

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 ASCII.

- 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;
    z = v.Length - 1;
    while (a <= z)
    {
        m = (a + z) / 2;
        if (v[m] == c) {
            return 1;
        }
        else if (v[m] < c)
        {
            a = m + 1;
        }
        else
        {
            z = m - 1;
        }
    }
    return 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 predicho
 - 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;
    bool isSwapped = true;
    while (isSwapped)
    {
        isSwapped = false;
        i++;
        Console.WriteLine("Before "+i+" iteration :");
        Console.WriteLine("");
        for (int j = 0; j < testArray.Length - i; j++)
        {
            if (testArray[j] > testArray[j + 1])
            {
                tempValue = testArray[j];
                testArray[j] = testArray[j + 1];
                testArray[j + 1] = tempValue;
                isSwapped = true;
            }
        }
    }
}
```