Merge Sort

**package** com.sorting;

**import** java.util.Scanner;

**public** **class** MyMergeSor {

**void** merge(**int** arr[], **int** beg, **int** mid, **int** end)

{

**int** l = mid - beg + 1;

**int** r = end - mid;

intLeftArray[] = **new** **int** [l];

intRightArray[] = **new** **int** [r];

**for** (**int** i=0; i<l; ++i)

LeftArray[i] = arr[beg + i];

**for** (**int** j=0; j<r; ++j)

RightArray[j] = arr[mid + 1+ j];

**int** i = 0, j = 0;

**int** k = beg;

**while** (i<l&&j<r)

{

**if** (LeftArray[i] <= RightArray[j])

{

arr[k] = LeftArray[i];

i++;

}

**else**

{

arr[k] = RightArray[j];

j++;

}

k++;

}

**while** (i<l)

{

arr[k] = LeftArray[i];

i++;

k++;

}

**while** (j<r)

{

arr[k] = RightArray[j];

j++;

k++;

}

}

**void** sort(**int** arr[], **int** beg, **int** end)

{

**if** (beg<end)

{

**int** mid = (beg+end)/2;

sort(arr, beg, mid);

sort(arr , mid+1, end);

merge(arr, beg, mid, end);

}

}

**public** **static** **void** main(String args[])

{

intarr[] = {90,23,101,45,65,23,67,89,34,23};

MyMergeSort ob = **new** MyMergeSort();

ob.sort(arr, 0, arr.length-1);

System.out.println("\nSorted array");

**for**(**int** i =0; i<arr.length;i++)

{

System.out.println(arr[i]+"");

}

}

}

Quick Sort

**package** com.sorting;

**public** **class** QuickSort {

**public** **static** **void** main(String[] args) {

**int** i;

**int**[] arr={90,23,101,45,65,23,67,89,34,23};

*quickSort*(arr, 0, 9);

System.***out***.println("\n The sorted array is: \n");

**for**(i=0;i<10;i++)

System.***out***.println(arr[i]);

}

**public** **static** **int** partition(**int** a[], **int** beg, **int** end)

{

**int** left, right, temp, loc, flag;

loc = left = beg;

right = end;

flag = 0;

**while**(flag != 1)

{

**while**((a[loc] <= a[right]) && (loc!=right))

right--;

**if**(loc==right)

flag =1;

elseif(a[loc]>a[right])

{

temp = a[loc];

a[loc] = a[right];

a[right] = temp;

loc = right;

}

**if**(flag!=1)

{

**while**((a[loc] >= a[left]) && (loc!=left))

left++;

**if**(loc==left)

flag =1;

elseif(a[loc] <a[left])

{

temp = a[loc];

a[loc] = a[left];

a[left] = temp;

loc = left;

}

}

}

returnloc;

}

**static** **void** quickSort(**int** a[], **int** beg, **int** end)

{

**int** loc;

**if**(beg<end)

{

loc = partition(a, beg, end);

quickSort(a, beg, loc-1);

quickSort(a, loc+1, end);

}

}