Ejercicio 1

Aplique la técnica del camino básico al diseño del mínimo caso de prueba para probar el siguiente método. Este método estático aplica la búsqueda binaria para encontrar un carácter c en un vector de caracteres v. El método devuelve 1 si se encuentra el carácter, 0 en caso contrario. El vector de entrada está ordenado de forma ascendente teniendo en cuenta el código ASCCI.

a) Dibuja correctamente el gráfico de flujo asociado
b) Calcula la complejidad ciclomática:
   I. Indica el número de regiones
   II. Indica el número de nodos
   III. Indica el número de nodos predicado
   IV. Indica el número de aristas
c) Especifica los caminos independientes
d) Provee casos de prueba asociados a los caminos independientes

```
static public int search(char c, char []v)
        {
            int a, z, m;
            a = 0;                          1
            z = v.Length - 1;
            while (a <= z)                  2
            {
                m = (a + z) / 2;            3
                if (v[m] == c) {            4
                    return 1;               5
                }
                else if(v[m] < c)           6
                {
                    a = m + 1;              7
                }
                else
                {
                    z = m - 1;              8
                }
            }
            return 0;                       9
        }                                   10
```

Regiones = A - N + 2 = 12 - 10 + 2 = 4
Nodos = 10
Nodos predicado = 3
Aristas = 12

Caminos independientes:
1-2-9-10, c='a', v="", O=0
1-2-3-4-5-10, c='a', v="a", O=1
1-2-3-4-6-7-2-9-10, c='c', v="b", O=0
1-2-3-4-6-8-2-9-10, c='b', v="c", O=0

Ejercicio 2

Aplique la técnica del camino básico al diseño del mínimo caso de prueba para probar el siguiente método. Este método estático ordena un array de enteros de forma ascendente

a) Dibuja correctamente el gráfico de flujo asociado
b) Calcula la complejidad ciclomática:
   I. Indica el número de regiones
   II. Indica el número de nodos
   III. Indica el número de nodos predicado
   IV. Indica el número de aristas
c) Especifica los caminos independientes
d) Provee casos de prueba asociados a los caminos independientes

```
static public void sort(int[] testArray)
{
            int tempValue;
            int i = 0;                                          1
            bool isSwapped = true;
            while (isSwapped)                                   2
            {   isSwapped = false;
                i++;
                Console.Out.WriteLine("Before "+i+" iteration :");          3
                Console.Out.WriteLine("");
                for (int j = 0; j < testArray.Length - i; j++)      4   7
               {
                    if (testArray[j] > testArray[j + 1])            5
                    {
                        tempValue = testArray[j];
                        testArray[j] = testArray[j + 1];            6
                        testArray[j + 1] = tempValue;
                        isSwapped = true;
                    }
                }
            }
} 8
```

```
Regiones = A - N + 2 = 10 - 8 + 2 = 4
Nodos = 8
Nodos predicado = 3
Aristas = 10
```

```
Camino independientes:
1-2-8, Input = No es posible, Output = No es posible
1-2-3-4-2-8, Input = [], Output = []
1-2-3-4-5-7-4-2-8, Input = [1,2], Output = [1,2]
1-2-3-4-5-6-7-4-2-8, Input = [2,1], Output = [1,2]
```