```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

class Search
{

    static int[] InitArray(int size)
    {
        Random rand = new Random();
        int[] array = new int[size];
        for (int i = 0; i < array.Length; i++)
        {
            array[i] = rand.Next(0,100);
        }
        return array;
    }

    private void Show(int[] array)
    {
        Console.WriteLine("Array: ");
        for (int i = 0; i < array.Length; i++)
        {
            Console.Write(array[i] + " ");
        }
        Console.WriteLine();
        Console.WriteLine();
    }


    private int Partition(int[] array, int start, int end)
    {

        int pivot = array[end];

        int i = start;
        for (int j = start; j < end; j++)
        {

            if (array[j] <= pivot)
            {

                Swap(ref array[i], ref array[j]);
                i++;

            }
        }

        Swap(ref array[i], ref array[end]);
```

```
    return i;
}

public void QuickSort(int[] array, int start, int end)
{

    if (start < end)
    {
        int q = Partition(array, start, end);
        QuickSort(array, start, q - 1);
        QuickSort(array, q + 1, end);

    }
}

private void Swap(ref int a, ref int b)
{
    int temp = a;
    a = b;
    b = temp;
}



static int LinearSearch(int[] array, int value)
{
    int elementNotFound = -1;

    for (int i = 0; i < array.Length; i++)
    {

        if (array[i] == value)
            return i;

    }

    return elementNotFound;
}

static int BinarySearch(int[] array, int value)
{

    int i = -1;
    if (array != null)
    {
        int start = 0;
        int end = array.Length;
        int middle;
        while (start < end)
        {

            middle = (start + end) / 2;
```

```
            if (value == array[middle])
            {
                i = middle;
                break;
            }
            else if (value < array[middle])
            {
                end = middle;
            }
            else
            {
                start = middle + 1;
            }
        }
    }
    return i;
}

static int InterpolationSearch(int[] array, int value)
{

    if (array != null)
    {
        int start = 0;
        int end = array.Length - 1;
        int middle;
        while (array[start] < value && value < array[end])
        {

            middle = start + ((end - start) * (value - array[start])) / (array[end] - array[start]);
            if (array[middle] < value)
            {
                start = middle + 1;
            }
            else if (value < array[middle])
            {
                end = middle - 1;
            }
            else
            {
                return middle;
            }
        }

        if (array[start] == value)
        {
            return start;
        }
        else if (array[end] == value)
        {
            return end;
        }
```

```
            else
            {
                return -1;
            }
        }
        return -1;
    }


    public static void Main()
    {
        int size = 1000000;
        Search search = new Search();
        int[] array = InitArray(size);
        //search.Show(array);
        search.QuickSort(array, 0, array.Length - 1);
        search.Show(array);
        Console.WriteLine();
        int value = 99;
        //int value = int.Parse(Console.ReadLine());
        Console.WriteLine("value: " + value);
        DateTime time = DateTime.Now;
        int index = InterpolationSearch(array, value);
        // If element was found
        if (index != -1)
        {
            Console.WriteLine("Element found  at index " + index);
        }
        else
        {
            Console.WriteLine("Element not found.");
        }
        Console.WriteLine();
        Console.WriteLine("LinearSearch Search took: {0} sec",(DateTime.Now - time).TotalSeconds);
        Console.ReadKey();
    }
}
```