EXPERIMENT - 1

Software Testing and Quality Assurance

Abstract

To determine the nature of roots of quadratic equations, its input is triple of +ve integers (say x,y,z) and values may be from interval[1,100] the program output may have one of the following:- [Not a Quadratic equations, Real roots, Imaginary roots, Equal roots]. Perform BVA and Robust Case Testing.

Syeda Reeha Quasar

14114802719

EXPERIMENT – 1

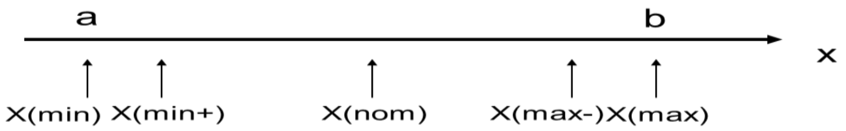
Aim:

Write a program to find whether a number is prime or not. To determine the nature of roots of a quadratic equation, its input is triple of +ve integers (say a,b,c) and values may be from the interval[1,100] The program output may have one of the following:-

- [Not Quadratic equations, Real roots, Imaginary roots, Equal roots]
- Perform BVA (Boundary-value analysis).
- Perform Robust Case Testing

Theory:

BVA(Boundary value analysis)is a black box test design technique and it is used to find the errors at boundaries of input domain rather than finding those errors in the center of input The basic boundary value analysis technique can be generalized in two ways: by the number of variables, and by the kinds of ranges. Generalizing the number of variables is easy: if we have a function of n variables, we hold all but one at the nominal values, and let the remaining variable assume the min, min+, nom, max- and max values, and repeat this for each variable. Thus for a function of n variables, boundary value analysis yields 4n + 1 test cases.



In the above program consider the value 1(minimum), 2(just above Minimum), 50 (Nominal), 99(Just below Maximum) and 100(Maximum). Total No. of test cases will be 4*3+1=13

If a, b, and c denote the three integer in quadratic equation a(x^2)+bx+c=0 then Calculate discriminant d=(b*b)-4*a*c

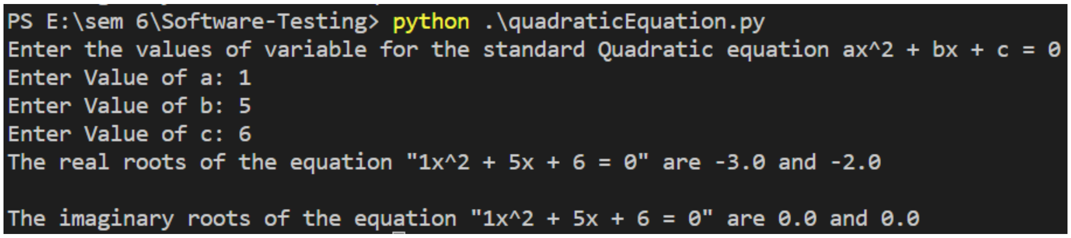
- if((a<1)||(b<1)||(c<1)||(a>100)||(b>100)||(c>100)) then "Invalid input"
- if(a==0) then "Not a quadratic equation”
- if (d==0) then "Roots are equal"
- if(d<0) then "Imaginary roots"
- otherwise "Real Roots"

Robust Test Cases –
Here, we go outside the legitimate boundary, it is an extension of boundary value analysis.

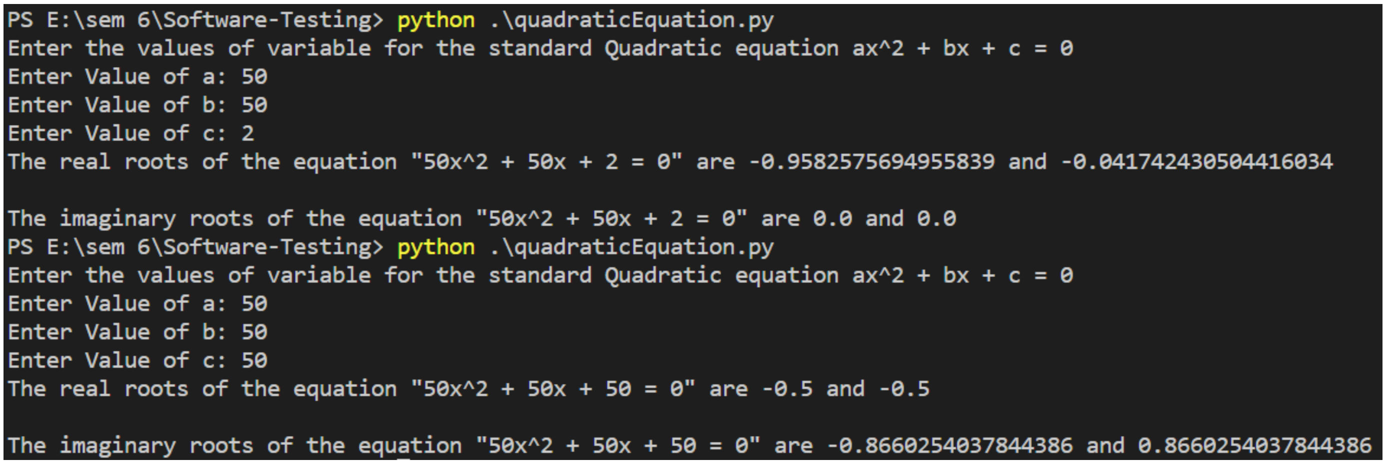
Source Code:

```
import cmath

print("Enter the values of variable for the standard Quadratic equation ax^2 + bx
+ c = 0")
a = int(input("Enter Value of a: "))
b = int(input("Enter Value of b: "))
c = int(input("Enter Value of c: "))

sqrt_d = cmath.sqrt((b**2) - (4*a*c))

root1 = (-b - sqrt_d)/(2*a)
root2 = (-b + sqrt_d)/(2*a)

print('The real roots of the equation "{0}x^2 + {1}x + {2} = 0" are {3} and
{4}\n'.format(a, b, c, root1.real, root2.real))

print('The imaginary roots of the equation "{0}x^2 + {1}x + {2} = 0" are {3} and
{4}'.format(a, b, c, root1.imag, root2.imag))
```

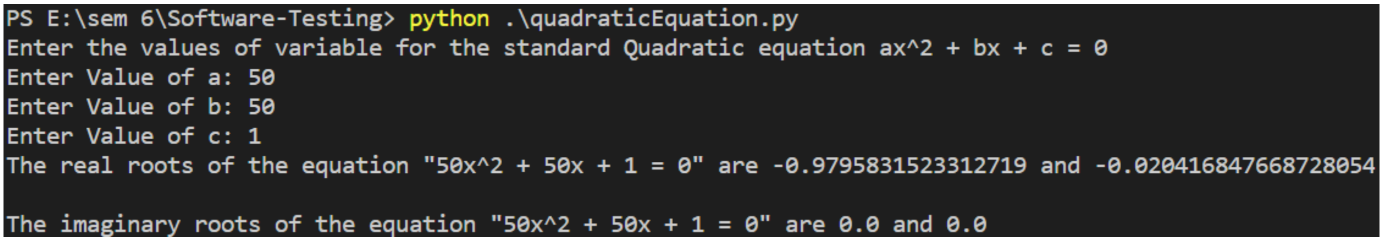
Output:



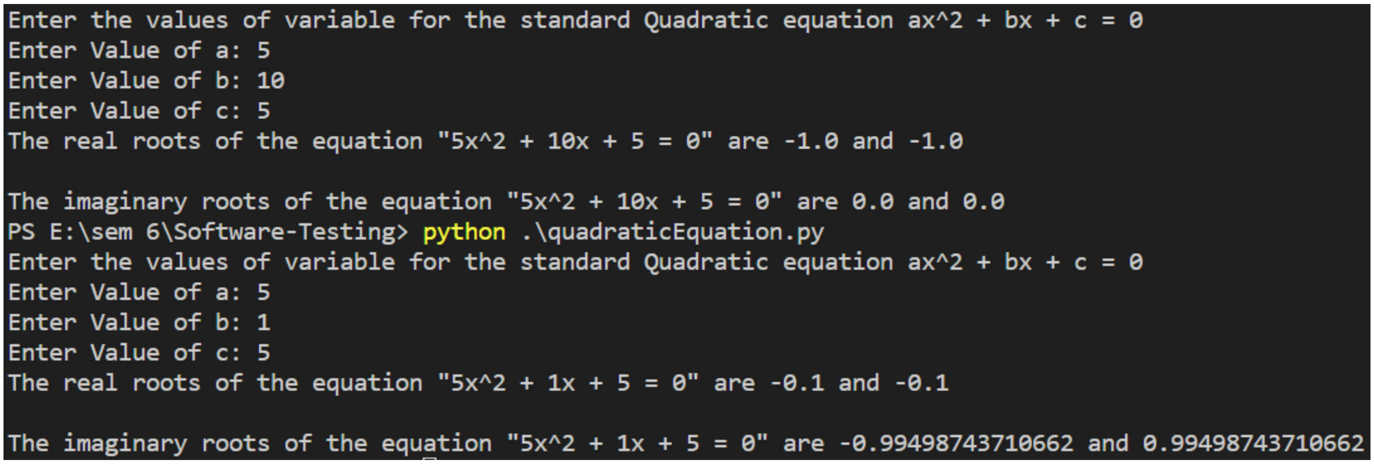
```
PS E:\sem 6\Software-Testing> python .\quadraticEquation.py
Enter the values of variable for the standard Quadratic equation ax^2 + bx + c = 0
Enter Value of a: 1
Enter Value of b: 5
Enter Value of c: 6
The real roots of the equation "1x^2 + 5x + 6 = 0" are -3.0 and -2.0

The imaginary roots of the equation "1x^2 + 5x + 6 = 0" are 0.0 and 0.0
```



```
PS E:\sem 6\Software-Testing> python .\quadraticEquation.py
Enter the values of variable for the standard Quadratic equation ax^2 + bx + c = 0
Enter Value of a: 50
Enter Value of b: 50
Enter Value of c: 2
The real roots of the equation "50x^2 + 50x + 2 = 0" are -0.9582575694955839 and -0.041742430504416034

The imaginary roots of the equation "50x^2 + 50x + 2 = 0" are 0.0 and 0.0
PS E:\sem 6\Software-Testing> python .\quadraticEquation.py
Enter the values of variable for the standard Quadratic equation ax^2 + bx + c = 0
Enter Value of a: 50
Enter Value of b: 50
Enter Value of c: 50
The real roots of the equation "50x^2 + 50x + 50 = 0" are -0.5 and -0.5

The imaginary roots of the equation "50x^2 + 50x + 50 = 0" are -0.8660254037844386 and 0.8660254037844386
```



```
PS E:\sem 6\Software-Testing> python .\quadraticEquation.py
Enter the values of variable for the standard Quadratic equation ax^2 + bx + c = 0
Enter Value of a: 50
Enter Value of b: 50
Enter Value of c: 1
The real roots of the equation "50x^2 + 50x + 1 = 0" are -0.9795831523312719 and -0.020416847668728054

The imaginary roots of the equation "50x^2 + 50x + 1 = 0" are 0.0 and 0.0
```



```
Enter the values of variable for the standard Quadratic equation ax^2 + bx + c = 0
Enter Value of a: 5
Enter Value of b: 10
Enter Value of c: 5
The real roots of the equation "5x^2 + 10x + 5 = 0" are -1.0 and -1.0

The imaginary roots of the equation "5x^2 + 10x + 5 = 0" are 0.0 and 0.0
PS E:\sem 6\Software-Testing> python .\quadraticEquation.py
Enter the values of variable for the standard Quadratic equation ax^2 + bx + c = 0
Enter Value of a: 5
Enter Value of b: 1
Enter Value of c: 5
The real roots of the equation "5x^2 + 1x + 5 = 0" are -0.1 and -0.1

The imaginary roots of the equation "5x^2 + 1x + 5 = 0" are -0.99498743710662 and 0.99498743710662
```

Test Cases:

Boundary Value analysis: The basic idea of boundary value analysis is to use input variable at their manimum, just above manimum, normal value, just below maximum and maximum.

In this program, we consider the value as 0 (minimum), 1(just above minimum), 50 (normal), 99 (just below maximum) and 100 (maximum).

Maximum of 4n+1 test cases

Test	Input Data			Expected Output	Actual Output	Pass/ Fail
Case ID	a	b	c			
1	1	50	50	Real Roots	Real Roots	Pass
2	2	50	50	Real Roots	Real Roots	Pass
3	50	50	50	Imaginary Roots	Imaginary Roots	Pass
4	99	50	50	Imaginary Roots	Imaginary Roots	Pass
5	100	50	50	Imaginary Roots	Imaginary Roots	Pass

6	**50**	**1**	**50**	**Imaginary Roots**	**Imaginary Roots**	**Pass**
7	**50**	**2**	**50**	**Imaginary Roots**	**Imaginary Roots**	**Pass**
8	**50**	**99**	**50**	**Imaginary Roots**	**Imaginary Roots**	**Pass**
9	**50**	**100**	**50**	**Real and Equal Roots**	**Real and Equal Roots**	**Pass**
10	**50**	**50**	**1**	**Real Roots**	**Real Roots**	**Pass**
11	**59**	**50**	**2**	**Real Roots**	**Real Roots**	**Pass**
12	**50**	**50**	**99**	**Imaginary Roots**	**Imaginary Roots**	**Pass**
13	**50**	**50**	**100**	**Imaginary Roots**	**Imaginary Roots**	**Pass**

Robust Testing: The term 'robust' is synonymous with strength. So robustness testing is the way to assess the quality of a software product. It the process of verifying whether a software system performs well under stress conditions or not.

The method of carrying out robustness testing follows a set of conventions. A set of invalid inputs or odd/stressful environment is set up. Sometimes it so happens that on providing certain inputs the program may crash . It becomes significant to capture those errors and rectify it in accordance with the requirement specifications.

Hence suitable test cases are developed to perform testing in an appropriate test environment. Total number of test cases generated in robust testing are **6*N +1** due to min-1 , min , min+1 , mid , max -1 , max and max+1.

Robustness testing ensures that a software system qualifies as the end product for which it was meant for, hence serving the right purpose. As we know that a complete software system comprises of various components, such kind of testing ensures reducing cost and time required for efficient delivery of a software system.

So robustness testing is carried out somewhat like this- a combination of valid and invalid inputs is passed to the system and checked for the performance . So a system is tested and validated under different conditions.

Maximum of 6n+1 test cases

Test Case Id	Input Data			Expected Output	Actual Output	Pass/ Fail
	a	b	c			
1	0	50	50	Invalid input	Invalid input	Pass
2	1	50	50	Real Roots	Real Roots	Pass
3	2	50	50	Real Roots	Real Roots	Pass
4	50	50	50	Imaginary Roots	Imaginary Roots	Pass
5	99	50	50	Imaginary Roots	Imaginary Roots	Pass
6	100	50	50	Imaginary Roots	Imaginary Roots	Pass
7	101	50	50	Invalid input	Invalid input	Pass
8	50	0	50	Invalid inputs	Invalid inputs	Pass
9	50	1	50	Imaginary roots	Imaginary roots	Pass
10	50	2	50	Imaginary roots	Imaginary roots	Pass
11	50	99	50	Imaginary roots	Imaginary roots	Pass
12	50	100	50	Equal roots	Equal roots	Pass
13	50	101	50	Invalid inputs	Invalid inputs	Pass
14	50	50	0	Invalid inputs	Invalid inputs	Pass
15	50	50	1	Real Roots	Real Roots	Pass

16	50	50	2	Real Roots	Real Roots	Pass
17	50	50	99	Imaginary Roots	Imaginary Roots	Pass
18	50	50	100	Imaginary Roots	Imaginary Roots	Pass
19	50	50	101	Invalid Inputs	Invalid inputs	Pass

Viva Questions

Q1. What is the difference BVA and Robustness Testing?

In BVA, we remain within the legitimate boundary of our range i.e. for testing we consider values like (min, min+, nom, max-, max) whereas in Robustness testing, we try to cross these legitimate boundaries as well.

Q2. What is Robustness Testing? Explain

Robustness testing is any quality assurance methodology focused on testing the robustness of software. That is the degree to which a system or component can function correctly in the presence of invalid inputs or stressful environmental conditions.

Q3. What kind of input do we need from the end user to begin proper testing?

The product has to be used by the user. He is the most important person as he has more interest than anyone else in the project. It includes Acceptance Plan, Requirement Documents, Risk Analysis, Live Data and Test Suite.

Q4. How many numbers of test cases are there in Robustness Testing?

Total Number of Test Cases = **6*N + 1** where N is the number of input variables.

Q5: How Many types of Black box Testing are there?

There are many types of Black Box Testing but the following are the prominent ones -

1 **Functional testing -** This black box testing type is related to the functional requirements of a system; it is done by software testers.

2 **Non-functional testing -** This type of black box testing is not related to testing of specific functionality, but non-functional requirements such as performance, scalability, usability.

3 **Regression testing -** Regression Testing is done after code fixes, upgrades or any other system maintenance to check the new code has not affected the existing code.