**EXPERIMENT 4**

**Aim :**

**(a)** Write a Program in C/C++ to determine the area of Circle, Triangle, Square and Rectangle. Values may be from the interval [1, 100] and perform Equivalence Class Testing.

**Algorithm :**

* Take inputs from the user according to Polygon.
* Check whether they lie in the given interval.
* If the condition is false, stop the program and exit.
* If the condition is true, calculate the area of Polygon.
  + If Circle, ***area = π \* radius2***
  + If Triangle, ***area = ½ \* base \* height***
  + If Square, ***area = side2***
  + If Rectangle, ***area = length \* breadth***
* Perform equivalence class testing accordingly.

**Code :**

#include <iostream>

using namespace std;

float circle()

{

    float r;

    cout << "Enter the Radius of Circle (r) : ";

    cin >> r;

    if (r < 1 || r > 100)

    { cout << "Out of Range";

        return 0; }

    float area = 3.14 \* r \* r;

    return area;

}

float triangle()

{

    float b, h;

    cout << "Enter the Base and Height of Triangle (b, h) : ";

    cin >> b >> h;

    if (b < 1 || b > 100 || h < 1 || h > 100)

    { cout << "Out of Range";

        return 0; }

    float area = 0.5 \* b \* h;

    return area;

}

float square()

{ float s;

    cout << "Enter the Side of Square (s) : ";

    cin >> s;

    if (s < 1 || s > 100)

    { cout << "Out of Range";

        return 0; }

    float area = s \* s;

    return area; }

float rectangle()

{ float l, b;

    cout << "Enter the Length and Breadth of Rectangle (l, b) : ";

    cin >> l >> b;

    if (l < 1 || l > 100 || b < 1 || b > 100)

    { cout << "Out of Range";

        return 0; }

    float area = l \* b;

    return area; }

int main()

{ int ch;

    float area;

    cout << "Area of : \n 1. Circle \n 2. Square \n 3. Triangle

\n 4. Rectangle";

    cout << "\nEnter Choice : ";

    cin >> ch;

    switch (ch)

    {

    case 1:

        area = circle();

        cout << "Area : " << area;

        break;

    case 2:

        area = square();

        cout << "Area : " << area;

        break;

    case 3:

        area = triangle();

        cout << "Area : " << area;

        break;

    case 4:

        area = rectangle();

        cout << "Area : " << area;

        break;

    default:

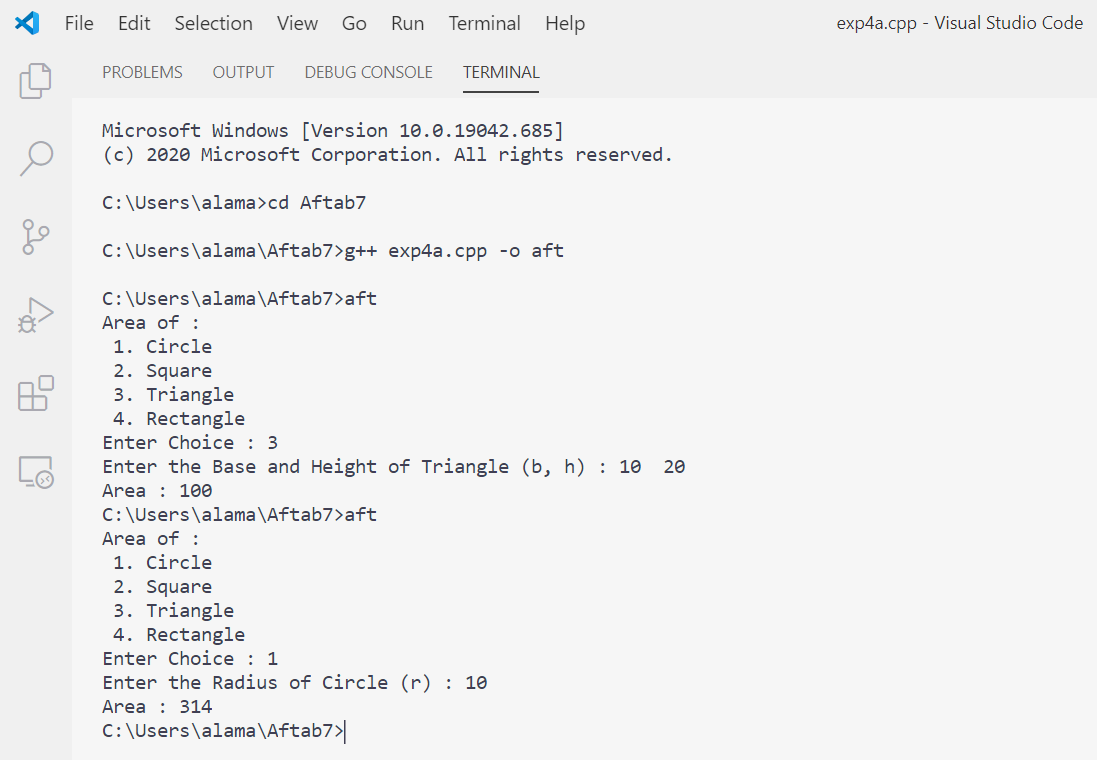
        cout << "Wrong Choice";

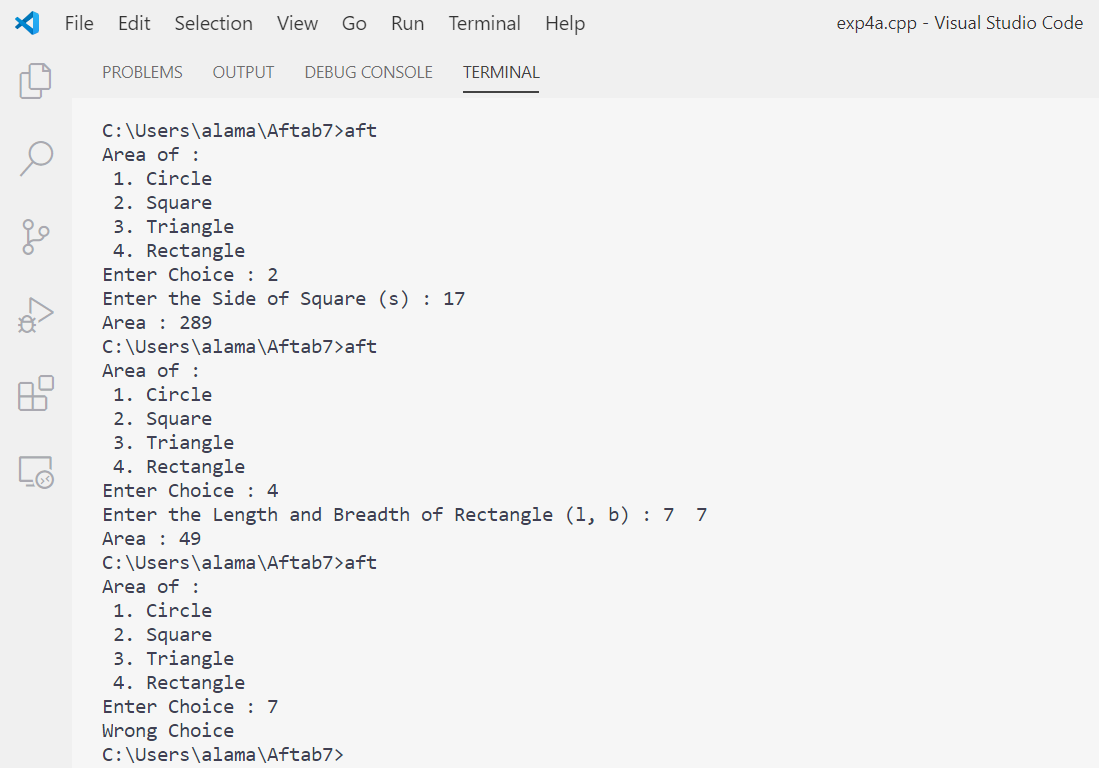
    }

    return 0;

}

**Output Screenshots :**





**Equivalence Class Testing :**

***Range :*** R [1, 200]

**Case 1 : CIRCLE**

***Input Domain : Output Domain :***

I1 = {r: r < = 0} O1 = {: Circle if 1 < = r < = 200}

I2 = {r: r > 200} O2 = {: Not a Circle if r < = 0}

I3 = {r: 1 < = r < = 200}

**Case 2 : SQUARE**

***Input Domain : Output Domain :***

I1 = {s: s < = 0} O1 = {: Square if s > 0}

I2 = {s: s > 200} O2 = {: Not a Square if s < = 0}

I3 = {s: 1 < = s < = 200}

**Case 3 : TRIANGLE**

***Input Domain : Output Domain :***

I1 = {h: h < = 0} O1 = {: Triangle if h > 0, b > 0}

I2 = {h: h > 200} O2 = {: Not a Triangle if h < = 0, b < = 0}

I3 = {h: 1 < = h < = 200}

I4 = {b: b < = 0}

I5 = {b: b > 200}

I6 = {b: 1 < = b < = 200}

**Case 4 : RECTANGLE**

***Input Domain : Output Domain :***

I1 = {l: l < = 0} O1 = {: Rectangle if l > 0, b > 0}

I2 = {l: l > 200} O2 = {: Not a Rectangle if l < = 0, b < = 0}

I3 = {l: 1 < = l < = 200}

I4 = {b: b < = 0}

I5 = {b: b > 200}

I6 = {b: 1 < = b < = 200}

**Circle Test Cases :**

|  |  |  |
| --- | --- | --- |
| **Test Case** | **r** | **Expected Output** |
| 1 | 0 | Invalid Input |
| 2 | 100 | 31400 |
| 3 | 201 | Invalid Input |

**Square Test Cases :**

|  |  |  |
| --- | --- | --- |
| **Test Case** | **s** | **Expected Output** |
| 1 | 0 | Invalid Input |
| 2 | 100 | 10000 |
| 3 | 201 | Invalid Input |

**Triangle Test Cases :**

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case** | **h** | **b** | **Expected Output** |
| 1 | 0 | 100 | Invalid Input |
| 2 | 100 | 100 | 5000 |
| 3 | 201 | 100 | Invalid Input |
| 4 | 100 | 0 | Invalid Input |
| 5 | 100 | 100 | 5000 |
| 6 | 100 | 201 | Invalid Input |

**Rectangle Test Cases :**

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case** | **l** | **b** | **Expected Output** |
| 1 | 0 | 100 | Invalid Input |
| 2 | 100 | 100 | 10000 |
| 3 | 201 | 100 | Invalid Input |
| 4 | 100 | 0 | Invalid Input |
| 5 | 100 | 100 | 10000 |
| 6 | 100 | 201 | Invalid Input |

**Aim :**

**(b)** Write a Program in C/C++ to determine the type of Triangle that is Equilateral, Isosceles, Scalene or Not a Triangle. Values may be from the interval [1, 100] and perform Decision Table Based Testing and Equivalence Class Testing.

**Algorithm :**

* Take 3 inputs from the user for the sides of the Triangle.
* Check whether they lie in the given interval.
* If the condition is false, stop the program and exit.
* If the condition is true, check the type of Triangle.
  + If all three sides are equal, Equilateral Triangle.
  + If any two sides are equal, Isosceles Triangle.
  + If all three sides are different, Scalene Triangle.
* Make decision table according to the output.
* Perform equivalence class testing accordingly.

**Code :**

#include <iostream>

using namespace std;

int main()

{

    int a, b, c;

    cout << "Enter the Sides of Triangle (a, b, c) : ";

    cin >> a >> b >> c;

    if (a < 1 || a > 100 || b < 1 || b > 100 || c < 1 || c > 100)

        cout << "Out of Range" << endl;

    else if ((a < b + c) && (b < a + c) && (c < a + b))

    {

        if ((a == b) && (b == c))

            cout << "Equilateral Triangle" << endl;

        else if ((a != b) && (b != c) && (c != a))

            cout << "Scalene Triangle" << endl;

        else

            cout << "Isosceles Triangle" << endl;

    }

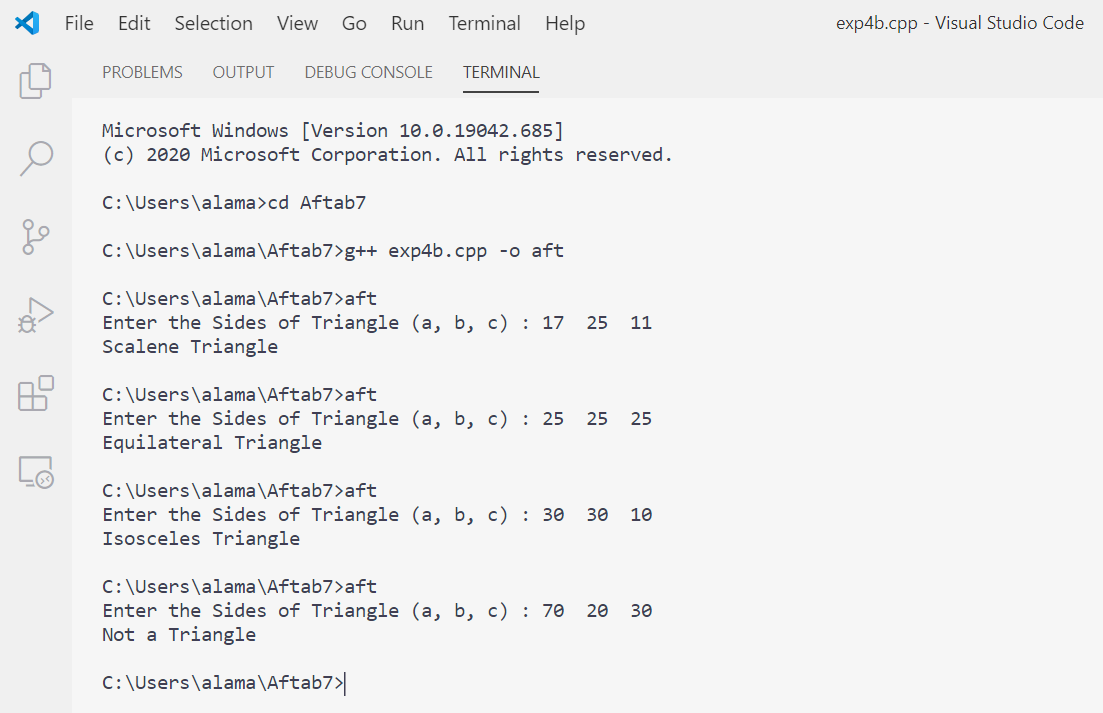
    else

        cout << "Not a Triangle" << endl;

    return 0;

}

**Output Screenshot :**



**Decision Table Based Testing :**

***Range :*** R [1, 100]

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| c1: a < b + c ? | **F** | **T** | **T** | **T** | **T** | **T** | **T** | **T** | **T** | **T** | **T** |
| c2: b < a + c ? | **—** | **F** | **T** | **T** | **T** | **T** | **T** | **T** | **T** | **T** | **T** |
| c3: c < a + b ? | **—** | **—** | **F** | **T** | **T** | **T** | **T** | **T** | **T** | **T** | **T** |
| c4: a = b ? | **—** | **—** | **—** | **T** | **T** | **T** | **T** | **F** | **F** | **F** | **F** |
| c5: a = c ? | **—** | **—** | **—** | **T** | **T** | **F** | **F** | **T** | **T** | **F** | **F** |
| c6: b = c ? | **—** | **—** | **—** | **T** | **F** | **T** | **F** | **T** | **F** | **T** | **F** |
| a1: Not a Triangle | **x** | **x** | **x** |  |  |  |  |  |  |  |  |
| a2: Scalene Triangle |  |  |  |  |  |  |  |  |  |  | **x** |
| a3: Isosceles Triangle |  |  |  |  |  | **x** |  |  | **x** | **x** |  |
| a4: Equilateral Triangle |  |  |  | **x** |  |  |  |  |  |  |  |
| a5: Impossible |  |  |  |  | **x** | **x** |  | **x** |  |  |  |

**Corresponding Test Cases :**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Case ID** | **a** | **b** | **c** | **Expected Output** |
| DT1 | 4 | 1 | 2 | Not a Triangle |
| DT2 | 1 | 4 | 2 | Not a Triangle |
| DT3 | 1 | 2 | 4 | Not a Triangle |
| DT4 | 5 | 5 | 5 | Equilateral Triangle |
| DT5 | ? | ? | ? | Impossible |
| DT6 | ? | ? | ? | Impossible |
| DT7 | 2 | 2 | 3 | Isosceles Triangle |
| DT8 | ? | ? | ? | Impossible |
| DT9 | 2 | 3 | 2 | Isosceles Triangle |
| DT10 | 3 | 2 | 2 | Isosceles Triangle |
| DT11 | 3 | 4 | 5 | Scalene Triangle |

**Equivalence Class Testing :**

***Range :*** R [1, 100]

***Input Domain :***

I1 = {0 < a < = 10} I11 = {a = b, b ! = c}

I2 = {a < 0} I12 = {b = c, c ! = a}

I3 = {a > 10} I13 = {a = c, c ! = b}

I4 = {0 < b < = 10} I14 = {a ! = b ! = c}

I5 = {b < 0} I15 = {a + b = c}

I6 = {b > 10} I16 = {a + b < c}

I7 = {0 < c < = 10} I17 = {b + c = a}

I8 = {c < 0} I18 = {b + c < a}

I9 = {c > 10} I19 = {c + a = b}

I10 = {a = b = c} I20 = {c + a > b}

***Output Domain :***

O1 = Not a Triangle

O2 = Equilateral Triangle

O3 = Isosceles Triangle

O4 = Scalene Triangle

**Test Cases :**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case** | **a** | **b** | **c** | **Expected Output** | **Actual Output** |
| O1 | 10 | 5 | 5 | Not a Triangle | Not a Triangle |
| O2 | 5 | 5 | 5 | Equilateral Triangle | Equilateral Triangle |
| O3 | 1 | 5 | 5 | Isosceles Triangle | Isosceles Triangle |
| O4 | 10 | 9 | 5 | Scalene Triangle | Scalene Triangle |
| I1 | 5 | 5 | 5 | Equilateral Triangle | Equilateral Triangle |
| I2 | 0 | 5 | 5 | Invalid Input | Invalid Input |
| I3 | 11 | 5 | 5 | Invalid Input | Invalid Input |
| I4 | 5 | 5 | 5 | Equilateral Triangle | Equilateral Triangle |
| I5 | 5 | 0 | 5 | Invalid Input | Invalid Input |
| I6 | 5 | 11 | 5 | Invalid Input | Invalid Input |
| I7 | 5 | 5 | 5 | Equilateral Triangle | Equilateral Triangle |
| I8 | 5 | 5 | 0 | Invalid Input | Invalid Input |
| I9 | 5 | 5 | 11 | Invalid Input | Invalid Input |
| I10 | 5 | 5 | 5 | Equilateral Triangle | Equilateral Triangle |
| I11 | 5 | 5 | 1 | Isosceles Triangle | Isosceles Triangle |
| I12 | 1 | 5 | 5 | Isosceles Triangle | Isosceles Triangle |
| I13 | 5 | 1 | 5 | Isosceles Triangle | Isosceles Triangle |
| I14 | 9 | 5 | 10 | Scalene Triangle | Scalene Triangle |
| I15 | 5 | 5 | 10 | Not a Triangle | Not a Triangle |
| I16 | 1 | 5 | 10 | Not a Triangle | Not a Triangle |
| I17 | 10 | 5 | 5 | Not a Triangle | Not a Triangle |
| I18 | 10 | 5 | 1 | Not a Triangle | Not a Triangle |
| I19 | 5 | 10 | 5 | Not a Triangle | Not a Triangle |
| I20 | 5 | 10 | 1 | Not a Triangle | Not a Triangle |