Problema CB1. Dada la siguiente función diseñe los casos de prueba utilizando la técnica del camino básico (prueba de caja blanca).

```
public static int busquedaBinaria( int [] array, int dato) {

// implementa la búsqueda binaria de un valor (dato) en un array
// devuelve la posición en el array donde se encuentra el dato
// en caso de no encontrarse o si el array está desordenado devuelve -1

  int inicio = 0;
  int fin = array.length - 1;
     int pos;
  while (inicio <= fin) {
      pos = (inicio+fin) / 2;
      if ( array[pos] == dato )
        return pos;
      else if ( array[pos] < dato ) {
       inicio = pos+1;
                }
                else {
       fin = pos-1;
                }
             }
  return -1;
 }
```

Problema CB2. Diseñe los casos de prueba utilizando la técnica del camino básico para la siguiente función. Deberá: dibujar el grafo de flujo etiquetado, calcular la complejidad ciclomática, definir los caminos independientes y los casos de prueba para cada camino.

```
public static int contarBisiestos(int inicio, int fin){
        // devuelve el número de años bisiestos entre inicio y fin
        int numBisiestos=0;
        if (inicio > fin) {
                System.out.println("Valor de fin debe ser mayor que inicio");
                return -1;
        }
        for (int año=inicio; año <= fin; año ++){
                if ((año % 4 == 0) && ((año % 100 !=0) || (año % 400 == 0)))
                        numBisiestos++;
        }
        return numBisiestos;
}
```

Problema CB3. ValidateId es un método que devuelve true si el NIF proporcionado es válido o false en caso contrario. ValidateId hace uso del método NIFLetter, que devuelve la letra asociada a los dígitos de un NIF. Si no se le proporcionan 8 dígitos, NIFLetter lanza una excepción del tipo ArgumentException. Diseñe los casos de prueba para el método ValidateId siguiendo la técnica del camino básico de caja blanca: dibuje el grafo de flujo, calcule la complejidad ciclomática, especifique los caminos independientes y los casos de prueba asociados a cada camino. Para el diseño de los casos de prueba, puede tener en cuenta que 63099845N es un nif válido.

```
public static bool ValidateId(string id)
{
  int validSize = 9;
   if (id.Length < validSize || id.Length > validSize)
      return false;
   //checking if the last digit is the correct letter
   string digitsId = id.Substring(0, id.Length - 1);
   try
   {
     char letter = NIFLetter(digitsId);
     if (id.EndsWith("" + letter))
       return true;
     else
       return false;

   }
   catch (ArgumentException)
   {
     return false;
   }
 }
```

Problema CB4. Diseñar los casos de prueba para el siguiente fragmento de código siguiendo la técnica del camino básico (etiqueta el código, dibuja el grafo de flujo, calcula la complejidad ciclomática, especifica los caminos independientes y los casos de prueba asociados a cada camino).

```
public int Metodo(int a, int b, int c)
{
   if ((c < 100) && ((b<100) || (a> 0)))
  {
      Console.Out.WriteLine(c / a);   // puede disparar una excepción si a == 0
      Console.Out.WriteLine(c / b);   // puede disparar una excepción si b == 0
     return a*b;

  }
    else
      return c;
}
```