WHITE BOX EXERCISES

Basis Path Technique

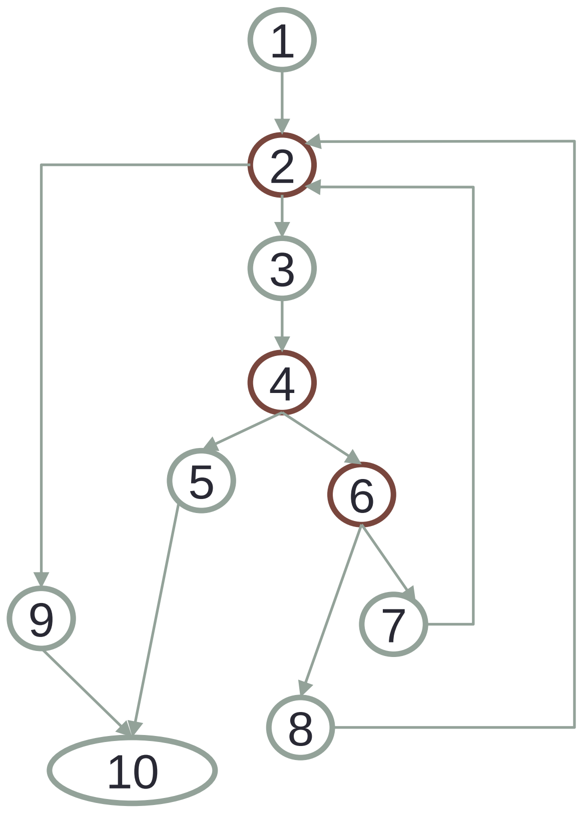
Ingeniería del Software DSIC-UPV Curso 2018-2019

Exercise 1

```
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;
        }
```

Exercise 1

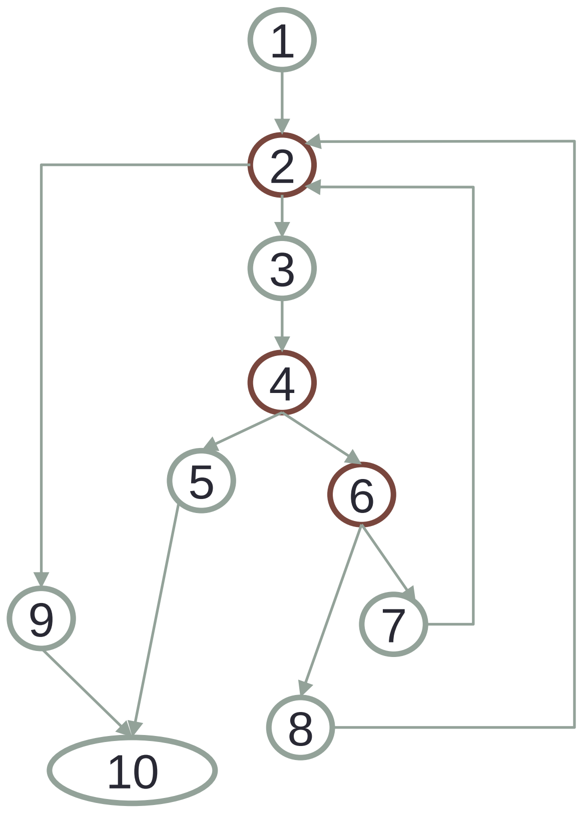
```
static public int search(char c, char []v)
        {
            int a, z, m;
        1   a = 0;
            z = v.Length - 1;
        2   while (a <= z)
            {
                m = (a + z) / 2;  3
            4   if (v[m] == c) {
                    return 1;  5
                }
        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



V(G)= 4
Areas = 4
Predicate Nodes = 3 → 3+1=4
Nodes = 10 → 12-10+2 =4
Edges = 12

Exercise 1

Path	Input	Output
{1,2,9,10} Empty string	V=“” c=‘a’	0
{1,2,3,4,5,10} In first place	V=“a” c=‘a’	1
{1,2,3,4,6,7,2,9,10} String with only one character, lower than the target	V=“a” c=‘b’	0
{1,2,3,4,6,8,2,9,10} String with only one character, higher than the target	V=“b” c=‘a’	0



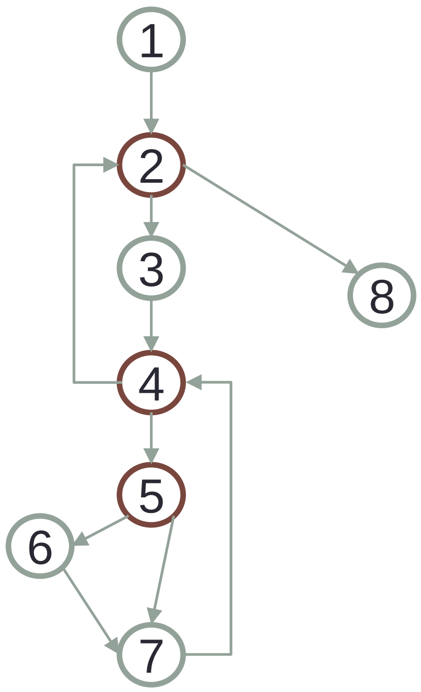
Exercise 2

```
static public void sort(int[] testArray)
        {
            int tempValue;
            int i = 0;
            bool isSwapped = true;
            while (isSwapped)
            {   isSwapped = false;
                i++;
                Console.Out.WriteLine("Before "+i+" iteration :");
                Console.Out.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;
                    }
                }
            }

        }
```

Exercise 2

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



V(G)= 4
Areas = 4
Predicate Nodes = 3 → 3-1=4
Nodes = 8 → 10-8+2 =4
Edges = 10

Exercise 2

Path	Input	Output
{1,2,8} Empty string	Not possible	Not possible
{1,2,3,4,2,8} Empty or 1 position	[ ]	Before 1 Iteration []
{1,2,3,4,5,6,7,4,2,8}*	Not Possible	Not Possible
{1,2,3,4,6,8,2,8} Two positions ordered	[1,2]	Before 1 Iteration [1,2]
*To make this path possible **{1,2,3,4,5,6,7,4,2,3,4,2,8}** Two positions ordered	[2,1]	Before 1 Iteration Before 2 Iteration [1,2]