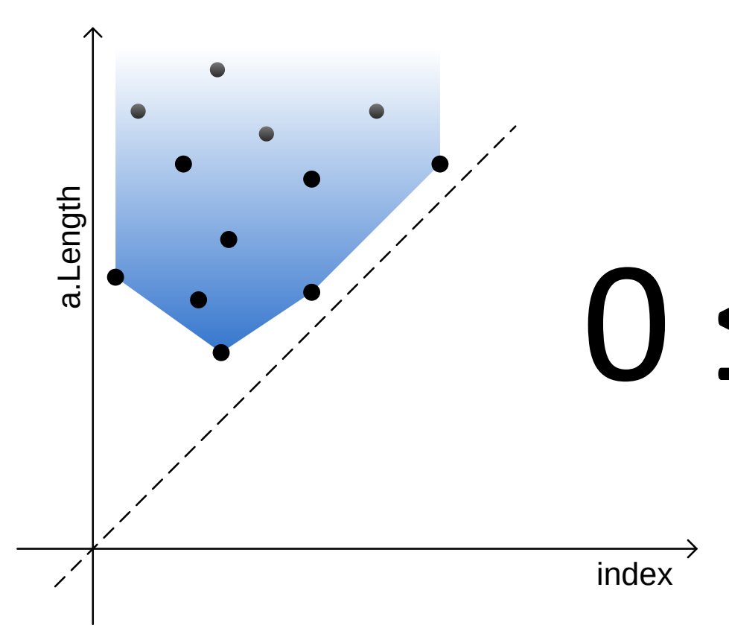
CLOUSOT

Semantic-based Static Checking for .NET

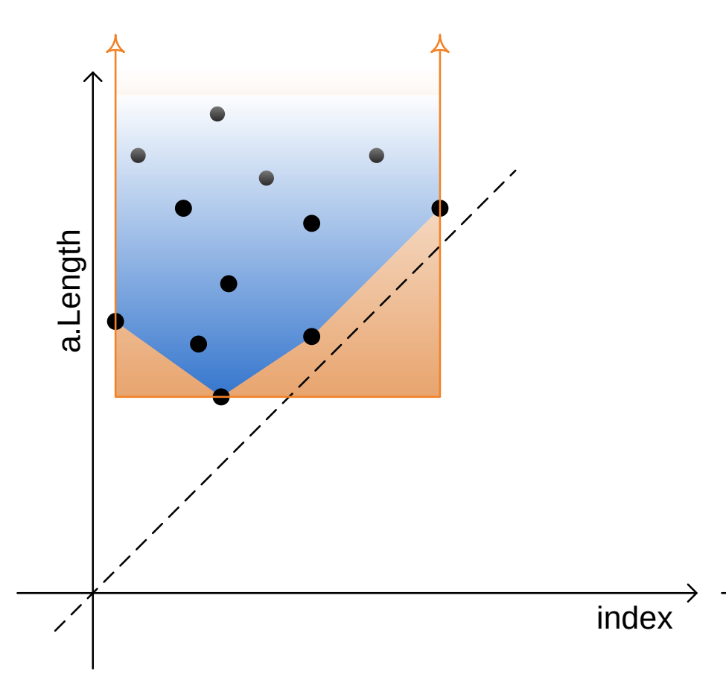
Programming Languages and Methods

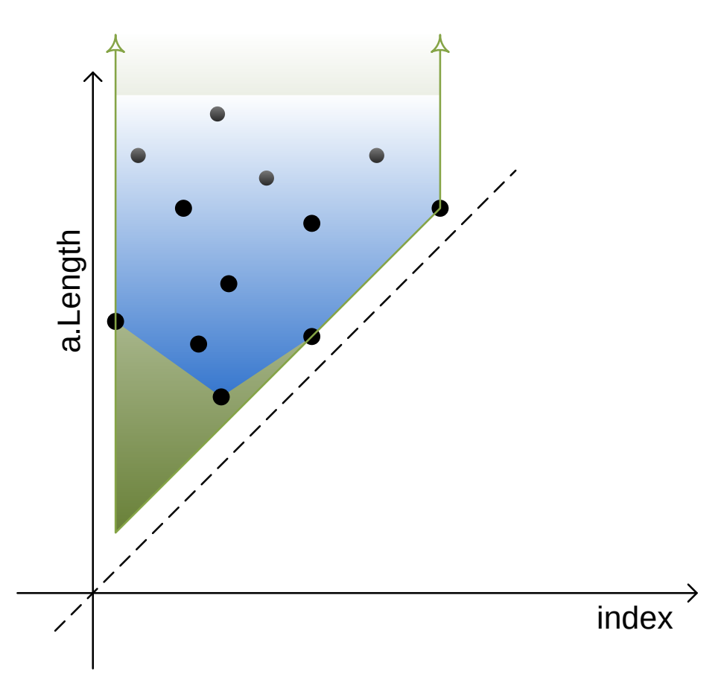
The theory

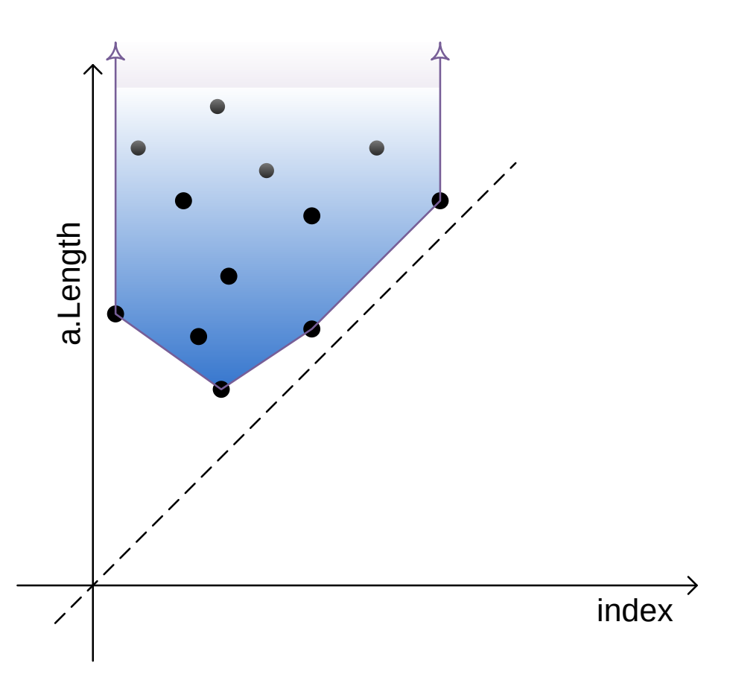


0 ≤ index < a.Length?

Abstract interpretation (Clousot) :
“Abstract the points”



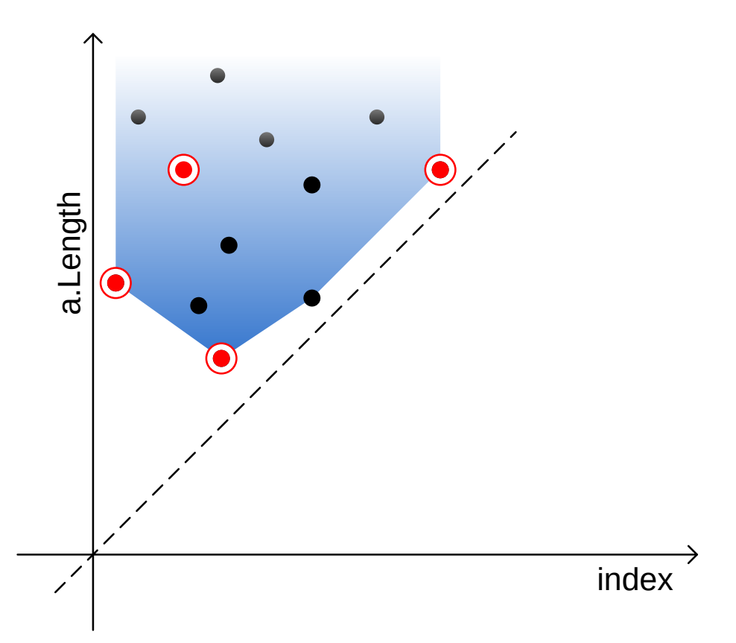


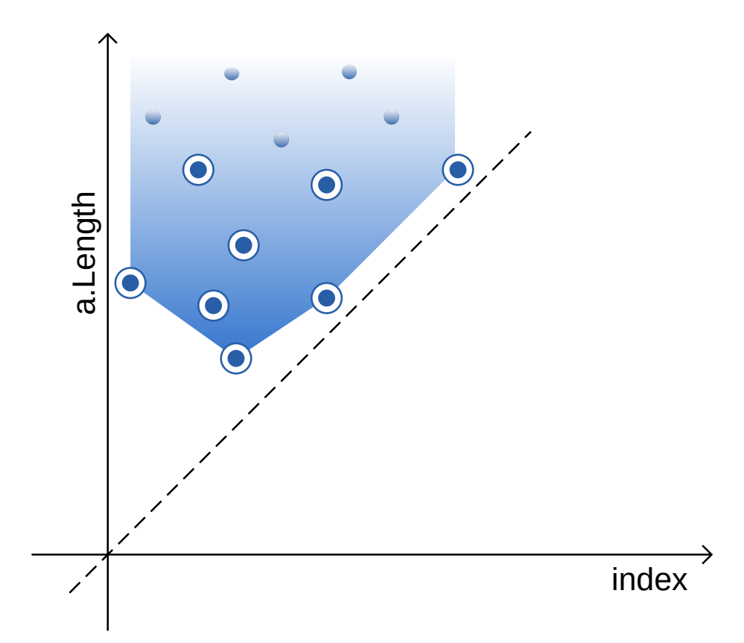


Property:	Property:	Property:
a ≤x≤ b	*a ≤x≤ b & x<y*	$\Sigma a_i\, x \le b$
Cost: O(n)	Cost: O(n^2)	Cost: O(2^n)

Testing:
“Try some points”

Model checking:
“Try all the points”



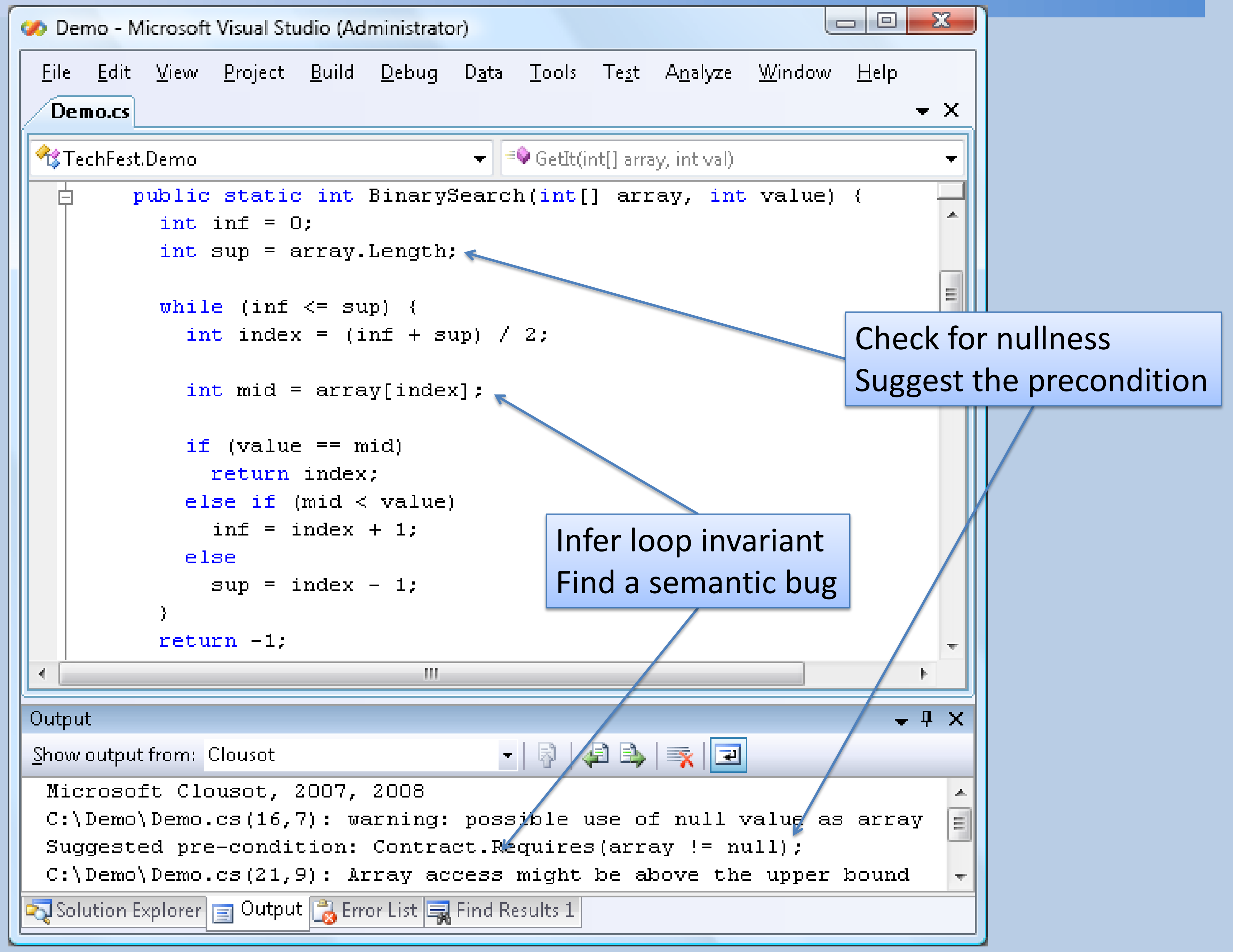


☹ Not exhaustive

☹ Infinite points

In practice

- ✓Cover all the execution paths
- ✓Tune precision/cost ratio
- ✓Precise
- ✓Scalable ☺
- ✓Focused
- ✓Language agnostic
- ✓Try it today: http://Clousot


Demo - Microsoft Visual Studio (Administrator)
File Edit View Project Build Debug Data Tools Test Analyze Window Help
Demo.cs
TechFest.Demo
GetIt(int[] array, int val)
Check for nullness
Suggest the precondition
Infer loop invariant
Find a semantic bug
Output
Show output from: Clousot
Solution Explorer
Output
Error List
Find Results 1

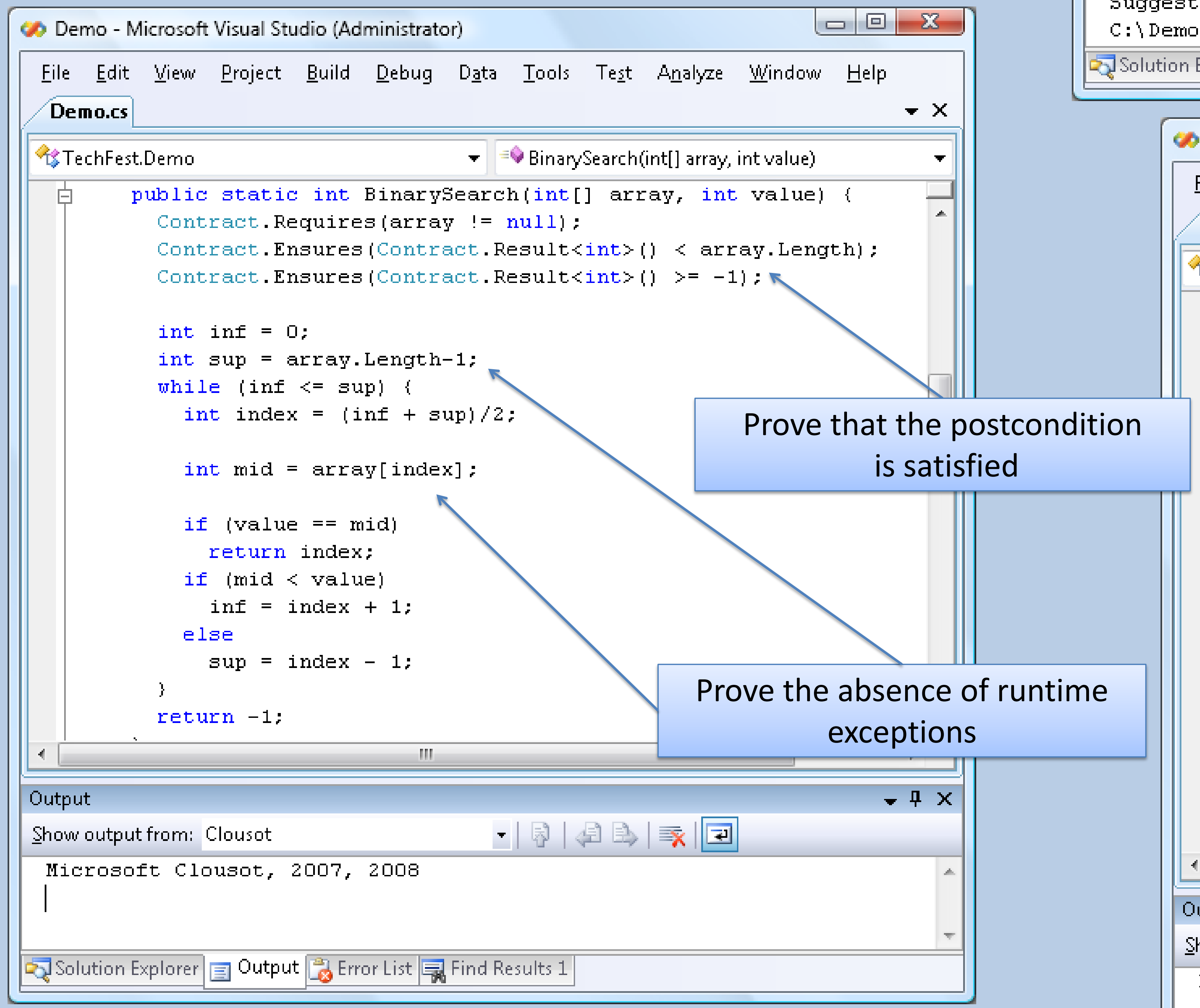
```
    public static int BinarySearch(int[] array, int value) {
      int inf = 0;
      int sup = array.Length;

      while (inf <= sup) {
        int index = (inf + sup) / 2;

        int mid = array[index];

        if (value == mid)
          return index;
        else if (mid < value)
          inf = index + 1;
        else
          sup = index - 1;
      }
      return -1;
```

```
Microsoft Clousot, 2007, 2008
C:\Demo\Demo.cs(16,7): warning: possible use of null value as array
Suggested pre-condition: Contract.Requires(array != null);
C:\Demo\Demo.cs(21,9): Array access might be above the upper bound
```


Demo - Microsoft Visual Studio (Administrator)
File Edit View Project Build Debug Data Tools Test Analyze Window Help
Demo.cs
TechFest.Demo
BinarySearch(int[] array, int value)
Prove that the postcondition is satisfied
Prove the absence of runtime exceptions
Output
Show output from: Clousot
Solution Explorer
Output
Error List
Find Results 1

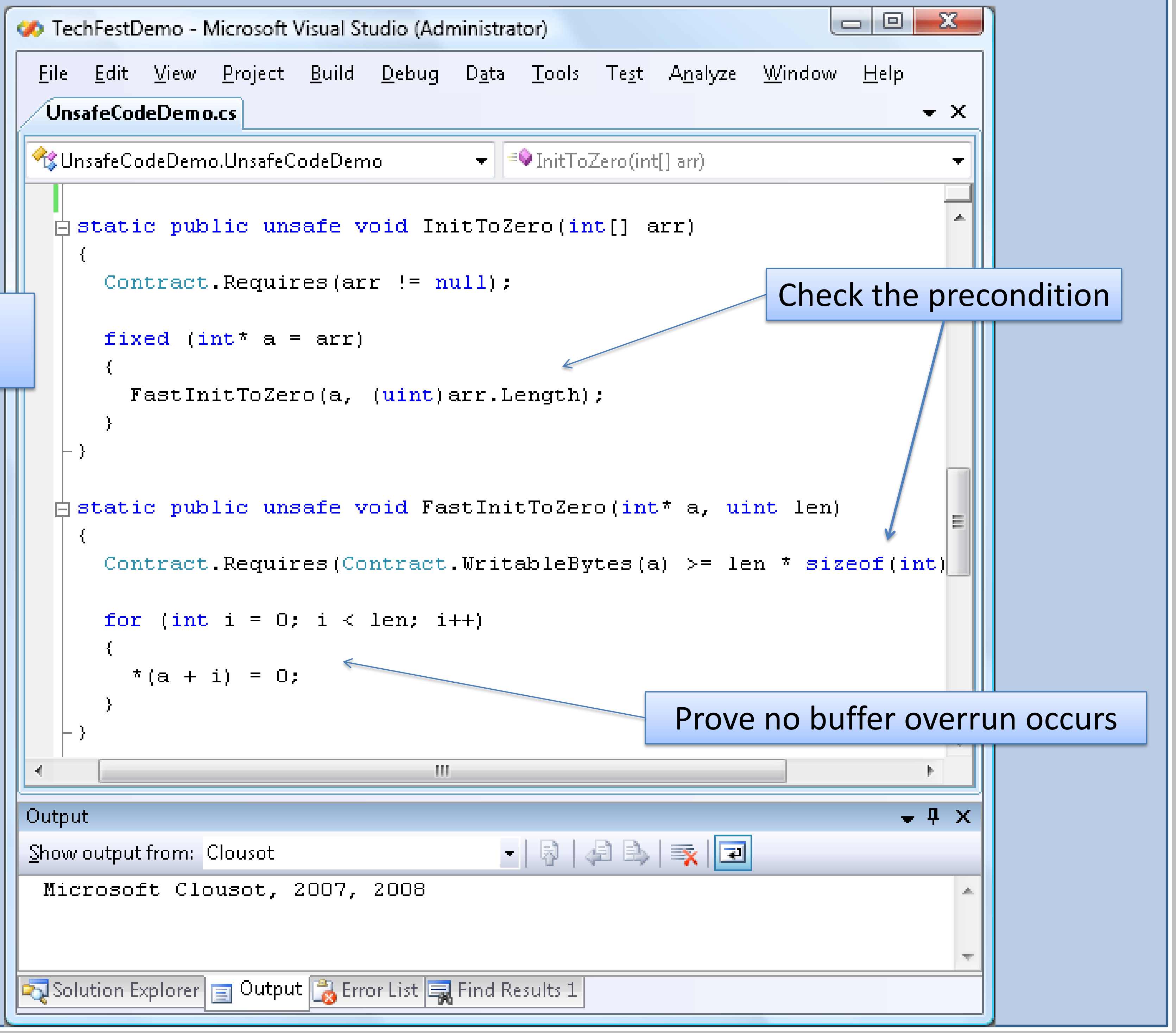
```
    public static int BinarySearch(int[] array, int value) {
      Contract.Requires(array != null);
      Contract.Ensures(Contract.Result<int>() < array.Length);
      Contract.Ensures(Contract.Result<int>() >= -1);

      int inf = 0;
      int sup = array.Length-1;
      while (inf <= sup) {
        int index = (inf + sup)/2;

        int mid = array[index];

        if (value == mid)
          return index;
        if (mid < value)
          inf = index + 1;
        else
          sup = index - 1;
      }
      return -1;
```

```
Microsoft Clousot, 2007, 2008
```


TechFestDemo - Microsoft Visual Studio (Administrator)
File Edit View Project Build Debug Data Tools Test Analyze Window Help
UnsafeCodeDemo.cs
UnsafeCodeDemo.UnsafeCodeDemo
InitToZero(int[] arr)
Check the precondition
Prove no buffer overrun occurs
Output
Show output from: Clousot
Solution Explorer
Output
Error List
Find Results 1

```
static public unsafe void InitToZero(int[] arr)
{
  Contract.Requires(arr != null);

  fixed (int* a = arr)
  {
    FastInitToZero(a, (uint)arr.Length);
  }
}

static public unsafe void FastInitToZero(int* a, uint len)
{
  Contract.Requires(Contract.WritableBytes(a) >= len * sizeof(int)

  for (int i = 0; i < len; i++)
  {
    *(a + i) = 0;
  }
}
```

```
Microsoft Clousot, 2007, 2008
```