CED03
SOFTWARE TESTING

PRACTICAL FILE

ANJALI MEENA

2018UCO1511

COE-1

PROBLEM 1:

Test a program by designing test cases using Boundary Value analysis

CODE:

soft.py file

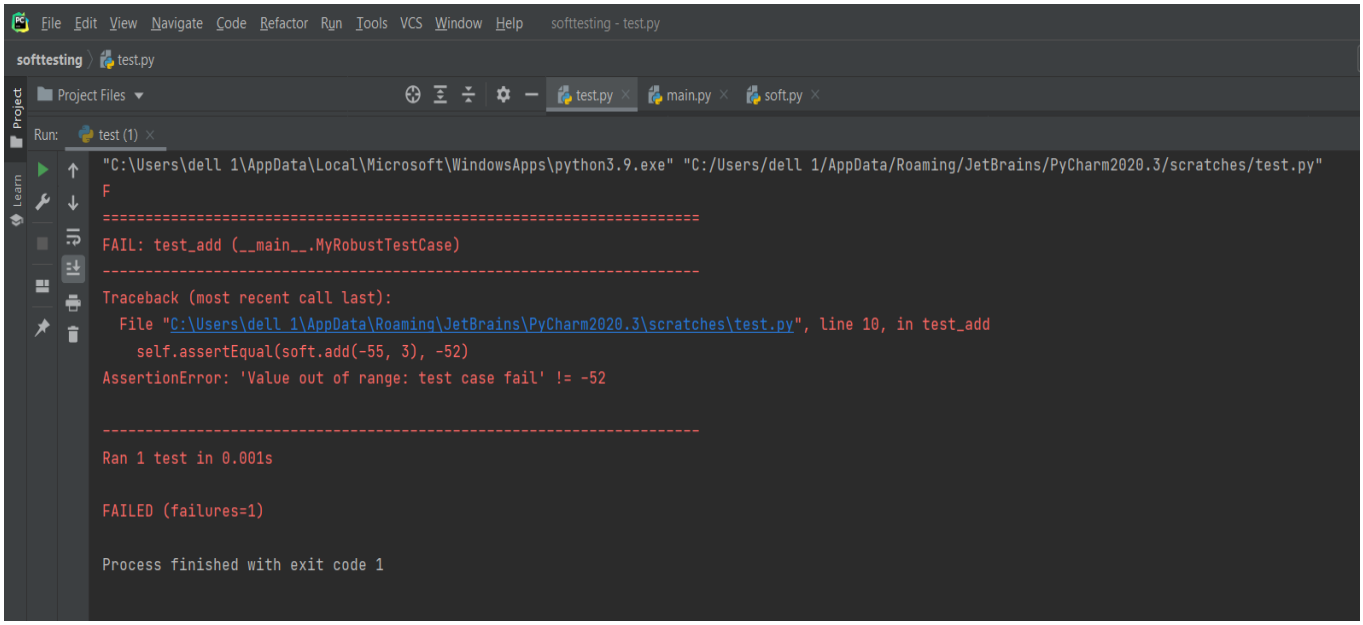
```
def add(x, y):
    if (x < -50 or y < -50) or (x > 50 or y > 50):
        ab = "Value out of range: test case fail"
        return ab
    return x + y
```

test.py file

```
import unittest
import soft



class MyRobustTestCase(unittest.TestCase):

    def test_add(self):
        self.assertEqual(soft.add(-11, 2), -9)
        self.assertEqual(soft.add(-55, 3), -52)
        self.assertEqual(soft.add(34, 43), 77)
        self.assertEqual(soft.add(9, 3), 12)
        self.assertEqual(soft.add(-40, 20), -20)
        self.assertEqual(soft.add(62, 23), 85)
        self.assertEqual(soft.add(1, 31), 32)
        self.assertEqual(soft.add(76, -87), -11)
        self.assertEqual(soft.add(23, 12), 35)
        self.assertEqual(soft.add(-5, -6), -11)
        self.assertEqual(soft.add(24, -24), 0)
        self.assertEqual(soft.add(2, 2), 4)
        self.assertEqual(soft.add(-5, 3), -2)

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

OUTPUT:

If we give a value out of range, one test case will fail



PROBLEM 2:

Test a program by designing test cases using Equivalence partitioning

CODE:

jee.py file

```
class Error(BaseException):
    pass


class OutOfRangeError(Error):
    def _init_(self, message):
        self.message = message


class CriteriaFalse(Error):
    def _init_(self, message):
        self.message = message


def checkcriteria(p, c, m):
    if p < 60 or p > 99:
        raise OutOfRangeError('Physics is out of the given
range')
    if c < 60 or c > 99:
        raise OutOfRangeError('Chemistry is out of the given
range')
    if m < 75 or m > 99:
        raise OutOfRangeError('Maths is out of the given
range')


def checkTotalmarks(p, c, m):
    if (p+c+m)/3 < 70:
        raise CriteriaFalse('Total percentage is not meeting
criteria ')


def qualified(p, c,m):
    checkcriteria(p, c, m)
    checkTotalmarks(p, c, m)

    if ((p+c+m)/3 <60):
        return "Not Qualified"

    else:
        return "Qualified"
```

```
def main():
    try:
        print("Enter marks obtained in Physics,
Chemistry,Maths")

        p = int(input('Enter marks obtained in Physics:'))
        c = int(input('Enter marks obtained in Chemistry: '))
        m = int(input('Enter marks obtained in Maths:'))
    except ValueError as v:
        print(v + " Raised :Input is not an integer.")
        exit(0)
    try:
        checkcriteria(p, c, m)
    except OutOfRangeError as e:
        print("OutOfRangeError:" + e.message)

    try:
        qualified(p, c, m)
    except CriteriaFalse as e:
        print('False criteria :' + e.message)

    status = qualified(p, c, m)

    print("Condition of student based on inserted marks: " +
status)


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

test_jee.py file

```
import pytest

from jee import OutOfRangeError
from jee import CriteriaFalse
from jee import qualified


def test_physics_min():
    with pytest.raises(OutOfRangeError):
        qualified(65, 65, 80)


def test_chemistry_min():
    with pytest.raises(OutOfRangeError):
        qualified(70, 50, 89)
```

```python
def test_maths_min():
    with pytest.raises(OutOfRangeError):
        qualified(90, 90, 70)


def test_physics_max():
    with pytest.raises(OutOfRangeError):
        qualified(100, 80, 82)


def test_chemistry_max():
    with pytest.raises(OutOfRangeError):
        qualified(61, 100, 95)


def test_maths_max():
    with pytest.raises(OutOfRangeError):
        qualified(80, 76, 100)


def test_total():
    with pytest.raises(CriteriaFalse):
        qualified(61, 61, 76)


def test_status():
    assert qualified(80, 80, 90) == "Qualified"
```

OUTPUT:

All test cases passed

```
rootdir: C:\Users\dell 1\PycharmProjects\Equivalence
collected 8 items

test_jee.py::test_physics_min PASSED
test_jee.py::test_chemistry_min PASSED
test_jee.py::test_maths_min PASSED
test_jee.py::test_physics_max PASSED
test_jee.py::test_chemistry_max PASSED
test_jee.py::test_maths_max PASSED
test_jee.py::test_total PASSED
test_jee.py::test_status PASSED

================================================================================ 8 passed in 0.05s ======

C:\Users\dell 1\PycharmProjects\Equivalence>
```

One test case failing, if we give out of range marks

```
================================================================================ FAILURES ===============
_________________________________________________________________________________ test_physics_min __________

    def test_physics_min():
        with pytest.raises(OutOfRangeError):
>           qualified(65, 65, 80)
E           Failed: DID NOT RAISE <class 'jee.OutOfRangeError'>

test_jee.py:10: Failed
============================================================================= short test summary info ======
FAILED test_jee.py::test_physics_min - Failed: DID NOT RAISE <class 'jee.OutOfRangeError'>
========================================================================== 1 failed, 7 passed in 0.08s ====

C:\Users\dell 1\PycharmProjects\Equivalence>
```

PROBLEM 3:

Combinatorial testing: Use following tolls to generate test cases using pairwise design

1. Pairwise Pict online

Test factors

Company: HP, ASUS, APPLE, DELL

Price: 70000, 50000, 80000, 100000

Size: 16, 15, 20, 32, 19, 34

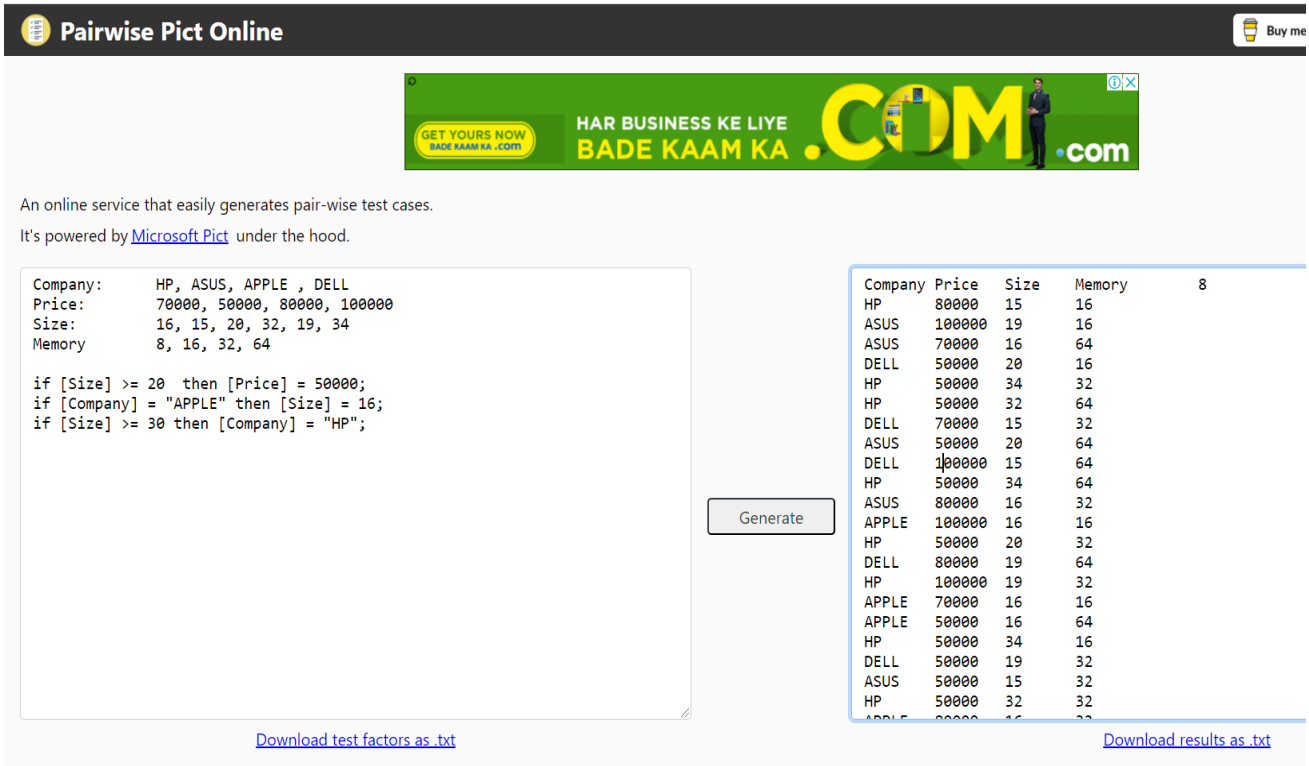
Memory 8, 16, 32, 64

if [Size] >= 20 then [Price] = 50000;

if [Company] = "APPLE" then [Size] = 16;

if [Size] >= 30 then [Company] = "HP";

Test results



2. CA Gen

Name Values Cardinality

Application

JFLAP,JMETER,Mozilla Firefox,MYSQL,Scrapy

5

Language

C++,Java,JavaScript,Python

4

HeapSize

1,4,2,3,6

5

SizeDisk

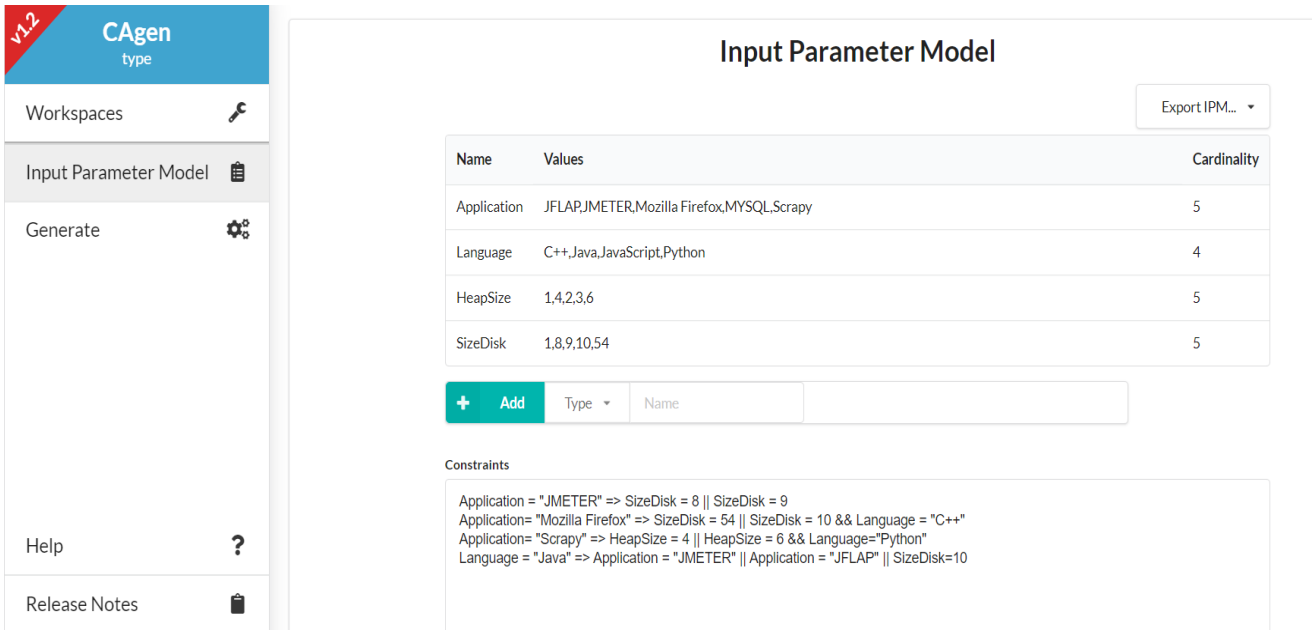
1,8,9,10,54

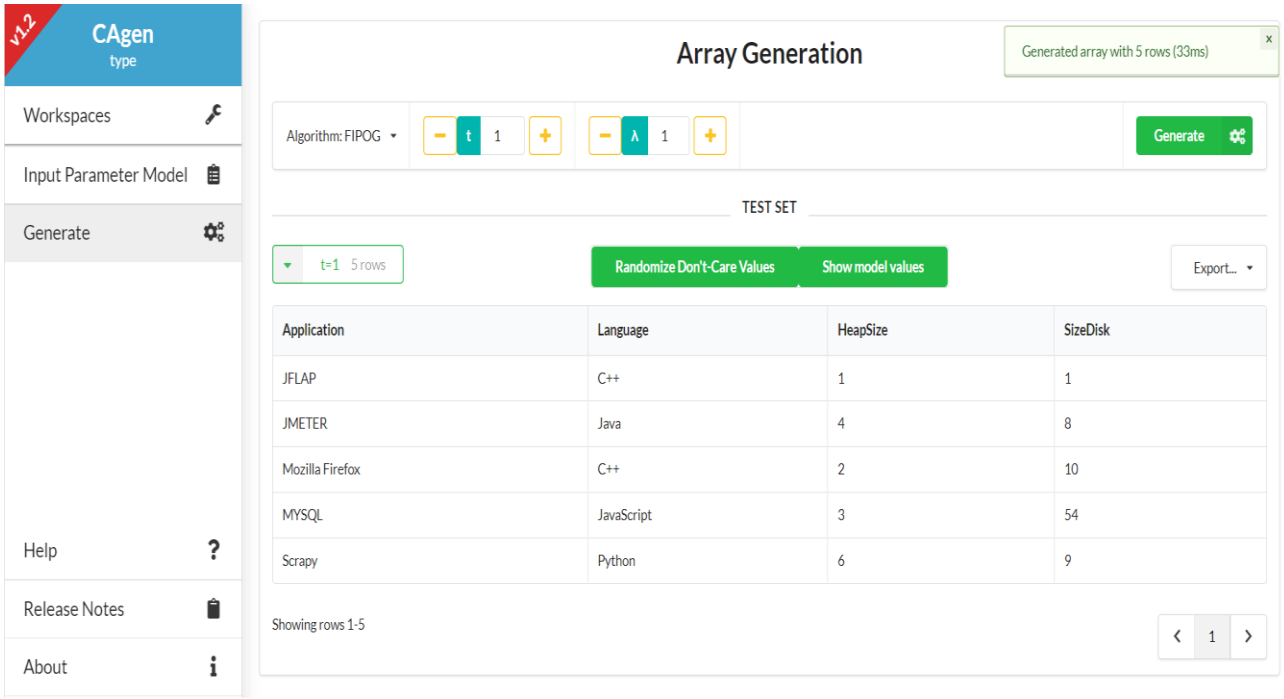
Application = "JMETER" => SizeDisk = 8 || SizeDisk = 9

Application= "Mozilla Firefox" => SizeDisk = 54 || SizeDisk = 10 && Language = "C++"

Application= "Scrapy" => HeapSize = 4 || HeapSize = 6 && Language="Python"

Language = "Java" => Application = "JMETER" || Application = "JFLAP" || SizeDisk=10





3. Test cover .com

Testcover.com

Results for Anjali Meena, NSUT

Tue Jun 08 06:52:01 UTC 2021

Company

Request data:

```
Company
Price
Size
Memory
#
+(HP ASUS) with (70000,50000)
HP ASUS
70000 50000
16 15 20 32 19 34
8 16 32 64
+(ASUS APPLE) with (50000,80000)
ASUS APPLE
50000 80000
16 15 20 32 19 34
8 16 32 64
+70000 with (32,64)
ASUS APPLE
50000 80000
70000
32 64
```

#1. Test case generation

Test Case ID	Price	Size	Memory	Factor 4	Combo Countdown Notice. 1 unnamed factor(s).
	3 Values	3 Values	7 Values	4 Values	103
1	HP	50000	19	16	97
2	ASUS	70000	16	64	91
3	HP	70000	32	32	85
4	ASUS	50000	34	8	79
5	APPLE	50000	15	32	73
6	APPLE	80000	32	64	67
7	ASUS	70000	20	16	62
8	ASUS	80000	16	32	57
9	HP	50000	20	8	53
10	HP	70000	15	64	49
11	APPLE	80000	34	8	45
12	APPLE	80000	15	16	41
13	APPLE	50000	70000	64	37
14	HP	70000	19	8	34
15	HP	70000	34	16	31
16	ASUS	50000	32	8	28
17	APPLE	80000	19	32	25
18	HP	50000	16	32	23
19	APPLE	50000	20	64	21
20	ASUS	80000	15	8	19
21	APPLE	80000	16	16	17
22	ASUS	50000	19	64	15
23	ASUS	50000	70000	32	13
24	HP	70000	20	32	12
25	ASUS	70000	34	32	11
26	ASUS	50000	16	8	10
27	ASUS	70000	32	16	9
28	ASUS	70000	34	64	8
29	ASUS	80000	20	8	7
30	APPLE	80000	70000	32	6

PROBLEM 4:

Write two classes connected through inheritance. Do the complete testing of classes using techniques discussed for OO testing and other testing techniques

CODE:

Product.py file

```
# base class
class Birds:

   name = ""

   def __init__(self):
      print("super class Birds")

   def sayYourName(self):

      pass


# derived class 1
class Avocet(Birds):

   Age = 10
   Species="American Avocet"

   def __init__(self, name, age):
      self.name = name
      self.Age = age

   def printDetails(self):
      print("name : " + self.name)
      print("Age : " + str(self.Age))
      print("Species : " + self.Species)

   def sayYourName(self):
      s = "I am a Avocet and my name is "+ self.name+" and I
am a "+ self.Species
      return s

#derived class 2
class Catbird(Birds):

   Age= 3
   Species="Spotted Catbird"

   def __init__(self, name, age):
      self.name = name
```

```
        self.Age = age

    def printDetails(self):
        print("name : " + self.name)
        print("Age : " + str(self.Age))
        print("Species : " + self.Species)

    def sayYourName(self):
        s = "I am a Catbird and my name is "+ self.name+" and I
am a "+ self.Species
        return s



def main():
    s = Avocet('Rocky', 12)
    s.printDetails()
    print(s.sayYourName())

    p = Birds()

    l = Catbirds('Creepy', 4)
    l.printDetails()
    print(l.sayYourName())


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

test_product.py

```
# importing the modules
import pytest
from product import Avocet,Catbird


class TestAvocet:

    def test_details(self):
        d = Avocet('Rocky' , 12)
        print("testing details : Avocet")
        assert ('ocky' , 12, "American Avocet") ==
(d.name, d.Age, d.Species)

    def test_sayYourName(self):
        d = Avocet('Rocket', 10)
        print("testing details : Avocet")
        st = d.sayYourName()
```

```
        assert st == "I am a Avocet and my name is
Rocket and I am a American Avocet"

class TestCatbird:

    def test_details(self):
        l = Catbird('Creepy' , 3)
        print("testing details : Catbird")
        assert ('Creepy' , 3, "Spotted Catbird") ==
(l.name, l.Age, l.Species)

    def test_sayYourName(self):
        l = Catbird('Slacky', 2)
        print("testing details : Catbird")
        st = l.sayYourName()
        assert st == "I am a Catbird and my name is
Slacky and I am a Spotted Catbird"
```

OUTPUT:

All test cases passed

```
========================================================================== 4 passed in 0.03s ==========

(venv) C:\Users\dell 1\PycharmProjects\OOP2>pytest
========================================================================== test session starts ==========
platform win32 -- Python 3.8.8, pytest-6.2.3, py-1.10.0, pluggy-0.13.1
rootdir: C:\Users\dell 1\PycharmProjects\OOP2
collected 4 items

test_product.py ....

========================================================================== 4 passed in 0.03s ==========

(venv) C:\Users\dell 1\PycharmProjects\OOP2>
```

One test case failing

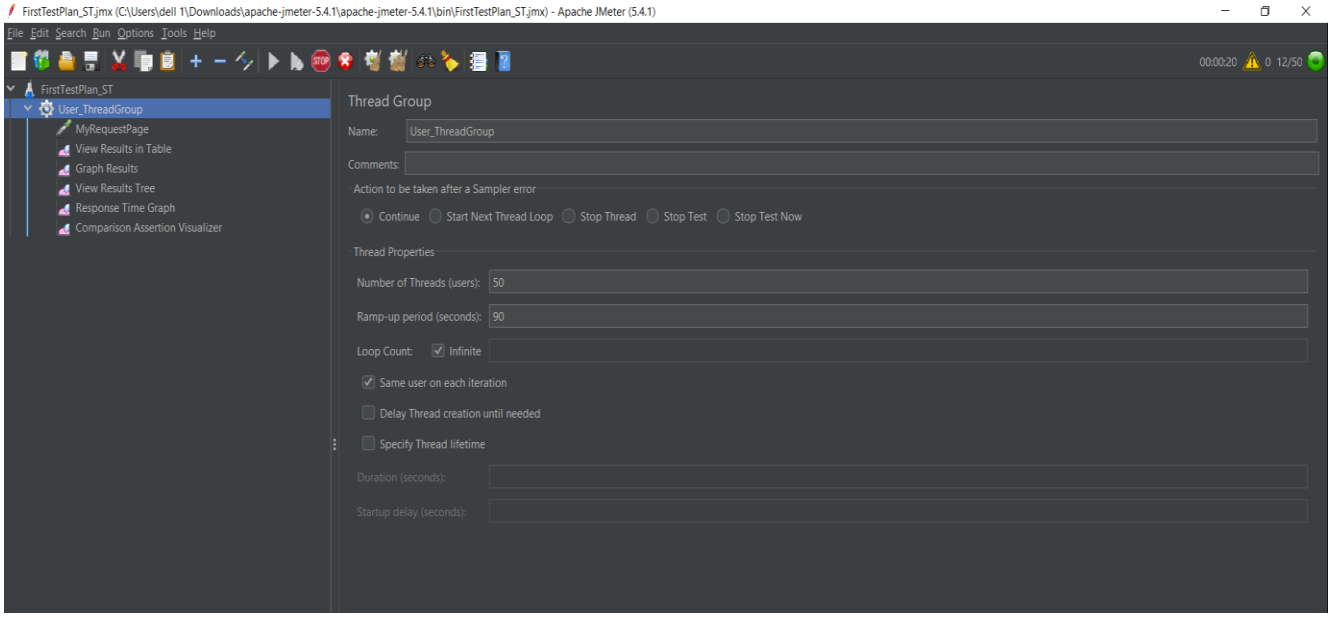
```
self = <test_product.TestAvocet object at 0x0000025576169E80>

    def test_details(self):
        d = Avocet('Rocky' , 12)
        print("testing details : Avocet")
>       assert ('ocky' , 12, "American Avocet") == (d.name, d.Age, d.Species)
E    AssertionError: assert ('ocky', 12, ...rican Avocet') == ('Rocky', 12,...rican Avocet')
E      At index 0 diff: 'ocky' != 'Rocky'
E      Full diff:
E      - ('Rocky', 12, 'American Avocet')
E      ?   -
E      + ('ocky', 12, 'American Avocet')

test_product.py:11: AssertionError
---------------------------------------------------------------------------- Captured stdout call ----------------------------------------
testing details : Avocet
========================================================================== short test summary info ==========================================
FAILED test_product.py::TestAvocet::test_details - AssertionError: assert ('ocky', 12, ...rican Avocet') == ('Rocky', 12,...rican Avocet')
========================================================================== 1 failed, 3 passed in 0.08s ==================================

(venv) C:\Users\dell 1\PycharmProjects\OOP2>
```

PROBLEM 5:

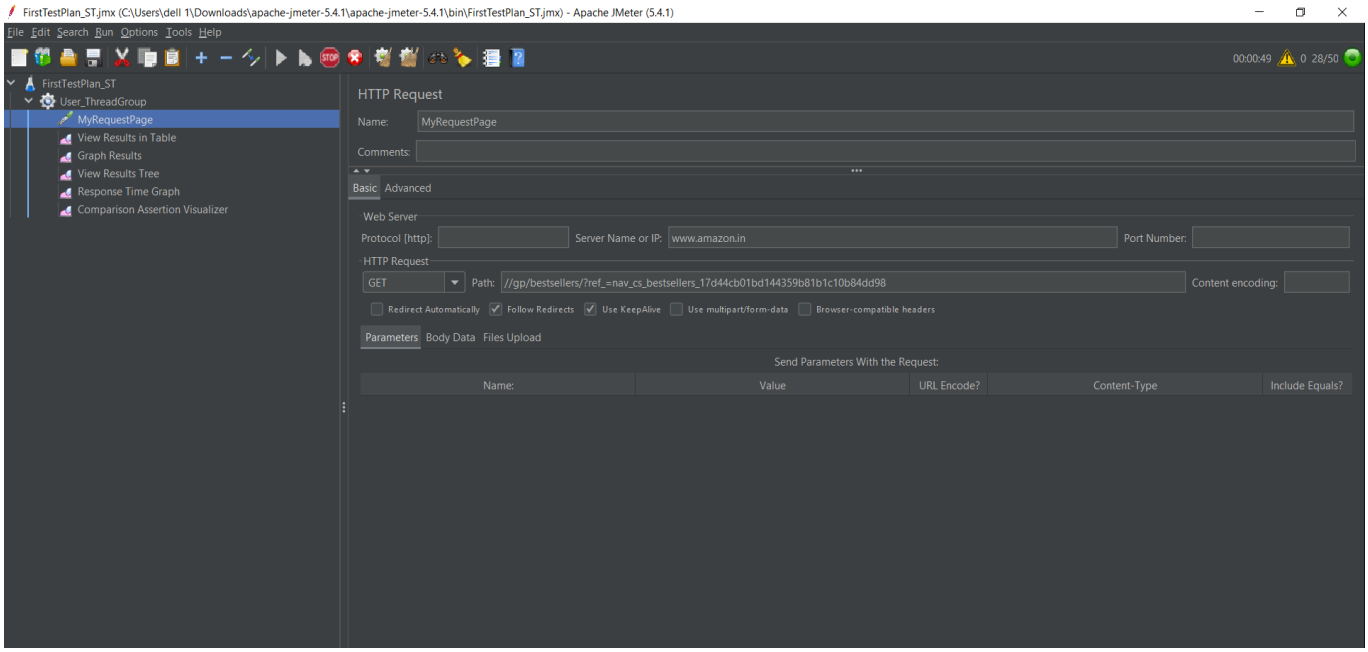
Explore the tool JMeter

Step 1: Open JMeter and make a Test Plan.

Step 2: Then add a thread group.

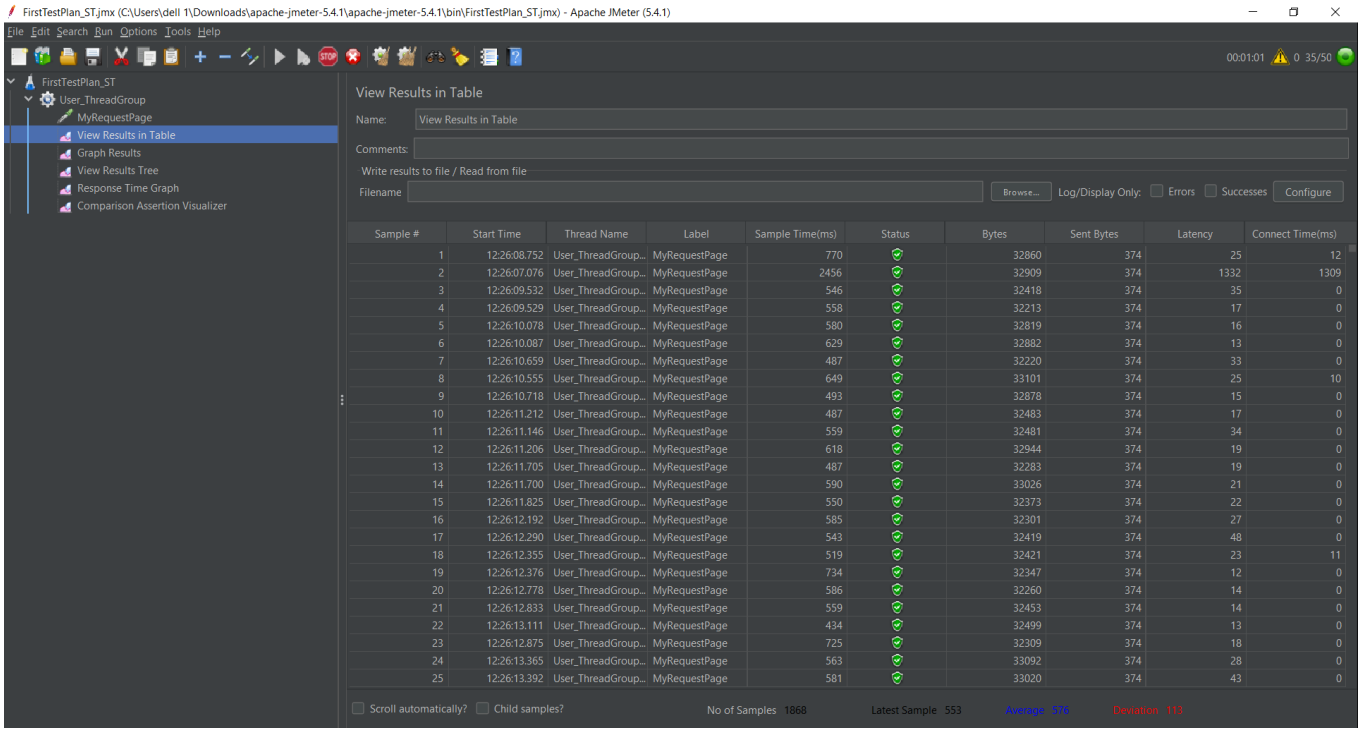


Step 3: Add a sampler (HTTP Request used here)



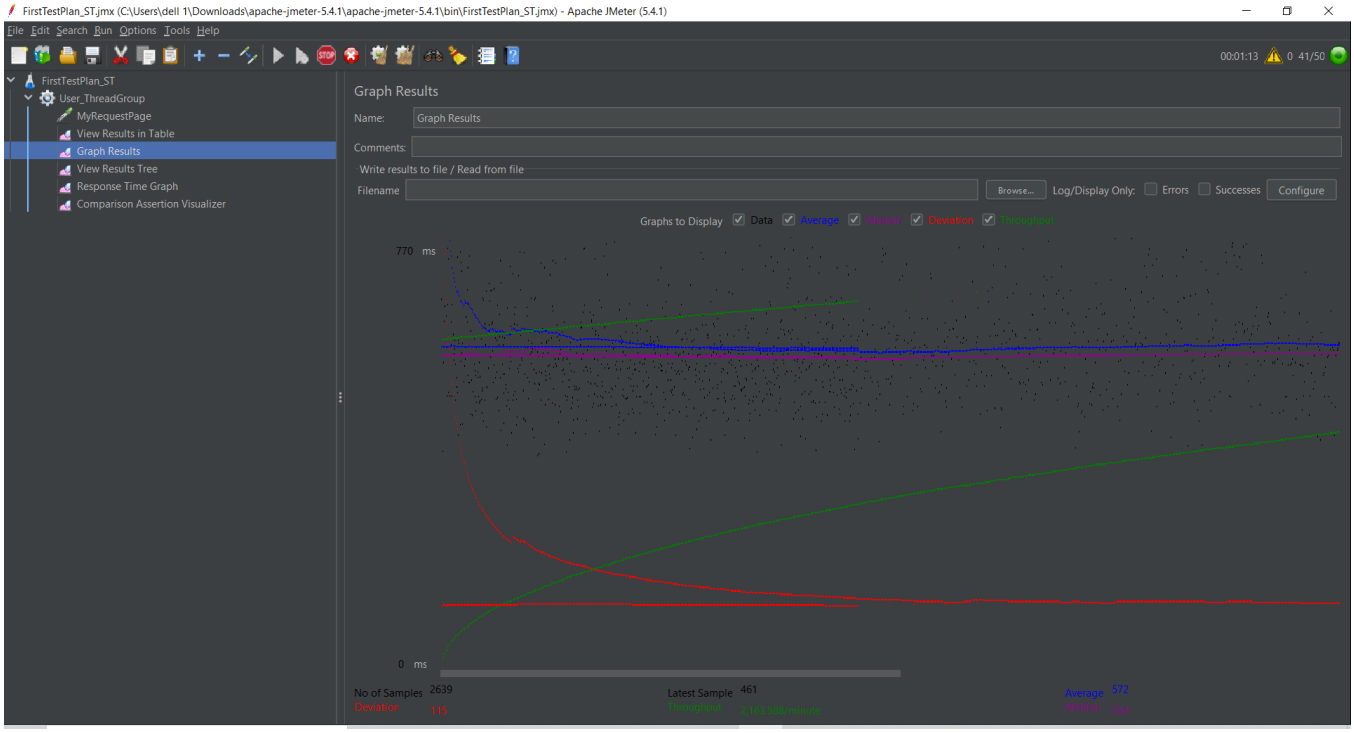
Step 4: Add few listeners.

i. Table results

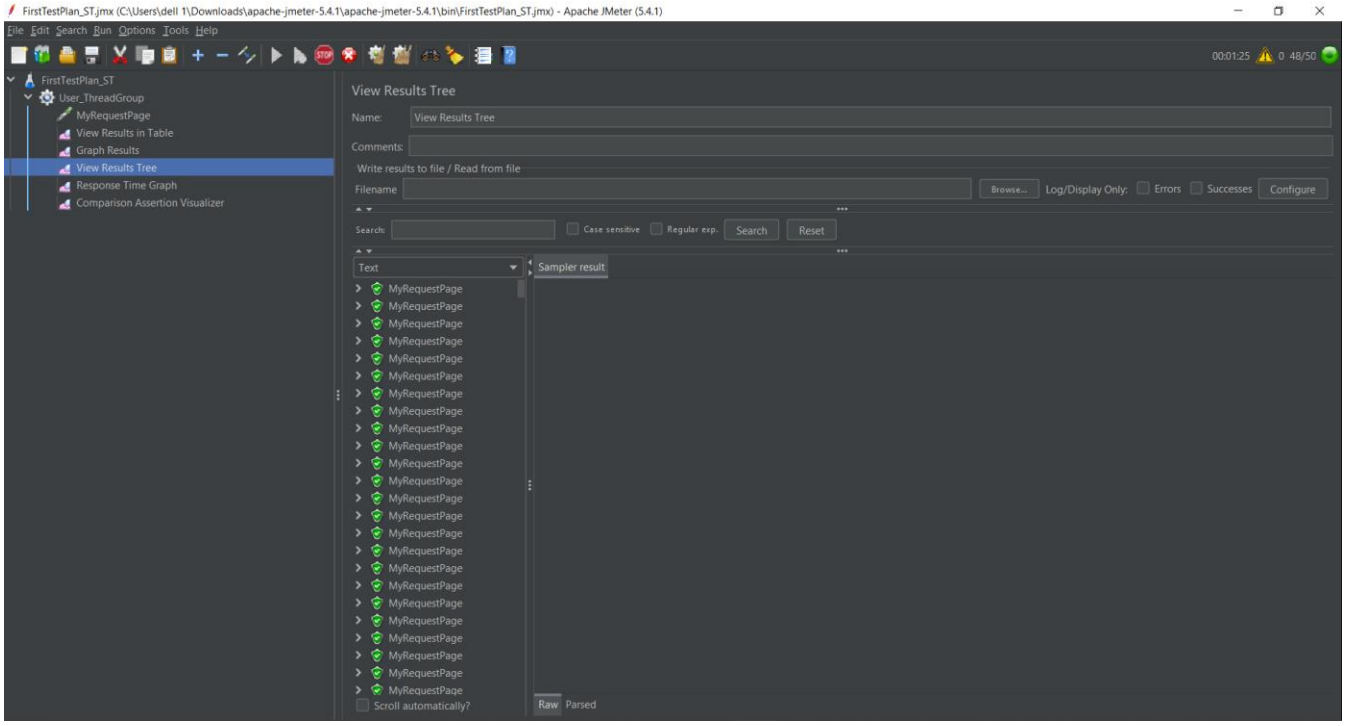


Sample #	Start Time	Thread Name	Label	Sample Time(ms)	Status	Bytes	Sent Bytes	Latency	Connect Time(ms)
1	12:26:08.752	User_ThreadGroup...	MyRequestPage	770		32860	374	25	12
2	12:26:07.076	User_ThreadGroup...	MyRequestPage	2456		32909	374	1332	1309
3	12:26:09.532	User_ThreadGroup...	MyRequestPage	546		32418	374	35	0
4	12:26:09.529	User_ThreadGroup...	MyRequestPage	558		32213	374	17	0
5	12:26:10.078	User_ThreadGroup...	MyRequestPage	580		32819	374	16	0
6	12:26:10.087	User_ThreadGroup...	MyRequestPage	629		32882	374	13	0
7	12:26:10.659	User_ThreadGroup...	MyRequestPage	487		32220	374	33	0
8	12:26:10.555	User_ThreadGroup...	MyRequestPage	649		33101	374	25	10
9	12:26:10.718	User_ThreadGroup...	MyRequestPage	493		32878	374	15	0
10	12:26:11.212	User_ThreadGroup...	MyRequestPage	487		32483	374	17	0
11	12:26:11.146	User_ThreadGroup...	MyRequestPage	559		32481	374	34	0
12	12:26:11.206	User_ThreadGroup...	MyRequestPage	618		32944	374	19	0
13	12:26:11.705	User_ThreadGroup...	MyRequestPage	487		32283	374	19	0
14	12:26:11.700	User_ThreadGroup...	MyRequestPage	590		33026	374	21	0
15	12:26:11.825	User_ThreadGroup...	MyRequestPage	550		32373	374	22	0
16	12:26:12.192	User_ThreadGroup...	MyRequestPage	585		32301	374	27	0
17	12:26:12.290	User_ThreadGroup...	MyRequestPage	543		32419	374	48	0
18	12:26:12.355	User_ThreadGroup...	MyRequestPage	519		32421	374	23	11
19	12:26:12.376	User_ThreadGroup...	MyRequestPage	734		32347	374	12	0
20	12:26:12.778	User_ThreadGroup...	MyRequestPage	586		32260	374	14	0
21	12:26:12.833	User_ThreadGroup...	MyRequestPage	559		32453	374	14	0
22	12:26:13.111	User_ThreadGroup...	MyRequestPage	434		32499	374	13	0
23	12:26:12.875	User_ThreadGroup...	MyRequestPage	725		32309	374	18	0
24	12:26:13.365	User_ThreadGroup...	MyRequestPage	563		33092	374	28	0
25	12:26:13.392	User_ThreadGroup...	MyRequestPage	581		33020	374	43	0

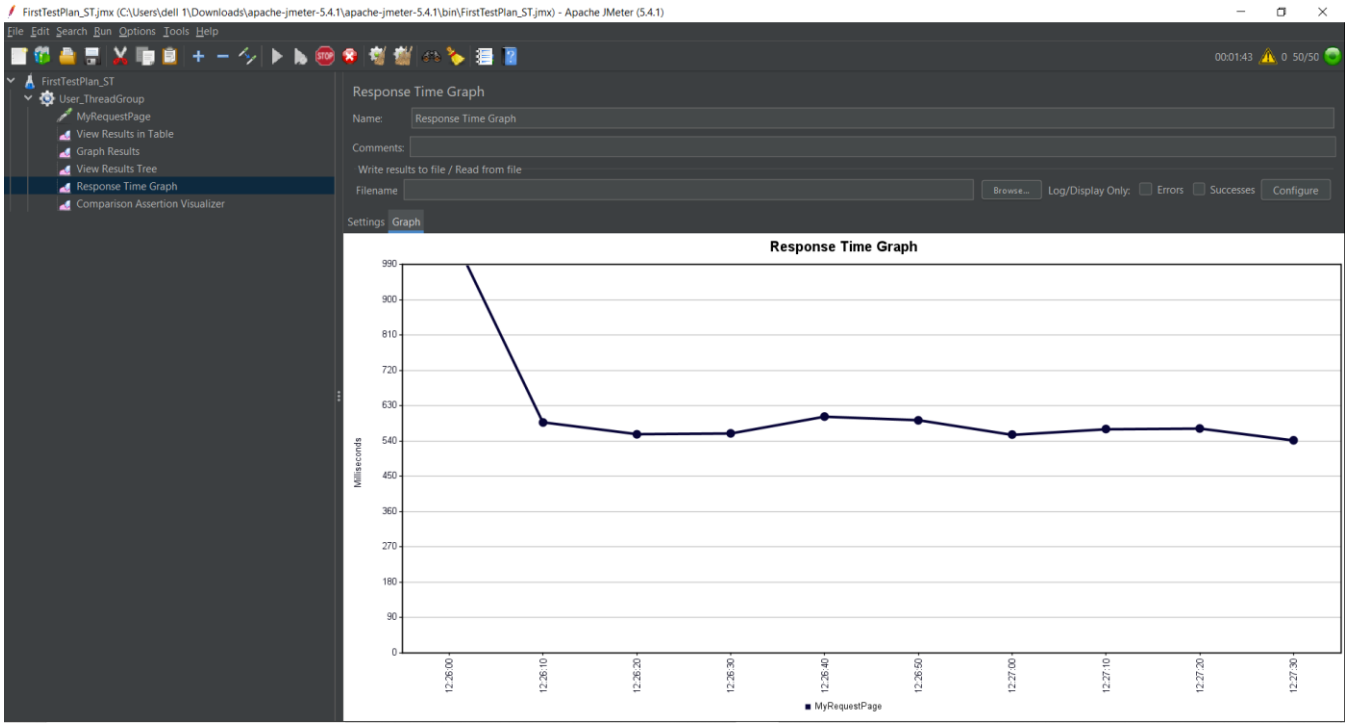
ii. Graph results



iii. Tree results



iv. Response time graph

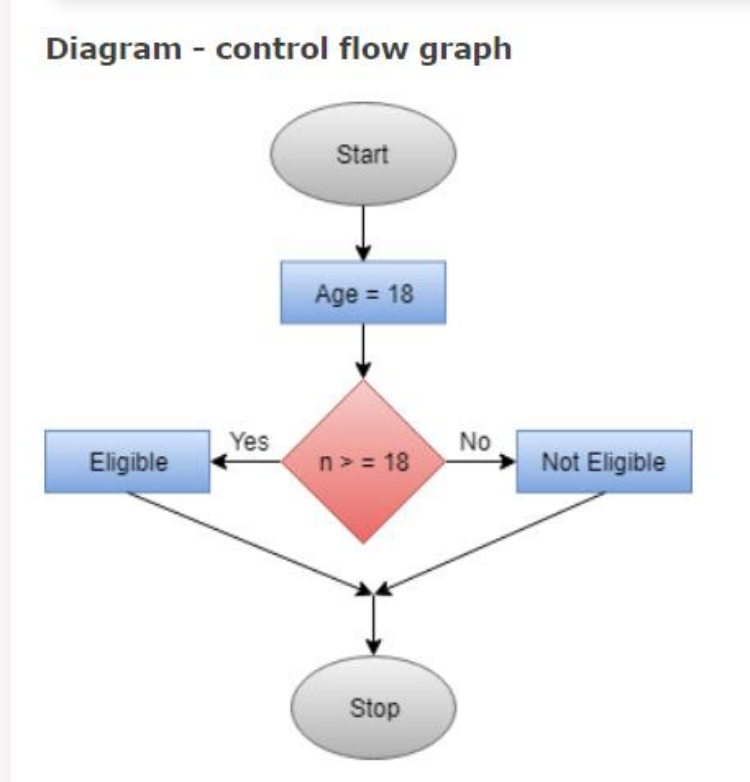


EXAMPLES OF WHITE BOX TESTING TECHNIQUES

1. Control flow **testing** example

Control flow testing is a testing technique that comes under white box testing. The aim of this technique is to determine the execution order of statements or instructions of the program through a control structure. The control structure of a program is used to develop a test case for the program.

```
void VoteEligiblityAge{
        int n=45;
        if(n>=18)
        {
           cout<<"You are eligible for voting"<<endl;
         } else
        {
          cout<<"You are not eligible for voting");
         }
    }
```



2. Data flow **testing** example

There are 8 statements in this code. In this code we cannot cover all 8 statements in a single path as if 2 is valid then **4, 5, 6, 7** are not traversed, and if 4 is valid then statement 2 and 3 will not be traversed.

```
1. read x;
2. If(x>0)                    (1, (2, t), x), (1, (2, f), x)
3. a= x+1                     (1, 3, x)
4. if (x<=0) {                (1, (4, t), x), (1, (4, f), x)
5. if (x<1)                   (1, (5, t), x), (1, (5, f), x)
6. x=x+1; (go to 5)           (1, 6, x)
else
7. a=x+1                      (1, 7, x)
8. print a;                   (6,(5, f)x), (6,(5,t)x)
                              (6, 6, x)
                              (3, 8, a), (7, 8, a).
```

Consider two paths so that we can cover all the statements.

x= 1

Path – 1, 2, 3, 8

Output = 2

If we consider **x = 1**, in step 1; x is assigned a value of 1 then we move to step 2 (since, x>0 we will move to statement **3 (a= x+1)** and at end, it will go to statement 8 and print x =2.

For the second path, we assign x as 1

Set x= -1

Path = 1, 2, 4, 5, 6, 5, 6, 5, 7, 8

Output = 2

x is set as 1 then it goes to step 1 to assign x as 1 and then moves to step 2 which is false as x is smaller than 0 (x>0 and here x=-1). It will then move to step 3 and then jump to step 4; as 4 is true (x<=0 and their x is less than 0) it will jump on 5 (x<1) which is true and it will move to step 6 **(x=x+1)** and here x is increased by 1.

So,

x=-1+1

x=0

x become 0 and it goes to step 5(x<1),as it is true it will jump to step

6 (x=x+1)

x=x+1

x= 0+1

x=1

x is now 1 and jump to step 5 (x<1) and now the condition is false and it will jump to step 7 **(a=x+1)** and set a=2 as x is 1. At the end the value of a is 2. And on step 8 we get the output as 2