BUBBLE SORT

```
bubbleSort(int arr[], int size)
{
  for(int i=0; i< size-1; i++)
   {
     for(int j=0; j< size-i-1; j++)
      {
        if (arr[j] > arr[j+1])
          {
            int temp = arr[j];
            arr[j] = arr[j+1];
            arr[j+1] = temp;
          }
      }
   }
}
```

SELECTION SORT

```
selectionSort(int arr[], int size)
{
    for(int i=0; i<size-1; i++)
    {
        int minPos = i;
        for(int j=i+1; j<size; j++)
        {
            if(arr[minPos] > arr[j])
                minPos = j;
        }
        if(i != minPos)
        {
            int temp = arr[minPos]
            arr[minPos] = arr[i]
            arr[i] = temp;
        }
    }
}
```

INSERTION SORT

```
insertionSort(int arr[], int size)
{
   for(int i=1; i<size; i++)
    {
      int currVal = arr[i];
      int prevIdx = i-1;
      while (prevIdx >= 0 &&
                    arr[prevIdx] > curr)
        {
          arr[prevIdx+1] = arr[prevIdx];
          prevIdx--;
        }
      arr[prevIdx+1] = curr;
    }
}
```

COUNTING SORT

```
countingSort(int arr[], int size)
{
    int max = -∞;
    for(int i = 0; i < arr.length; i++)
        max = Math.max(max, arr[i]);

    int freqSize = max + 1; // Including 0
    int freqArr[] = new freqArr[freqSize];

    for(int i=0; i < size; i++)
        freqArr[arr[i]]++;

    int idx = 0;
    for(int i=0; i < freqSize; i++)
        for(int j = 0; j < freqArr[i]; j++)
            arr[idx++] = i;

        freqArr[i] = 0; // Not Neccessary
}
```