Note: Marge sort and Shell sort is missing….

Class: Array

```
import java.util.*;

public abstract class Array {

    String s;
    int arr[];
    Scanner input = new Scanner(System.in);

    public Array(){

        System.out.println("Input the size of array");
        int length = input.nextInt();
        arr = new int[length];

    }

    public void input() {

        System.out.println("input " + arr.length + " values for array");
```

```java
        for(int x = 0; x < arr.length; x++)
            arr[x] = input.nextInt();

    }


    public abstract void sort();


    public void display() {

        System.out.println("Output");
        for(int x = 0; x < arr.length; x++)
            System.out.println(arr[x]);

    }


}
```

Class: BubbleSort

```java
public class BubbleSort extends Array {

    public void sort() {

        int x, y;

        for(x = 0; x < arr.length-1; x++)

            for(y = 0; y < arr.length-1; y++)

                if(arr[y] > arr[y+1]) {

                    int temp = arr[y];
                    arr[y] = arr[y+1];
                    arr[y+1] = temp;

                }

    }

}
```

Class: SelectionSort

```java
public class SelectionSort extends Array{

    public void sort() {

        for(int y=0; y < arr.length - 1; y++) {

            int hold  = y;

            for(int x = y; x < arr.length; x++)

                if(arr[hold] > arr[x]) {
                    int temp = hold;
                    hold = x;
                    x = temp;
                }


            int temp = arr[hold];
            arr[hold] = arr[y];
            arr[y]= temp;

        }

    }

}
```

Class: InsertionSort

```java
public class InsertionSort extends Array{

    public void sort() {

        for(int y=1; y < arr.length; y++)

            for(int x=y; x > 0; x--)

                if(arr[x] < arr[x-1]) {

                    int temp = arr[x];
                    arr[x] = arr[x-1];
                    arr[x-1] = temp;
                }


    }

}
```

Class: QuickSort

```java
public class QuickSort extends Array {

    public void sort() {

        Qsort(arr, 0, arr.length - 1);

    }

    public void Qsort(int arr[], int low, int high) {

        int pi;

        if(low < high) {

            pi = partation(arr, low, high);
            Qsort(arr, low, pi-1);
            Qsort(arr, pi+1, high);

        }

    }


    private int partation(int arr[], int low, int high) {

        int i = low - 1;
        int pivot = arr[high];

        for(int j = low; j <= high - 1; j++)
```

```
            if(arr[j] < pivot) {
                i++;
                int temp = arr[i];
                arr[i] = arr[j];
                arr[j] = temp;
            }

        int temp = arr[i+1];
        arr[i+1] = arr[high];
        arr[high] = temp;


        return i + 1;
    }


}
```

Class: MargeSort

```
public class MargeSort extends Array {
    public void sort() {
        System.out.println("marge sort is not availabe");
    }
}
```

Class: ShellSort

```java
public class ShellSort extends Array{
    public void sort() {
        System.out.println("shell sort is not availabe");
    }
}
```

Class: Execute

```
import java.util.*;
public class Execute {

    public static void main(String args[]) {

        Array obj = null;

        System.out.println("input"
                + "\n 1 for bubble sort,"
                + "\n 2 for selection sort,"
                + "\n 3 for insertion sort,"
                + "\n 4 for quick sort,"
                + "\n 5 for marge sort,"
                + "\n 6 for shell sort");


        Scanner input = new Scanner(System.in);

        switch(input.nextInt()) {

        case 1:
            obj = new BubbleSort();
            break;

        case 2:
            obj = new SelectionSort();
            break;

        case 3:
            obj = new InsertionSort();
            break;
        case 4:
            obj = new QuickSort();
            break;

        case 5:
            obj = new MargeSort();
            break;

        case 6:
            obj = new ShellSort();
            break;


        default:
            System.out.println("unexcepted input, terminating the program");
            System.exit(0);
            break;

        }

        obj.input();
        obj.sort();
        obj.display();

        input.close();
        System.exit(0);

    }

}
```