Sorting

```
class InsertionSort {

	void insertionSort(int arr[]){
		int len = arr.length;
		int key;
		for(int i=1;i<len;i++){
			key = arr[i];
			int j = i-1;
			while (j>=0 && arr[j] > key)
		{
			arr[j+1] = arr[j];
			j = j-1;
		}
	arr[j+1] = key;
		}
	}

	void printArray(int arr[]){
		for(int i=0;i<arr.length;i++){
			System.out.print(arr[i] + " ");
		}
		System.out.println();
	}

	public static void main(String args[]){
		int arr[] = {5,4,2,7,4,1,9};
		InsertionSort s = new InsertionSort();
		System.out.println("Array Before Sorting....");
		s.printArray(arr);
		System.out.println("Array After Insertion Sorting....");
		s.insertionSort(arr);
		s.printArray(arr);
	}
}
```

```
class MergeSort{

	public void mergeSort(int arr[],int low, int high){
		if(low < high){
			int mid = (low + high)/2;
			mergeSort(arr,low,mid);
			mergeSort(arr,mid+1,high);
			merge(arr,low,mid,high);
		}
	}

	public void merge(int arr[],int low,int mid,int high){
		int len1 = mid - low + 1;
		int len2 = high - mid;
		int left[] = new int[len1];
		int right[] = new int[len2];
		for(int i=0;i<len1;i++){
			left[i] = arr[i + low];
		}
		for(int j=0;j<len2;j++){
			right[j] = arr[mid + j + 1];
		}
		int i = 0;
		int j = 0;
		int k = low;

		while(i < len1 && j < len2){
			if(left[i] <= right[j]){
				arr[k] = left[i];
				i++;
			}
			else {
				arr[k] = right[j];
				j++;
			}
			k++;
		}

		while(i < len1){
			arr[k] = left[i];
			k++;
			i++;
		}
```

```
            while(j < len2){
                arr[k] = right[j];
                k++;
                j++;
            }

        }

        public void printArray(int arr[]){
            for(int i=0;i<arr.length;i++){
                System.out.print(arr[i] + " ");
            }
            System.out.println();
        }

        public static void main(String args[]){
            int arr[] = {5,4,2,7,8,10,34,6,1,9};
            MergeSort m = new MergeSort();
            System.out.println("Array Before Sorting....");
            m.printArray(arr);
            System.out.println("Array After Merge Sorting....");
            m.mergeSort(arr,0,(arr.length)-1);
            m.printArray(arr);
        }
}
```

```
class QuickSort{

	void quickSort(int arr[], int low, int high){

		if(low == high)
			return;
		if(low < high){
			int pivot = partition(arr,low,high);
			quickSort(arr,low,pivot-1);
			quickSort(arr,pivot+1,high);
		}
	}

	int partition(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;
	}

	public void printArray(int arr[]){
		for(int i=0;i<arr.length;i++){
			System.out.print(arr[i] + " ");
		}
		System.out.println();
	}

	public static void main(String args[]){
		int arr[] = {5,4,2,7,8,10,34,6,1,9};
		QuickSort qs = new QuickSort();
		System.out.println("Array Before Sorting....");
```

```
                qs.printArray(arr);
                System.out.println("Array After Quick Sorting....");
                qs.quickSort(arr,0,(arr.length)-1);
                qs.printArray(arr);
        }
}



class HeapSort{

        public void heapSort(int arr[]){
                int len = arr.length;
                for(int i=len/2-1;i>=0;i--)
                        heapify(arr,len,i);

                for(int i=len-1;i>=0;i--){
                        int temp = arr[0];
                        arr[0] = arr[i];
                        arr[i] = temp;

                        heapify(arr,i,0);
                }
        }

        public void heapify(int arr[], int n, int i){
                int largest = i;
                int left = 2 * i;
                int right = 2 * i + 1;

                if(left < n && arr[left] > arr[largest])
                        largest = left;
                if(right < n && arr[right] > arr[largest])
                        largest = right;

                if(largest != i){
                        int temp = arr[largest];
                        arr[largest] = arr[i];
                        arr[i] = temp;

                        heapify(arr,n,largest);
                }

        }
```

```
	public void printArray(int arr[]){
		for(int i=0;i<arr.length;i++){
			System.out.print(arr[i] + " ");
		}
		System.out.println();
	}

	public static void main(String args[]){
		int arr[] = {5,4,2,7,8,10,34,6,1,9};
		HeapSort hs = new HeapSort();
		System.out.println("Array Before Sorting....");
		hs.printArray(arr);
		System.out.println("Array After Heap Sorting....");
		hs.heapSort(arr);
		hs.printArray(arr);
	}
}


class CountSort{

	public void countSort(int arr[], int k){
		int C[] = new int[k];
		int len = arr.length;
		int B[] = new int[len];

		for(int i=0; i<k; i++){
			C[i] = 0;
		}

		for(int i=0; i<len; i++){
			C[arr[i]] = C[arr[i]] + 1;
		}
		//printArray(C);

		for(int i=1; i<k; i++){
			C[i] = C[i] + C[i-1];
		}
		//printArray(C);

		for(int i=(len-1); i>=0; i--){
			B[C[arr[i]] - 1] = arr[i];
			C[arr[i]] = C[arr[i]] - 1;
```

```
            }

            printArray(B);

        }

        void printArray(int arr[]){
            for(int i=0;i<arr.length;i++){
                System.out.print(arr[i] + " ");
            }
            System.out.println();
        }

        public static void main(String args[]){
            CountSort cs = new CountSort();
            int arr[] = {3,6,1,8,3,9,0,3,2,6,7,8,8,0,1,5,4,4,2};
            System.out.println("Before Sorting.....");
            cs.printArray(arr);
            System.out.println("After Sorting......");
            cs.countSort(arr,10);
        }

}
```