**Método type ()**

Cantidad de Nodos: 9

Cantidad de Aristas: 12

Complejidad Ciclomática: Nro. Aristas – Nro. Nodos + 2 = 12 – 9 + 2 = 5

Nro. Condiciones + 1 = 5

**Casos de Prueba**

1. Escaleno Válido – Camino 1,6,8,9,4

Datos de Ingreso:

Lado1: 2

Lado2: 3

Lado3: 4

Pasa por 1 (consulta que Lado1 == Lado2) como es falso pasa a 6 (Lado1 == Lado3) Como es falso pasa a la condición 8 (Lado2 == Lado3) pasa a 9 (return “Es triangulo Escaleno”) y sale por 4 (Fin)

1. Equilátero Valido – Camino 1,2,3,4

Datos de Ingreso:

Lado1: 2

Lado2: 2

Lado3: 2

Pasa por 1 (consulta que Lado1 == Lado2) al ser verdadero pasa a 2 (Lado1 == Lado3)

al ser verdadero pasa a 3 (return “Es triangulo Equilátero”) y sale por 4 (Fin).

1. Isósceles Valido - Camino 1,6,7,4

Datos de Ingreso:

Lado1: 2

Lado2: 3

Lado3: 2

Pasa por 1 (consulta que Lado1 == Lado2) como es falso pasa a 6 (Lado1 == Lado3) al ser verdadero pasa a 7 (return “Es triangulo Isósceles”) y sale por 4 (Fin).

**Método isTriangulo ()**

Cantidad de Nodos: 6

Cantidad de Aristas: 8

Complejidad Ciclomática: Nro. Aristas – Nro. Nodos + 2 = 8 – 6 + 2 =4

Nro. Condiciones + 1 = 4

**Casos de Prueba**

1. Es triangulo

Datos de ingreso:

Lado1: 2

Lado2: 3

Lado3: 2

Pasa por 1 (Lado1 + Lado2 > Lado3) como es verdadero pasa por 2 (Lado1 + Lado3 > Lado2) como es verdadero pasa por 3 ( Lado2 + Lado3 > Lado1) y pasa por 4 (return True) y sale por 5 (Fin).

1. No es Triangulo

Datos de ingreso:

Lado1: 2

Lado2: 1

Lado3: 1

Pasa por 1 (Lado1 + Lado2 > Lado3) como es verdadero pasa por 2 (Lado1 + Lado3 > Lado2) como es verdadero pasa por 3 ( Lado2 + Lado3 > Lado1) y pasa por 6 (return False) y sale por 5 (Fin).

Metodo mostrarResultado()