**Método de Ordenamiento de Inserción Directa**

El método de inserción directa es el que generalmente utilizan los jugadores de cartas cuando ordenan éstas, de ahí que también se conozca con el nombre de método de la baraja.

La idea central de este algoritmo consiste en insertar un elemento del arreglo en la parte izquierda del mismo, que ya se encuentra ordenada. Este proceso se repite desde el segundo hasta el n-esimo elemento.

**Ejemplo:**

Se desean ordenarse las siguientes clave del arreglo A: 15, 67, 08, 16, 44, 27, 12, 35

**Primera pasada**

*A[2] < A[1] 67 < 15 No hay intercambio*

*A: 15, 67, 08, 16, 44, 27, 12, 35*

**Segunda pasada**

*A[3] < A[2] 08 < 67 Si hay intercambio*

*A[2] < A[1] 08 < 15 Si hay*

*A: 15, 08, 67, 16, 44, 27, 12, 35*

**Tercera pasada**

*A[4] < A[3] 08 < 15 Si hay intercambio*

*A[3] < A[2] 08 < 15 Si hay intercambio*

*A= 08, 15, 67, 16, 44, 27, 12, 35*

Hasta la séptima pasada el arreglo queda ordenado:   08, *12, 15, 16, 27, 35, 44, 67*  
**A continuación veamos un ejemplo en Pseudo-Código Para el Método de Shell Sort**  
 *void InsercionDirecta (int arreglo[], int tam)*  
*{*  
 *int i ,  j , valorBusqueda;*  
 *for(i = 1; i < tam; i++)*  
 *{*  
 *valorBusqueda = a[i];*  
 *for (j = i - 1; j >= 0 && a[j] > valorBusqueda; j--)*  
 *{*  
 *a[j + 1] = a[j];*  
 *}*  
 *a[j + 1] = valorBusqueda;*  
 *}*  
 *val = a[i];*  
*}*

**Ejemplo del Método de Ordenamiento Inserción Directa**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace PruebaVector

{

    class PruebaVector

    {

        private int[] vector;

        public void Cargar()

        {

            Console.WriteLine("Metodo de insercion directa");

            Console.Write("Cuantos longitud del vector:");

            string linea;

            linea = Console.ReadLine();

            int cant;

            cant = int.Parse(linea);

            vector = new int[cant];

            for (int f = 0; f < vector.Length; f++)

            {

                Console.Write("Ingrese elemento "+(f+1)+": ");

                linea = Console.ReadLine();

                vector[f] = int.Parse(linea);

            }

        }

        public void InsercionDirecta()

        {

            int auxili;

            int j;

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

            {

                auxili = vector[i];

                j = i - 1;

                while (j >= 0 && vector[j] > auxili)

                {

                    vector[j + 1] = vector[j];

                    j--;

                }

                vector[j + 1] = auxili;

            }

        }

        public void Imprimir()

        {

            Console.WriteLine("Vector ordenados en forma ascendente");

            for (int f = 0; f < vector.Length; f++)

            {

                Console.Write(vector[f]+"  ");

            }

            Console.ReadKey();

        }

        static void Main(string[] args)

        {

            PruebaVector pv = new PruebaVector();

            pv.Cargar();

            pv.InsercionDirecta();

            pv.Imprimir();

        }

    }

}

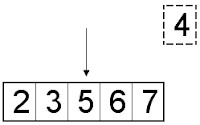
### Método de Ordenamiento de Inserción Binaria

La Inserción Binaria, es una mejora de la Inserción Directa.

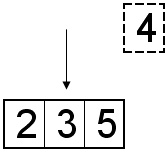
La Inserción Binaria es un método de inserción, que difiere de la Inserción Directa simplemente en la búsqueda de la posición de la Inserción.

Veamos un ejemplo de como funciona este método de ordenamiento de Inserción Binaria, analizando paso a paso.

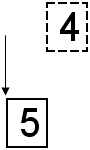
1. Hallamos el elemento central del área comprendida por la parte ordenada más la posición del elemento a insertar.

[](http://2.bp.blogspot.com/-MUDQRWsUSec/UeVTVcx0NPI/AAAAAAAABUU/Mx2kc5b_knk/s1600/insercion+directa+c#+1.jpg)

2. Comparamos el elemento central con el elemento que queremos insertar. Si dicho elemento central es menor o igual, nos quedamos con la parte derecha (Sin incluir el elemento central). En caso contrario, nos quedamos con la parte izquierda incluyendo al elemento central.

[](http://3.bp.blogspot.com/-giFHJen3Z6o/UeVT1p4phAI/AAAAAAAABUc/qzMfpXzLh1U/s1600/insercion+directa+c#+2.jpg)

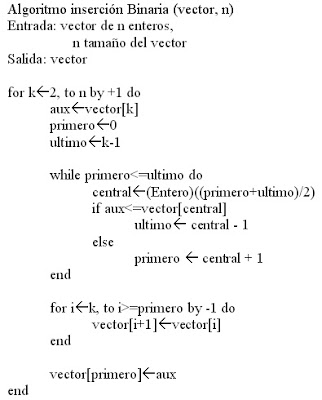
3. Repetimos el mismo proceso sobre el área con la que nos quedamos, hasta que dicha área sea nula.

[](http://3.bp.blogspot.com/-f8nlJ3p4NYc/UeVULgsHDYI/AAAAAAAABUk/gcWBzwFIarE/s1600/insercion+directa+c#+3.jpg)

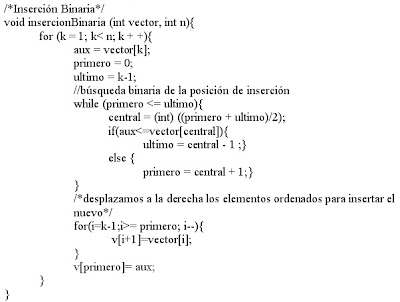
4. Cuando el área sea nula, tendremos la posición de inserción.

[http://1.bp.blogspot.com/--iAnJwvGYMw/UeVUTyB51oI/AAAAAAAABUs/ZEReKeu47q8/s1600/insercion+directa+c%23+4.jpg](http://1.bp.blogspot.com/--iAnJwvGYMw/UeVUTyB51oI/AAAAAAAABUs/ZEReKeu47q8/s1600/insercion+directa+c#+4.jpg)

**A continuación veamos un ejemplo en Pseudo-Código Para el Método de Inserción Binaria**

[](http://2.bp.blogspot.com/-Wev9Xxn3xiw/UeVUnCrwBhI/AAAAAAAABU0/BqJjkj8yGGY/s1600/pseudocodigo+de+insercion+binaria.jpg)

### Ejemplo del Código Implementado

[](http://4.bp.blogspot.com/-gXBaPwN5in0/UebuLfXvT5I/AAAAAAAABWM/tXkTLsfP8Sc/s1600/insercion+binaria+en+C#.jpg)

### Un ejemplo de método de inserción binaria

                    int auxiliar;

                    int q;

                    int izqui;

                    int dere;

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

                    {

                        auxiliar = numeros[i];

                        izqui = 0;

                        dere = i - 1;

                        while (izqui <= dere)

                        {

                            q = ((izqui + dere) / 2);

                            if (auxiliar < numeros[q])

                            {

                                dere = q - 1;

                            }

                            else

                            {

                                izqui = q + 1;

                            }

                            j = i - 1;

                            while (j >= izqui)

                            {

                                numeros[j + 1] = numeros[j];

                                j = j - 1;

                            }

                            numeros[izqui] = auxiliar;

                        }

                    }

                    for (int i = 0; i < longitud; i++)

                        Console.WriteLine(" " + numeros[i]);