Task#3 Remaining

Sorting Algorithm

Quick sort:

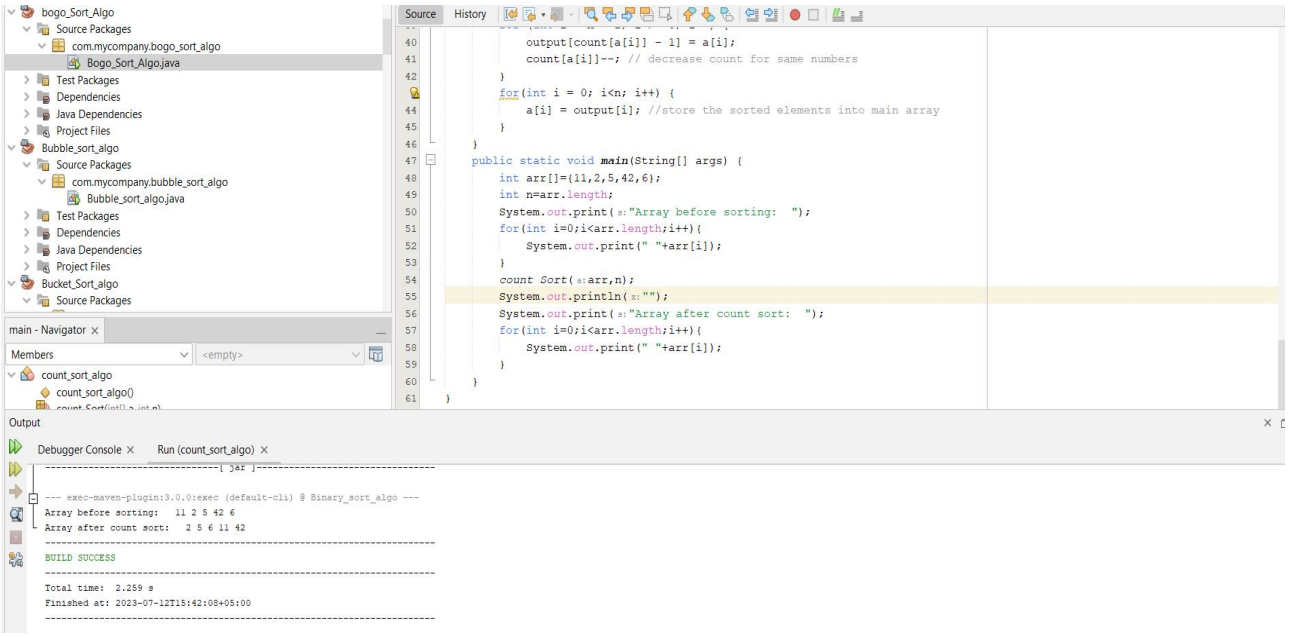
```
public class Quick_sort_algo {
    public static void swap(int[] arr,int j,int k) {
        int temp=arr[j];
        arr[j]=arr[k];
        arr[k]=temp;
    }
     public static int Array_partition(int[] arr,int l,int h) {
        int pivot = arr[h];
        int i = (l - 1);

        for (int j = l; j <= h - 1; j++) {
            if (arr[j] < pivot) {
                i++;
                swap(arr, j:i, k:j);
            }
        }
        swap(arr, i + 1, k:h);
        return (i + 1);
    }
    public static void quick_Sort(int[] arr, int l,int h) {
        if(l<h){
            int partition=Array_partition(arr,l,h);
            quick_Sort(arr, l, partition - 1);
            quick_Sort(arr, partition + 1, h);
        }

    }

    public static void main(String[] args) {
        int arr[]={11,2,5,42,6};
        int n=arr.length-1;
        System.out.print( s:"Array before sorting:  ");
        for(int i=0;i<arr.length;i++){
            System.out.print(" "+arr[i]);
```

```
    }

    public static void main(String[] args) {
        int arr[]={11,2,5,42,6};
        int n=arr.length-1;
        System.out.print( s:"Array before sorting:  ");
        for(int i=0;i<arr.length;i++){
            System.out.print(" "+arr[i]);
        }
        quick_Sort(arr, l:0, h:n);
        System.out.println( x:"");
        System.out.print( s:"Array after quick sort:  ");
        for(int i=0;i<arr.length;i++){
            System.out.print(" "+arr[i]);
        }
    }
}
```

```
--------------------------------[ jar ]---------------------------------

--- exec-maven-plugin:3.0.0:exec (default-cli) @ Quick_sort_algo ---
Array before sorting:   11 2 5 42 6
Array after quick sort:   2 5 6 11 42
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  1.610 s
Finished at: 2023-07-12T15:39:45+05:00
------------------------------------------------------------------------
```

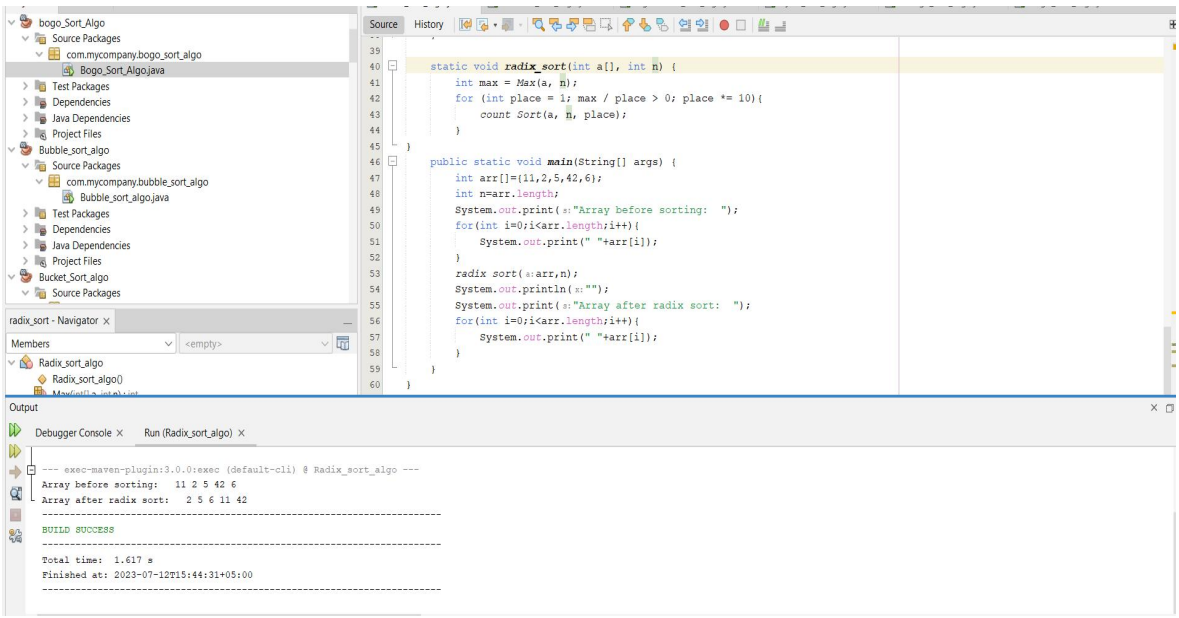
Count sort:

```
*/
public class count_sort_algo {

    static int getMax(int a[], int n) {
        int max = a[0];
        for(int i = 1; i<n; i++) {
            if(a[i] > max){
                max = a[i];
            }
        }
        return max;
    }
     static void count_Sort(int a[], int n){
        int[] output=new int[n+1];
        int max = getMax(a, n);
        int[] count=new int[max+1];
        for (int i = 0; i <= max; ++i)
        {
            count[i] = 0;
        }
        for (int i = 0; i < n; i++)
        {
            count[a[i]]++;
        }

        for(int i = 1; i<=max; i++){
            count[i] += count[i-1];
        }
        for (int i = n - 1; i >= 0; i--) {
            output[count[a[i]] - 1] = a[i];
            count[a[i]]--; // decrease count for same numbers
        }
        for(int i = 0; i<n; i++) {
            a[i] = output[i]; //store the sorted elements into main array
        }
```



Radix sort:

```java
public class Radix_sort_algo {
    static int Max(int a[], int n) {
        int max = a[0];
        for(int i = 1; i<n; i++) {
            if(a[i] > max)
            max = a[i];
        }
    return max;
    }

    static void count_Sort(int a[], int n, int place)
    {
        int[] output = new int[n+1];
        int[] count = new int[10];
        for (int i = 0; i < n; i++){
            count[(a[i] / place) % 10]++;
        }
        for (int i = 1; i < 10; i++){
            count[i] += count[i - 1];
        }
        for (int i = n - 1; i >= 0; i--) {
            output[count[(a[i] / place) % 10] - 1] = a[i];
            count[(a[i] / place) % 10]--;
        }
        for (int i = 0; i < n; i++){
            a[i] = output[i];  }
    }

    static void radix_sort(int a[], int n) {
        int max = Max(a, n);
        for (int place = 1; max / place > 0; place *= 10){
            count_Sort(a, n, place);
        }
}
```



Cocktail sort:

```java
public class Cocktail_sort_Algo {

    public static void main(String[] args) {
        int arr[]={11,2,5,42,6};
        System.out.print( s:"Array before sorting:  ");
        for(int i=0;i<arr.length;i++){
            System.out.print(" "+arr[i]);
        }
        boolean swap = true;
        int S = 0;
        int E = arr.length;

        while (swap== true)
        {
            swap = false;
            for (int i = S; i < E - 1; ++i)
            {
                if (arr[i] > arr[i + 1]) {
                    int temp = arr[i];
                    arr[i] = arr[i + 1];
                    arr[i + 1] = temp;
                    swap = true;
                }
            }
            if (swap == false){
                break;
            }
            swap = false;
            E= E - 1;
            for (int i = E - 1; i >= S; i--)
            {
                if (arr[i] > arr[i + 1])
                {
                    int temp = arr[i];
                    arr[i] = arr[i + 1];
                    arr[i + 1] = temp;
                    swap = true;
                }
            }
            if (swap == false){
                break;
            }
            swap = false;
            E= E - 1;
            for (int i = E - 1; i >= S; i--)
            {
                if (arr[i] > arr[i + 1])
                {
                    int temp = arr[i];
                    arr[i] = arr[i + 1];
                    arr[i + 1] = temp;
                    swap = true;
                }
            }
            S = S + 1;
        }
        System.out.println( x:"");
        System.out.print( s:"Array after cocktail sort:  ");
        for(int i=0;i<arr.length;i++){
            System.out.print(" "+arr[i]);
        }
    }
}
```

```
--- exec-maven-plugin:3.0.0:exec (default-cli) @ Cocktail_sort_Algo ---
Array before sorting:   11 2 5 42 6
Array after cocktail sort:   2 5 6 11 42
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  1.719 s
Finished at: 2023-07-12T15:48:10+05:00
------------------------------------------------------------------------
```

Pigeonhole sort:

```java
public class Pigeonhole_sort_algo {

    public static void main(String[] args) {
        int arr[]={11,2,5,42,6};
        System.out.print( s:"Array before sorting:  ");
        for(int i=0;i<arr.length;i++){
            System.out.print(" "+arr[i]);
        }
        int n=arr.length;
        int min = arr[0];
        int max = arr[0];
        int range, j, k, index;

        for(int a=0; a<n; a++)
        {
            if(arr[a] > max)
                max = arr[a];
            if(arr[a] < min)
                min = arr[a];
        }

        range = max - min + 1;
        int[] pigeonhole = new int[range];
        Arrays.fill( a:pigeonhole,  val:0);

        for(j = 0; j<n; j++){
            pigeonhole[arr[j] - min]++;
        }

        index = 0;

        for(k = 0; k<range; k++){
            while(pigeonhole[k]-->0){
                max = arr[a];
            if(arr[a] < min)
                min = arr[a];
        }

        range = max - min + 1;
        int[] pigeonhole = new int[range];
        Arrays.fill( a:pigeonhole,  val:0);

        for(j = 0; j<n; j++){
            pigeonhole[arr[j] - min]++;
        }

        index = 0;

        for(k = 0; k<range; k++){
            while(pigeonhole[k]-->0){
                arr[index++]=k+min;}
        }
        System.out.println( x:"");
        System.out.print( s:"Array after pigeonhole sort Algorithm:  ");
        for(int a=0;a<arr.length;a++){
            System.out.print(" "+arr[a]);
        }
    }
}
```

```
--- exec-maven-plugin:3.0.0:exec (default-cli) @ pigeonhole_sort_algo ---
Array before sorting:   11 2 5 42 6
Array after pigeonhole sort Algorithm:   2 5 6 11 42
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  1.920 s
Finished at: 2023-07-12T15:50:46+05:00
------------------------------------------------------------------------
```

Cycle sort:

```java
public class Cycle sort algo {

    public static void main(String[] args) {
        int arr[]={11,2,5,42,6};
        System.out.print( s:"Array before sorting:  ");
        for(int i=0;i<arr.length;i++){
            System.out.print(" "+arr[i]);
        }
        int n=arr.length;
        int count = 0;
        for (int cycle_s= 0; cycle_s <= n - 2; cycle_s++) {
            int item= arr[cycle_s];
            int p = cycle_s;
            for (int i = cycle_s + 1; i < n; i++){
                if (arr[i] < item)
                {
                    p++;
                }
            }
            if (p == cycle_s){
                continue;}
            while (item == arr[p]){
                p=p+1;

            }
            if (p != cycle_s) {
                int temp = item;
                item = arr[p];
                arr[p] = temp;
                count++;
            }
          , while (p != cycle_s) {
            if (p != cycle_s) {
                int temp = item;
                item = arr[p];
                arr[p] = temp;
                count++;
            }
            while (p != cycle_s) {
                p = cycle_s;
                for (int i = cycle_s + 1; i < n; i++)
                    if (arr[i] < item)
                        p=p + 1;
                while (item == arr[p])
                {
                    p += 1;
                }
                if (item != arr[p]) {
                    int temp = item;
                    item = arr[p];
                    arr[p] = temp;
                    count++;
                }
            }
        }
        System.out.println( x:"");
        System.out.print( s:"Array after cycle sort Algorithm:  ");
        for(int a=0;a<arr.length;a++){
            System.out.print(" "+arr[a]);
        }
    }
}
```

```
] --- exec-maven-plugin:3.0.0:exec (default-cli) @ cycle_sort_algo ---
  Array before sorting:   11 2 5 42 6
- Array after cycle sort Algorithm:   2 5 6 11 42
  ------------------------------------------------------------------------
  BUILD SUCCESS
  ------------------------------------------------------------------------
  Total time:  1.699 s
  Finished at: 2023-07-12T15:53:54+05:00
  ------------------------------------------------------------------------
```

Stooge sort:

```java
public class Stooge_sort_Algo {
    static void stooge_sort(int arr[], int l, int h)
    {
        if (l >= h)
            return;
        if (arr[l] > arr[h]) {
            int t = arr[l];
            arr[l] = arr[h];
            arr[h] = t;
        }
        if (h - l + 1 > 2) {
            int t = (h - l + 1) / 3;
            stooge_sort(arr, l, h - t);
            stooge_sort(arr, l + t, h);
            stooge_sort(arr, l, h - t);
        }
    }
    public static void main(String[] args) {
        int arr[]={11,2,5,42,6};
        System.out.print( s:"Array before sorting:  ");
        for(int i=0;i<arr.length;i++){
            System.out.print(" "+arr[i]);
        }
        int n=arr.length;
        stooge_sort(arr, 1:0,n-1);
        System.out.println( x:"");
        System.out.print( s:"Array after stooge sort algorithm:  ");
        for(int i=0;i<arr.length;i++){
            System.out.print(" "+arr[i]);
        }
    }
}
```

```
--------------------------------[ jar ]---------------------------------

--- exec-maven-plugin:3.0.0:exec (default-cli) @ stooge_sort_Algo ---
Array before sorting:   11 2 5 42 6
Array after stooge sort algorithm:   2 5 6 11 42
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  1.611 s
Finished at: 2023-07-12T15:56:09+05:00
------------------------------------------------------------------------
```

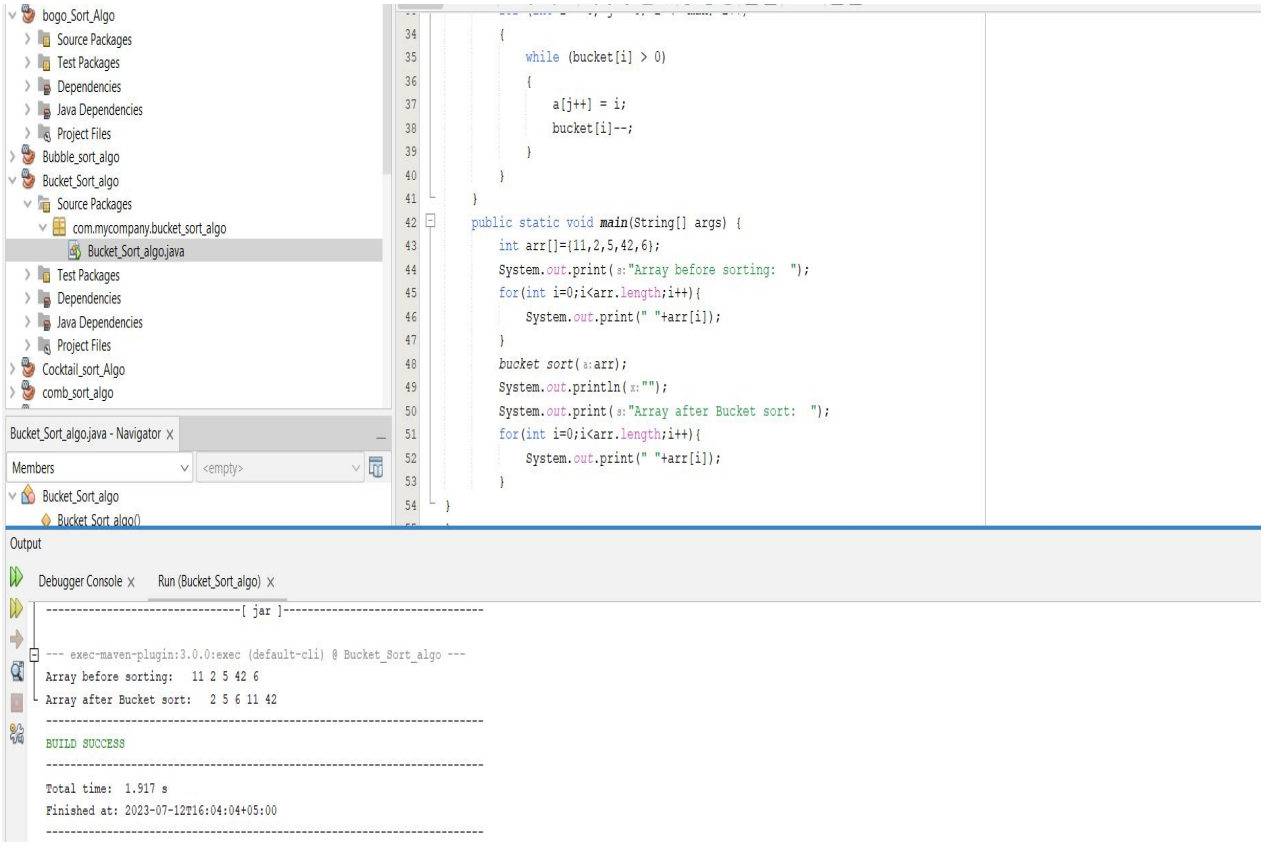
Bogo sort:

```java
public class Bogo_Sort_Algo {
    static void bogo_sort(int[] a)
    {
        while (sort(a) == false)
            shuffle(a);
    }
    static void shuffle(int[] a)
    {
        for (int i = 1; i < a.length; i++)
            swap(a, i, (int)(Math.random() * i));
    }
    static void swap(int[] a, int i, int j)
    {
        int temp = a[i];
        a[i] = a[j];
        a[j] = temp;
    }
    static boolean sort(int[] a)
    {
        for (int i = 1; i < a.length; i++)
            if (a[i] < a[i - 1])
                return false;
        return true;
    }
    public static void main(String[] args) {
        int arr[]={11,2,5,42,6};
        System.out.print( s:"Array before sorting:  ");
        for(int i=0;i<arr.length;i++){
            System.out.print(" "+arr[i]);
        }
        bogo_sort( a:arr);
        System.out.println( x:"");
        System.out.print( s:"Array after Bogo sort algorithm:  ");
        for(int i=0;i<arr.length;i++){
            System.out.print(" "+arr[i]);
```

```
-------------------< com.mycompany:bogo_Sort_Algo >-------------------
Building bogo_Sort_Algo 1.0-SNAPSHOT
--------------------------------[ jar ]---------------------------------

--- exec-maven-plugin:3.0.0:exec (default-cli) @ bogo_Sort_Algo ---
Array before sorting:   11 2 5 42 6
Array after Bogo sort algorithm:   2 5 6 11 42
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  1.606 s
Finished at: 2023-07-12T15:58:38+05:00
------------------------------------------------------------------------
```

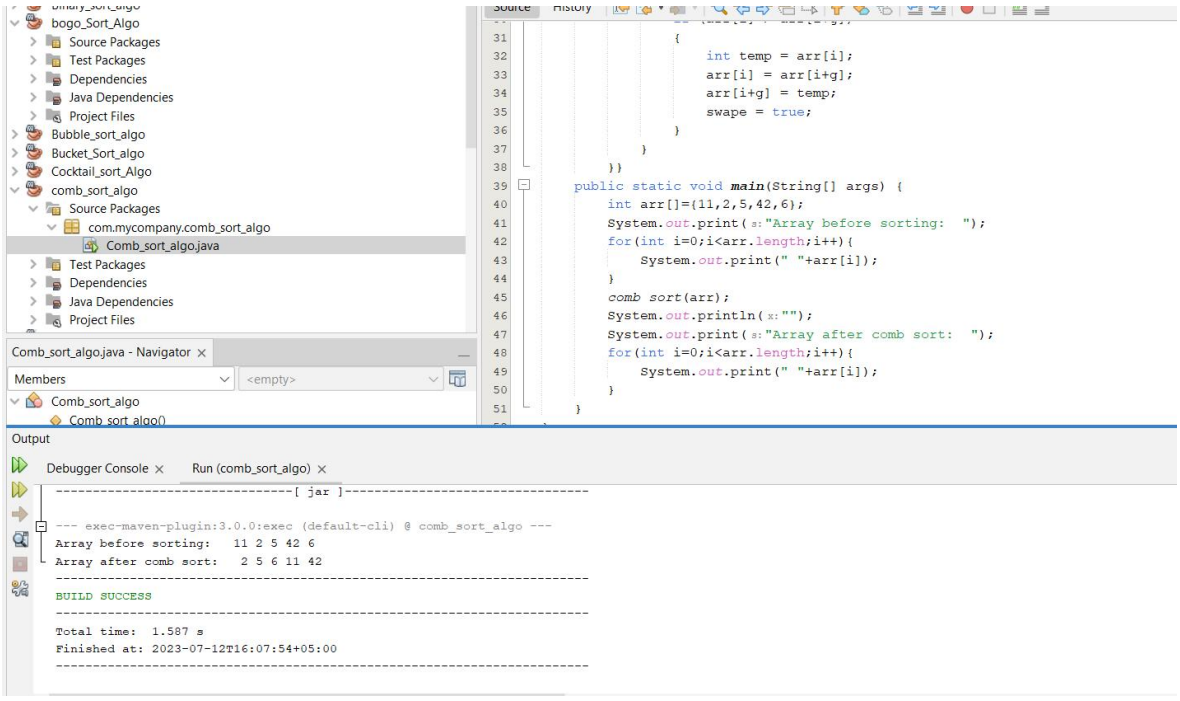
Bucket sort:

```
public class Bucket_Sort_algo {
    static int Max(int a[], int n) {
        int max = a[0];
        for(int i = 1; i<n; i++) {
            if(a[i] > max)
            max = a[i];
        }
    return max;
    }
    static void bucket_sort(int a[]){
        int n = a.length;
        int max = Max(a,n);
        int bucket[] = new int[max+1];
        for (int i = 0; i <= max; i++)
        {
            bucket[i] = 0;
        }
        for (int i = 0; i < n; i++)
        {
            bucket[a[i]]++;
        }
        for (int i = 0, j = 0; i <= max; i++)
        {
            while (bucket[i] > 0)
            {
                a[j++] = i;
                bucket[i]--;
            }
        }
    }
    public static void main(String[] args) {
        int arr[]={11,2,5,42,6};
        System.out.print( s: "Array before sorting:  ");
        for(int i=0;i<arr.length;i++){
```



Comb sort:

```java
public class Comb_sort_algo {
    static int Next_Gap(int g)
    {
        g= (g*10)/13;
        if (g < 1)
            return 1;
        return g;
    }
    static void comb_sort(int arr[])
    {
        int n = arr.length;
        int g = n;
        boolean swape = true;
        while (g!= 1 || swape == true){
            g =Next_Gap(g);
            swape = false;
            for (int i=0; i<n-g; i++)
            {
                if (arr[i] > arr[i+g])
                {
                    int temp = arr[i];
                    arr[i] = arr[i+g];
                    arr[i+g] = temp;
                    swape = true;
                }
            }
        }}
    public static void main(String[] args) {
        int arr[]={11,2,5,42,6};
        System.out.print( s:"Array before sorting:  ");
        for(int i=0;i<arr.length;i++){
            System.out.print(" "+arr[i]);
        }
        comb sort(arr);
        System.out.println( x:"");
```



Shell sort:

```java
public class Shell_Sort_algo {
    static int shell_sort(int arr[])
    {
        int n = arr.length;
        for (int g = n/2; g > 0; g /= 2)
        {
            for (int i = g; i < n; i += 1)
            {
                int temp = arr[i];
                int j;
                for (j = i; j >= g && arr[j - g] > temp; j -= g){
                    arr[j] = arr[j - g];
                }
                arr[j] = temp;
            }
        }
        return 0;
    }
    public static void main(String[] args) {
        int arr[]={11,2,5,42,6};
        System.out.print( s:"Array before sorting:  ");
        for(int i=0;i<arr.length;i++){
            System.out.print(" "+arr[i]);
        }
        shell_sort(arr);
        System.out.println( x:"");
        System.out.print( s:"Array after shell sort:  ");
        for(int i=0;i<arr.length;i++){
            System.out.print(" "+arr[i]);
    }
    }
}
```

```
--- exec-maven-plugin:3.0.0:exec (default-cli) @ Shell_Sort_algo ---
Array before sorting:   11 2 5 42 6
Array after shell sort:   2 5 6 11 42
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  1.646 s
Finished at: 2023-07-12T16:09:51+05:00
------------------------------------------------------------------------
```