```
method LinearSearch(input:array<int>,find:int) returns (i:int)
  requires input != null
  ensures 0 <= i ==> i < input.Length && input[i] == find
  ensures i < 0 ==> forall k :: 0 <= k < input.Length ==> input[k] != find
{
  i:=0;
  while i<input.Length
    invariant 0 <= i <= input.Length
    invariant forall k :: 0 <= k < i ==> input[k] != find
  {
    if input[i]==find
    {
      return;
    }
    i:=i+1;
  }
  i:=-1;
}
```

```
predicate sorted(a: array<int>)
  requires a != null
  reads a
{
  forall j, k :: 0 <= j < k < a.Length ==> a[j] <= a[k]
}
```

```
method BinarySearch(input: array<int>, find: int) returns (i: int)
  requires input != null && sorted(input)
  ensures 0 <= i ==> i < input.Length && input[i] == find
  ensures i < 0 ==> forall k :: 0 <= k < input.Length ==> input[k] != find
{
  var low, high := 0, input.Length;
  while low < high
    invariant 0 <= low <= high <= input.Length
    invariant forall a ::
    0 <= a < input.Length && !(low <= a < high) ==> input[a] != find
  {
    var mid := (low + high) / 2;
    if input[mid] < find
    {
      low := mid + 1;
    }
    else if find < input[mid]
    {
      high := mid;
    }
    else
    {
      return mid;
    }
  }
  return -1;
}
```

```
method main() returns (b:int)
//Testing Method
{
//Array
  var  a :=new int[2];
  a[0],a[1]:=3,4;

//Sorted
  assert sorted(a)==true;

//Search Comparison
  var search1:int;
  var search2:int;
  search1:=LinearSearch(a,4);
  search2:=BinarySearch(a,4);
  assert search1==search2;

  return 1;
}
```