```
using System;
using System.IO;
using System.Diagnostics;

namespace BubbleSort
{
    class Bubble_Sort
    {
        static void Main(string[] args)
        {
            int seed = (int)DateTime.Now.Ticks & 0x0000FFFF;

            // Antras etapas
            Test_File_Array_List(seed);

        }
        public static void BubbleSort(DataArray items)
        {
            double prevdata, currentdata;
            for (int i = items.Length - 1; i >= 0; i--)
            {
                currentdata = items[0];
                for (int j = 1; j <= i; j++)
                {
                    prevdata = currentdata;
                    currentdata = items[j];
                    if (prevdata > currentdata)
                    {
                        items.Swap(j, currentdata, prevdata);
                        currentdata = prevdata;
                    }
                }
            }
        }

        public static void BubbleSort(DataList items)
        {
            double prevdata, currentdata;
            for (int i = items.Length - 1; i >= 0; i--)
            {
                currentdata = items.Head();
                for (int j = 1; j <= i; j++)
                {
                    prevdata = currentdata;
                    currentdata = items.Next();
                    if (prevdata > currentdata)
                    {
                        items.Swap(currentdata, prevdata);
                        currentdata = prevdata;
                    }
                }
            }
        }

        public static void Test_File_Array_List(int seed)
        {
            int n = 12;
            string filename;
            filename = @"mydataarray.dat";
            MyFileArray myfilearray = new MyFileArray(filename, n, seed);
            using (myfilearray.fs = new FileStream(filename, FileMode.Open, FileAccess.ReadWrite))
            {
                Console.WriteLine("\n FILE ARRAY \n");
                myfilearray.Print(n);
                BubbleSort(myfilearray);
                myfilearray.Print(n);
            }
            filename = @"mydatalist.dat";
            MyFileList myfilelist = new MyFileList(filename, n, seed);
            using (myfilelist.fs = new FileStream(filename, FileMode.Open, FileAccess.ReadWrite))
            {
                Console.WriteLine("\n FILE LIST \n");
                myfilelist.Print(n);
                BubbleSort(myfilelist);
                myfilelist.Print(n);
            }
        }

    }
    abstract class DataArray
    {
        protected int length;
        public int Length { get { return length; } }
        public abstract double this[int index] { get; }
        public abstract void Swap(int j, double a, double b);
        public void Print(int n)
        {
            for (int i = 0; i < n; i++)
                Console.Write(" {0:F5} ", this[i]);
            Console.WriteLine();
        }
    }

    abstract class DataList
    {
        protected int length;
        public int Length { get { return length; } }
        public abstract double Head();
        public abstract double Next();
        public abstract void Swap(double a, double b);
        public void Print(int n)
        {
            Console.Write(" {0:F5} ", Head());
            for (int i = 1; i < n; i++)
                Console.Write(" {0:F5} ", Next());
            Console.WriteLine();
        }
    }
}
```

```
using System;
using System.IO;

namespace BubbleSort
{
   class MyFileArray : DataArray
  {
    public MyFileArray(string filename, int n, int seed)
    {
      double[] data = new double[n];
      length = n;
      Random rand = new Random(seed);
      for (int i = 0; i < length; i++)
      {
        data[i] = rand.NextDouble();
      }
      if (File.Exists(filename)) File.Delete(filename);
      try
      {
        using (BinaryWriter writer = new BinaryWriter(File.Open(filename, FileMode.Create)))
        {
          for (int j = 0; j < length; j++)
            writer.Write(data[j]);
        }
      }
      catch (IOException ex)
      {
        Console.WriteLine(ex.ToString());
      }
    }

    public FileStream fs { get; set; }

    public override double this[int index]
    {
      get
      {
        Byte[] data = new Byte[8];
        fs.Seek(8 * index, SeekOrigin.Begin);
        fs.Read(data, 0, 8);
        double result = BitConverter.ToDouble(data, 0);
        return result;
      }
    }
    public override void Swap(int j, double a, double b)
    {
      Byte[] data = new Byte[16];
      BitConverter.GetBytes(a).CopyTo(data, 0);
      BitConverter.GetBytes(b).CopyTo(data, 8);
      fs.Seek(8 * (j - 1), SeekOrigin.Begin);
      fs.Write(data, 0, 16);
```

```
        }
    }
}

```

```
using System;
using System.IO;

namespace BubbleSort
{
     class MyFileList : DataList
    {
        int prevNode;
        int currentNode;
        int nextNode;

        public MyFileList(string filename, int n, int seed)
        {
            length = n;
            Random rand = new Random(seed);
            if (File.Exists(filename)) File.Delete(filename);
            try
            {
                using (BinaryWriter writer = new BinaryWriter(File.Open(filename, FileMode.Create)))
                {
                    writer.Write(4);
                    for (int j = 0; j < length; j++)
                    {
                        writer.Write(rand.NextDouble());
                        writer.Write((j + 1) * 12 + 4);
                    }
                }
            }
            catch (IOException ex)
            {
                Console.WriteLine(ex.ToString());
            }
        }
        public FileStream fs { get; set; }
        public override double Head()
        {
            Byte[] data = new Byte[12];
            fs.Seek(0, SeekOrigin.Begin);
            fs.Read(data, 0, 4);
            currentNode = BitConverter.ToInt32(data, 0);
            prevNode = -1;
            fs.Seek(currentNode, SeekOrigin.Begin);
            fs.Read(data, 0, 12);
            double result = BitConverter.ToDouble(data, 0);
            nextNode = BitConverter.ToInt32(data, 8);
            return result;
        }
        public override double Next()
        {
            Byte[] data = new Byte[12];
            fs.Seek(nextNode, SeekOrigin.Begin);
            fs.Read(data, 0, 12);
            prevNode = currentNode;
            currentNode = nextNode;
            double result = BitConverter.ToDouble(data, 0);
            nextNode = BitConverter.ToInt32(data, 8);
            return result;
        }

        public override void Swap(double a, double b)
        {
            Byte[] data;
            fs.Seek(prevNode, SeekOrigin.Begin);
            data = BitConverter.GetBytes(a);
            fs.Write(data, 0, 8);
            fs.Seek(currentNode, SeekOrigin.Begin);
            data = BitConverter.GetBytes(b);
            fs.Write(data, 0, 8);
        }

    }
}

```