "tests" directory under the "yarn" for the files "test_env.py" and "yarnfile.py"

-- Yarn

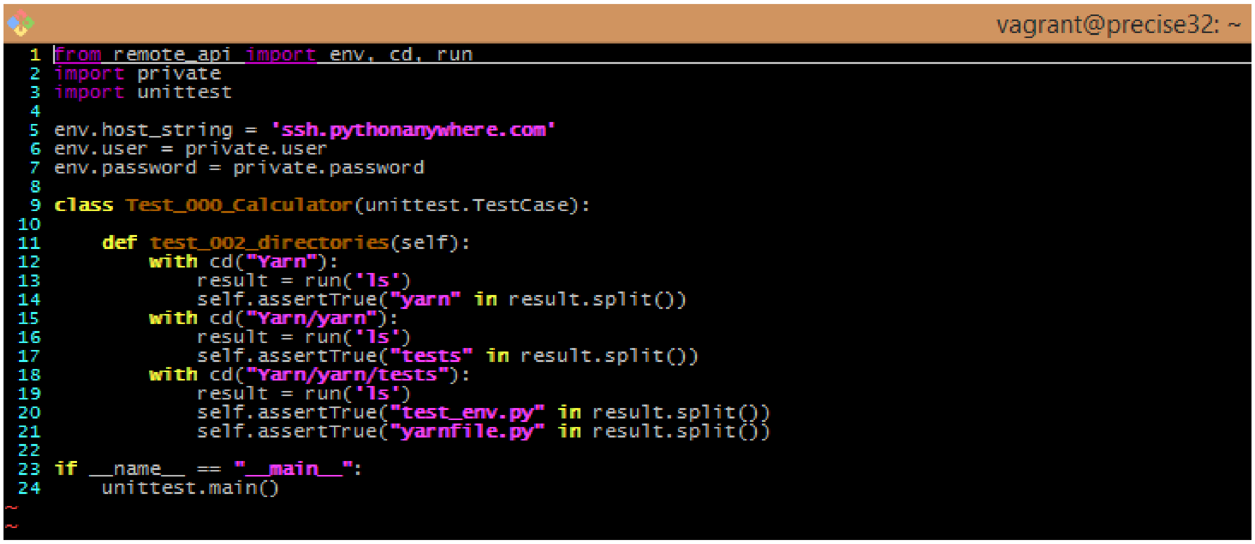
-- yarn

--tests

-- test_env.py

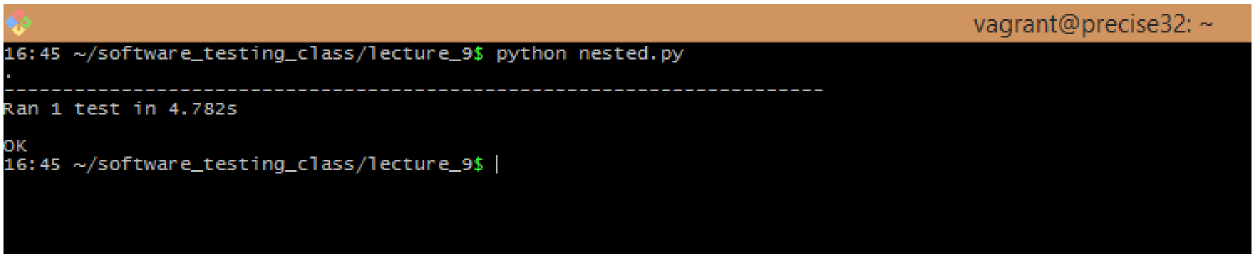
--yarnfile.py

Code



```
from remote_api import env, cd, run
import private
import unittest

env.host_string = 'ssh.pythonanywhere.com'
env.user = private.user
env.password = private.password

class Test_000_Calculator(unittest.TestCase):

    def test_002_directories(self):
        with cd("Yarn"):
            result = run('ls')
            self.assertTrue("yarn" in result.split())
        with cd("Yarn/yarn"):
            result = run('ls')
            self.assertTrue("tests" in result.split())
        with cd("Yarn/yarn/tests"):
            result = run('ls')
            self.assertTrue("test_env.py" in result.split())
            self.assertTrue("yarnfile.py" in result.split())

if __name__ == "__main__":
    unittest.main()
~
~
```

Output

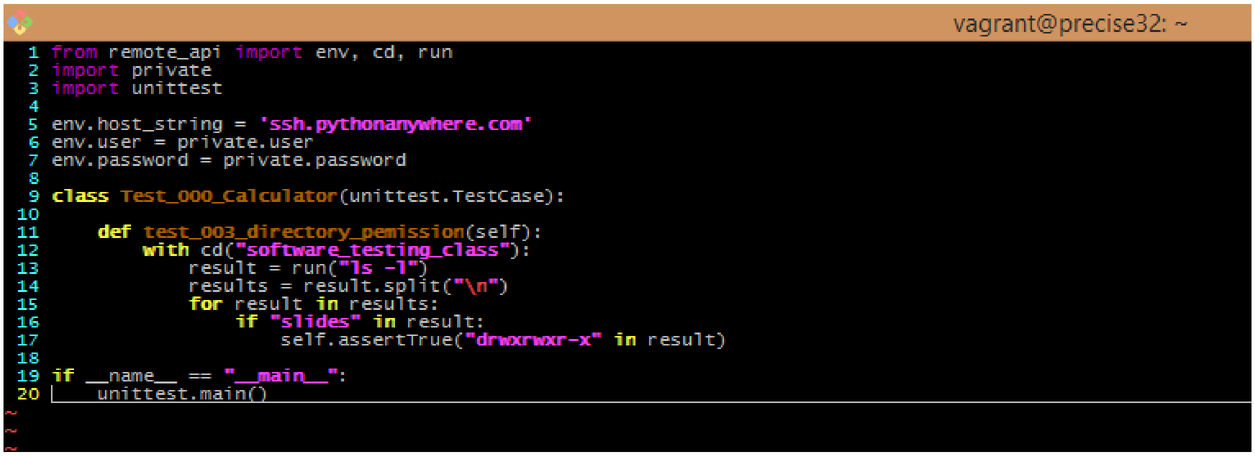


```
16:45 ~/software_testing_class/lecture_9$ python nested.py
.
----------------------------------------------------------------------
Ran 1 test in 4.782s

OK
16:45 ~/software_testing_class/lecture_9$ |
```

TEST CASE 4:

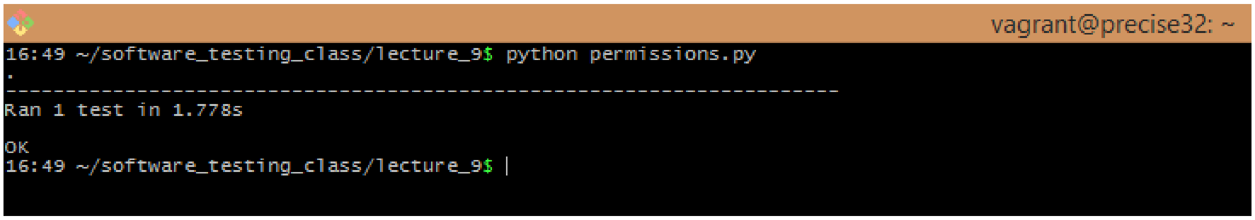
This test case will determine the permissions of the directory “software_testing_class”

Code



```python
from remote_api import env, cd, run
import private
import unittest

env.host_string = 'ssh.pythonanywhere.com'
env.user = private.user
env.password = private.password

class Test_000_Calculator(unittest.TestCase):

    def test_003_directory_pemission(self):
        with cd("software_testing_class"):
            result = run("ls -l")
            results = result.split("\n")
            for result in results:
                if "slides" in result:
                    self.assertTrue("drwxrwxr-x" in result)

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

Output

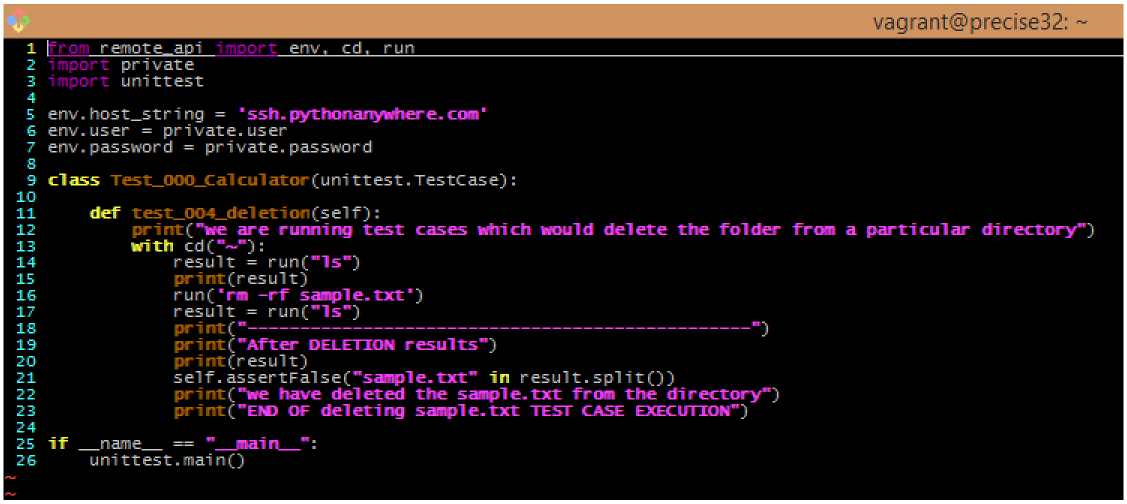


```
16:49 ~/software_testing_class/lecture_9$ python permissions.py
.
----------------------------------------------------------------------
Ran 1 test in 1.778s

OK
16:49 ~/software_testing_class/lecture_9$ |
```

TEST CASE 5:

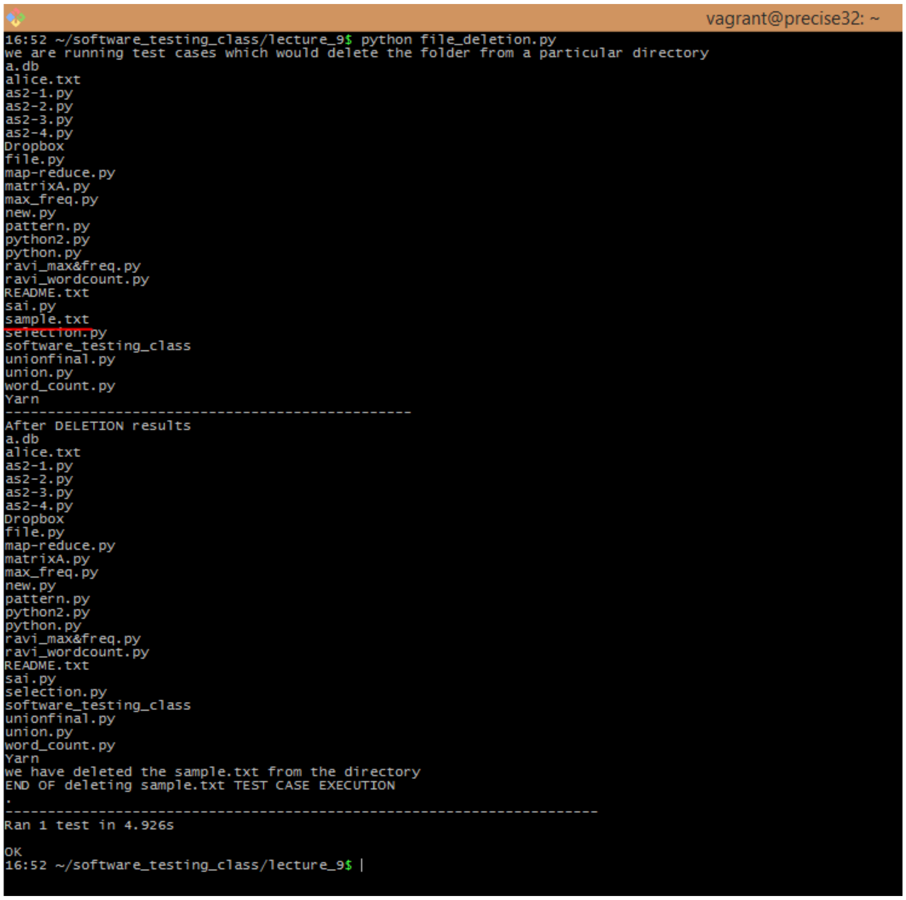
In this test case, the program will search for a text file “*sample.txt*”, then removes it from the directory and displaying all the files in that directory just to make sure.

Code



```python
from remote_api import env, cd, run
import private
import unittest

env.host_string = 'ssh.pythonanywhere.com'
env.user = private.user
env.password = private.password

class Test_000_Calculator(unittest.TestCase):

    def test_004_deletion(self):
        print("we are running test cases which would delete the folder from a particular directory")
        with cd("~"):
            result = run("ls")
            print(result)
            run('rm -rf sample.txt')
            result = run("ls")
            print("-------------------------------------------------")
            print("After DELETION results")
            print(result)
            self.assertFalse("sample.txt" in result.split())
            print("we have deleted the sample.txt from the directory")
            print("END OF deleting sample.txt TEST CASE EXECUTION")

if __name__ == "__main__":
    unittest.main()
~
~
```

Output

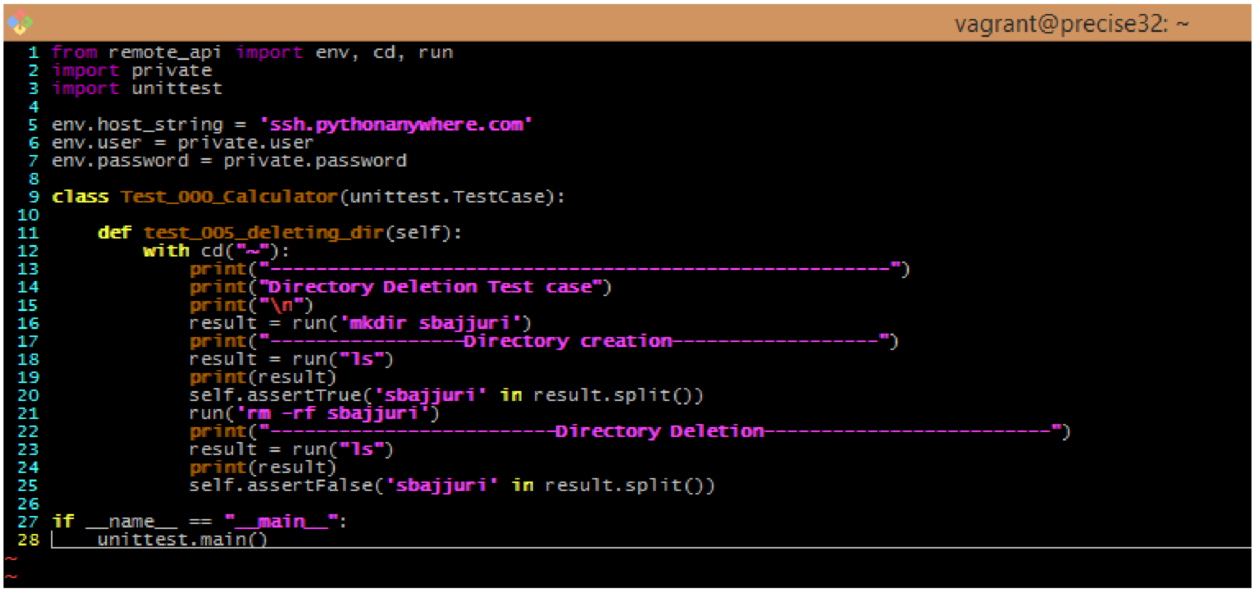


```
16:52 ~/software_testing_class/lecture_9$ python file_deletion.py
we are running test cases which would delete the folder from a particular directory
a.db
alice.txt
as2-1.py
as2-2.py
as2-3.py
as2-4.py
Dropbox
file.py
map-reduce.py
matrixA.py
max_freq.py
new.py
pattern.py
python2.py
python.py
ravi_max&freq.py
ravi_wordcount.py
README.txt
sai.py
sample.txt
selection.py
software_testing_class
unionfinal.py
union.py
word_count.py
Yarn
-------------------------------------------------
After DELETION results
a.db
alice.txt
as2-1.py
as2-2.py
as2-3.py
as2-4.py
Dropbox
file.py
map-reduce.py
matrixA.py
max_freq.py
new.py
pattern.py
python2.py
python.py
ravi_max&freq.py
ravi_wordcount.py
README.txt
sai.py
selection.py
software_testing_class
unionfinal.py
union.py
word_count.py
Yarn
we have deleted the sample.txt from the directory
END OF deleting sample.txt TEST CASE EXECUTION
.
----------------------------------------------------------------------
Ran 1 test in 4.926s

OK
16:52 ~/software_testing_class/lecture_9$ |
```

TEST CASE 6:

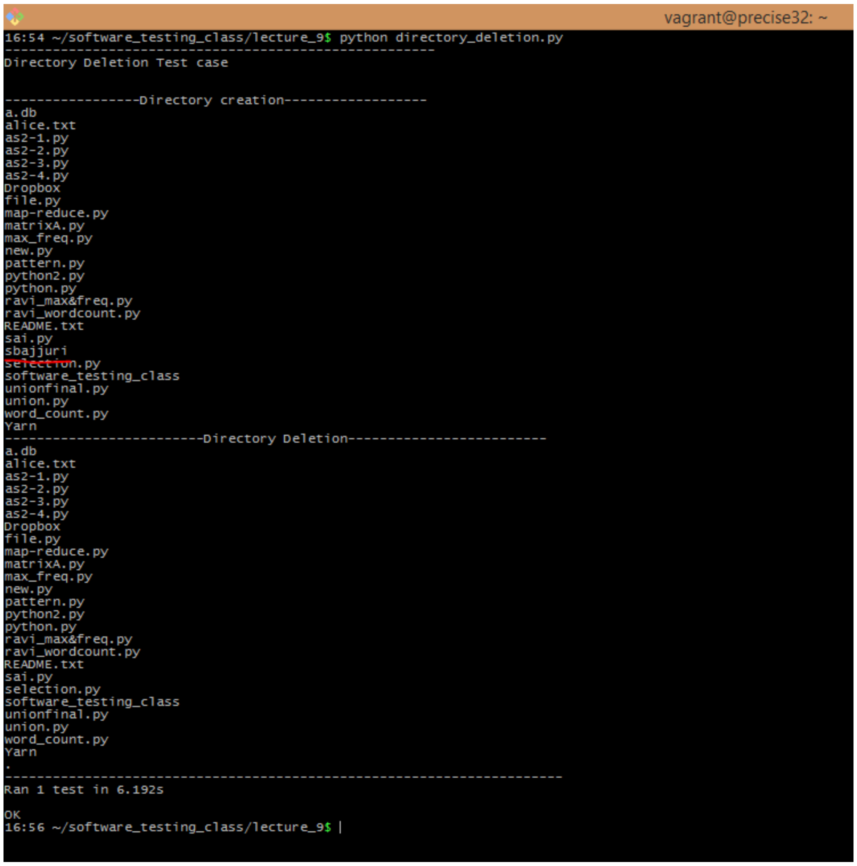
This test case will first create a directory "sbajjuri" and delete that directory "sbajjuri" again.

Code



```python
from remote_api import env, cd, run
import private
import unittest

env.host_string = 'ssh.pythonanywhere.com'
env.user = private.user
env.password = private.password

class Test_000_Calculator(unittest.TestCase):

    def test_005_deleting_dir(self):
        with cd("~"):
            print("-----------------------------------------------------------------")
            print("Directory Deletion Test case")
            print("\n")
            result = run('mkdir sbajjuri')
            print("-----------------Directory creation------------------")
            result = run("ls")
            print(result)
            self.assertTrue('sbajjuri' in result.split())
            run('rm -rf sbajjuri')
            print("-------------------------Directory Deletion-------------------------")
            result = run("ls")
            print(result)
            self.assertFalse('sbajjuri' in result.split())

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

Output



```
16:54 ~/software_testing_class/lecture_9$ python directory_deletion.py
----------------------------------------------------------------------
Directory Deletion Test case


----------------Directory creation------------------
a.db
alice.txt
as2-1.py
as2-2.py
as2-3.py
as2-4.py
Dropbox
file.py
map-reduce.py
matrixA.py
max_freq.py
new.py
pattern.py
python2.py
python.py
ravi_max&freq.py
ravi_wordcount.py
README.txt
sai.py
sbajjuri
selection.py
software_testing_class
unionfinal.py
union.py
word_count.py
Yarn
-------------------------Directory Deletion-------------------------
a.db
alice.txt
as2-1.py
as2-2.py
as2-3.py
as2-4.py
Dropbox
file.py
map-reduce.py
matrixA.py
max_freq.py
new.py
pattern.py
python2.py
python.py
ravi_max&freq.py
ravi_wordcount.py
README.txt
sai.py
selection.py
software_testing_class
unionfinal.py
union.py
word_count.py
Yarn
.
----------------------------------------------------------------------
Ran 1 test in 6.192s

OK
16:56 ~/software_testing_class/lecture_9$ |
```

TEST CASE 7:

In this test case, we perform several operations here. They are:

a) Make a directory (sbajjuri)

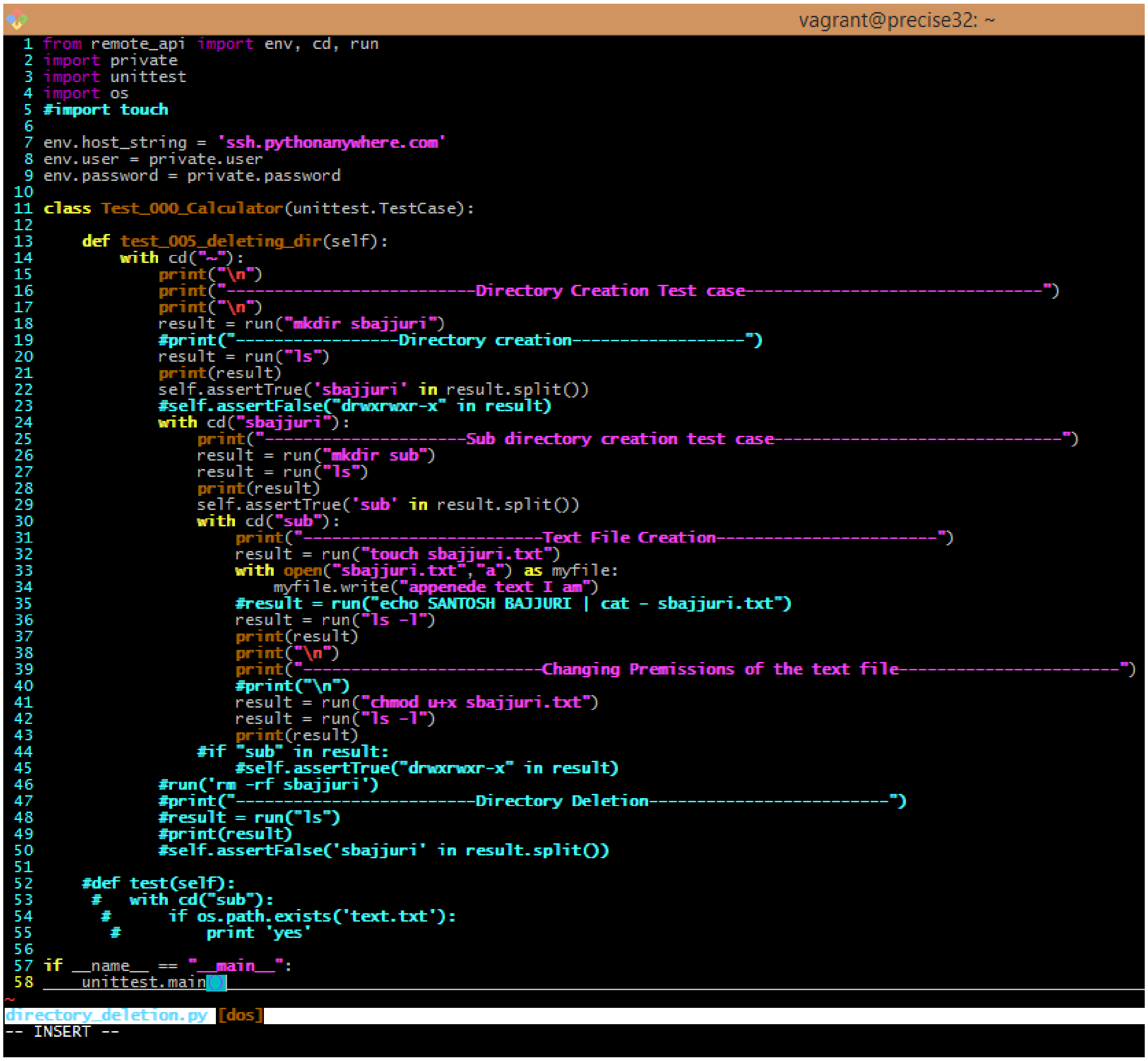
b) Make a sub directory under the directory (sub)

c) Create a text file in this sub directory (sbajjuri.txt)

d) Getting the permissions of this sbajjuri.txt

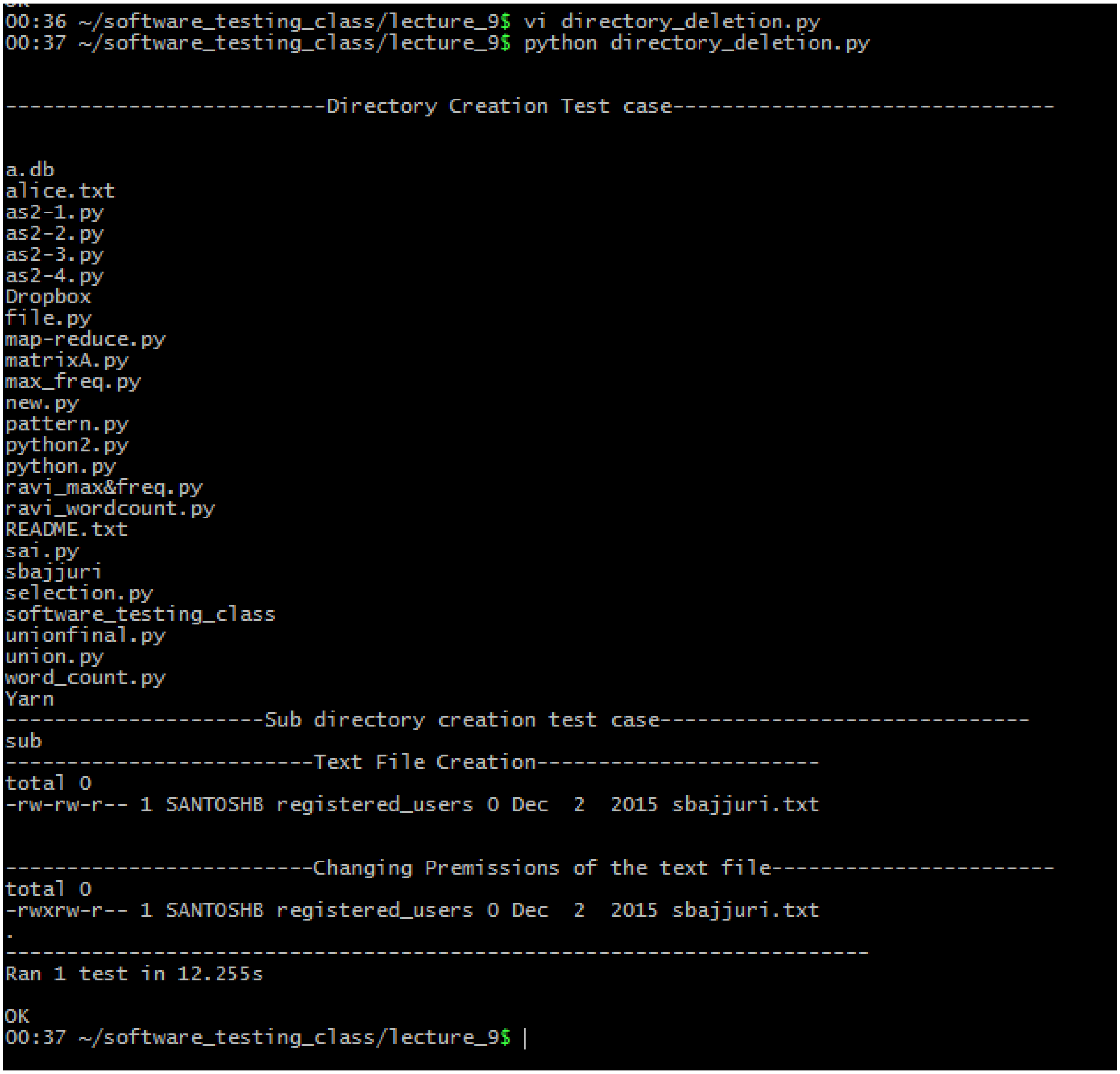
e) Changing the permissions of this text file

Code:



```
from remote_api import env, cd, run
import private
import unittest
import os
#import touch

env.host_string = 'ssh.pythonanywhere.com'
env.user = private.user
env.password = private.password

class Test_000_Calculator(unittest.TestCase):

    def test_005_deleting_dir(self):
        with cd("~"):
            print("\n")
            print("--------------------------Directory Creation Test case-------------------------------")
            print("\n")
            result = run("mkdir sbajjuri")
            #print("-----------------Directory creation------------------")
            result = run("ls")
            print(result)
            self.assertTrue('sbajjuri' in result.split())
            #self.assertFalse("drwxrwxr-x" in result)
            with cd("sbajjuri"):
                print("---------------------Sub directory creation test case------------------------------")
                result = run("mkdir sub")
                result = run("ls")
                print(result)
                self.assertTrue('sub' in result.split())
                with cd("sub"):
                    print("-------------------------Text File Creation-----------------------")
                    result = run("touch sbajjuri.txt")
                    with open("sbajjuri.txt","a") as myfile:
                        myfile.write("appenede text I am")
                    #result = run("echo SANTOSH BAJJURI | cat - sbajjuri.txt")
                    result = run("ls -l")
                    print(result)
                    print("\n")
                    print("-------------------------Changing Premissions of the text file-----------------------")
                    #print("\n")
                    result = run("chmod u+x sbajjuri.txt")
                    result = run("ls -l")
                    print(result)
                #if "sub" in result:
                    #self.assertTrue("drwxrwxr-x" in result)
            #run('rm -rf sbajjuri')
            #print("-------------------------Directory Deletion-------------------------")
            #result = run("ls")
            #print(result)
            #self.assertFalse('sbajjuri' in result.split())

    #def test(self):
     #   with cd("sub"):
      #      if os.path.exists('text.txt'):
       #         print 'yes'

if __name__ == "__main__":
    unittest.main()
~
directory_deletion.py [dos]
-- INSERT --
```

Output:



```
00:36 ~/software_testing_class/lecture_9$ vi directory_deletion.py
00:37 ~/software_testing_class/lecture_9$ python directory_deletion.py


--------------------------Directory Creation Test case-------------------------------


a.db
alice.txt
as2-1.py
as2-2.py
as2-3.py
as2-4.py
Dropbox
file.py
map-reduce.py
matrixA.py
max_freq.py
new.py
pattern.py
python2.py
python.py
ravi_max&freq.py
ravi_wordcount.py
README.txt
sai.py
sbajjuri
selection.py
software_testing_class
unionfinal.py
union.py
word_count.py
Yarn
---------------------Sub directory creation test case------------------------------
sub
-------------------------Text File Creation------------------------
total 0
-rw-rw-r-- 1 SANTOSHB registered_users 0 Dec  2  2015 sbajjuri.txt


-------------------------Changing Premissions of the text file------------------------
total 0
-rwxrw-r-- 1 SANTOSHB registered_users 0 Dec  2  2015 sbajjuri.txt
.
----------------------------------------------------------------------
Ran 1 test in 12.255s

OK
00:37 ~/software_testing_class/lecture_9$ |
```