**Insertion Sort**

public static void sort(int[] x){

for (int i = 1; i < x.length; i++){

int next = x[i];

int j = i;

while (j > 0 && next < x[j - 1]){

x[j] = x[j - 1];

j--;

}

x[j] = next;

}

}

**Merge Sort**

public class MergeSort{

public static void sort(int[] a)

{ sort(a, 0, a.length - 1); }

private static void sort(int[] a, int low, int high){

if (low < high)

{

int mid = (low + high) / 2;

sort(a, low, mid);

sort(a, mid + 1, high);

merge(a, low, mid, high);

}

}

/\*\* Precondition: a[low]...a[mid] is sorted && \* a[mid+1]...a[high] is sorted

\* Postcondition: a[low]...a[high] is sorted \*/

private static void merge(int[] a, int low, int mid, int high)

{

int[] b = new int[mid + 1 - low];

System.arraycopy(a, low, b, 0, b.length);

int aLowerIndex = low, bIndex = 0, aHigherIndex = mid + 1;

while (aLowerIndex < aHigherIndex && aHigherIndex <= high){

if (a[aHigherIndex] < b[bIndex])

a[aLowerIndex++] = a[aHigherIndex++];

else

a[aLowerIndex++] = b[bIndex++];

}

while (aLowerIndex < aHigherIndex)

a[aLowerIndex++] = b[bIndex++];

}

}