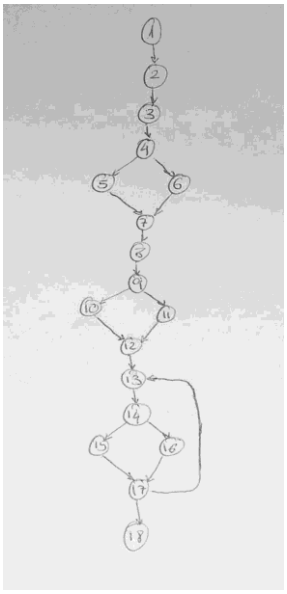
EJERCICIO 3.) COMPLEJIDAD CICLOMÁTICA

La complejidad ciclomática de la clase "main" se puede calcular utilizando la fórmula:

V(G) = E - N + 2

Donde M es la complejidad ciclomática, E es el número de aristas o caminos del grafo de control de flujo, y N es el número de nodos del grafo de control de flujo.

En este caso, el grafo de control de flujo para la clase "main" es el siguiente:



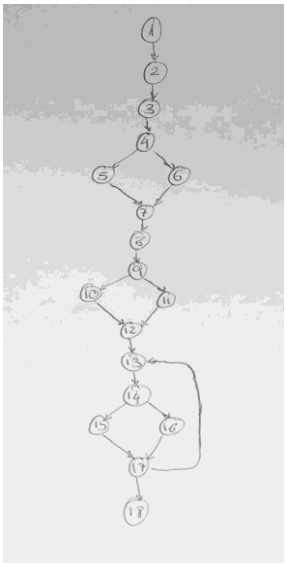
Leyenda:

1. Inicio
2. Madr
3. Ppp
4. If
8. Ppp
9. If
13. For
14. If
18. Fin

Podemos ver que hay un total de 21 aristas y 18 nodos, por lo que la complejidad ciclomática es:

V(G) = 21 - 18 + 2 = 5

Por lo tanto, hay 5 caminos básicos en la clase "main":



Para validar estos caminos básicos, se pueden crear pruebas unitarias que cubran todos los caminos posibles, tal como se detalló en la respuesta anterior:

```
@Test
public void testMadr1() {
      System.out.println("madr");
      int a = 6;
      int b = 4;
      int c = 2;
      int expResult = 2;
      int result = JavaApplication2.madr(a, b, c);
      assertEquals(expResult, result);

   }
```

En test pasará en los dos primeros if por la izquierda, ya que 4 es mayor que 2 y 6 es mayor que 2, el for solamente lo realiza una vez porque 'mmm' es igual a 2 y en el if pasará por la izquierda, ya que 'i', en este caso 2, divide a 'a', 'b' y 'c'.

```
   @Test
   public void testMadr2() {
      System.out.println("madr");
      int a = 6;
      int b = 2;
      int c = 4;
      int expResult = 2;
      int result = JavaApplication2.madr(a, b, c);
      assertEquals(expResult, result);

   }
```

En test pasará en el primer if por la derecha, ya que 2 no es mayor que 4, en el segundo if por la izquierda, porque 6 es mayor que 2, el for solamente lo realiza una vez porque 'mmm' es igual a 2 y en el if pasará por la izquierda, ya que 'i', en este caso 2, divide a 'a', 'b' y 'c'.

```
   @Test
   public void testMadr3() {
      System.out.println("madr");
      int a = 2;
      int b = 6;
      int c = 4;
      int expResult = 2;
      int result = JavaApplication2.madr(a, b, c);
      assertEquals(expResult, result);

   }
```

En test pasará en el primer if por la izquierda, ya que 6 es mayor que 4, en el segundo if por la derecha, porque 2 no es mayor que 4, el for solamente lo realiza una vez porque 'mmm' es igual a 2 y en el if pasará por la izquierda, ya que 'i', en este caso 2, divide a 'a', 'b' y 'c'.

```
@Test
public void testMadr4() {
    System.out.println("madr");
    int a = 6;
    int b = 3;
    int c = 2;
    int expResult = 1;
    int result = JavaApplication2.madr(a, b, c);
    assertEquals(expResult, result);

}
```

En test pasará en el primer if por la izquierda, ya que 3 es mayor que 2, en el segundo if por la izquierda, porque 6 es mayor que 2, el for lo realizará una vez porque 'mmm' es igual a 2 y en el if pasará por la derecha, ya que 'i', en este caso 2, no divide a 'a', 'b' y 'c'.

```
@Test
public void testMadr5() {
    System.out.println("madr");
    int a = 10;
    int b = 6;
    int c = 4;
    int expResult = 2;
    int result = JavaApplication2.madr(a, b, c);
    assertEquals(expResult, result);

}
```

En test pasará en el primer if por la izquierda, ya que 6 es mayor que 4, en el segundo if por la izquierda, porque 10 es mayor que 4, el for lo realizará tres veces porque 'mmm' es igual a 4 y en el if pasará por la derecha, ya que 'i', en este caso 2, divide a 'a', 'b' y 'c'.