```
1: //-->HARSH MISHRA<--
2:
3:
4: void selectionSort(int A[], int n) {
5:     for(int i = 0;i < n-1; i++) {
6:         int iMin = i;
7:         for(int j = i+1; j < n;j++)
8:             if(A[j] < A[iMin])  iMin = j;
9:         swap(A[i],A[iMin]);
10:    }
11: }
12:
13:
14: void bubbleSort(int A[], int n) {
15:     for(int i = 0; i < n-1; i++) {
16:         for(int j = 0; j <n-1-i;j++)
17:             if(A[j] > A[j+1])   swap(A[j],A[j+1]);
18:     }
19: }
20:
21:
22: void insertionSort(int A[], int n) {
23:     for(int i = 1;i < n;i++) {
24:         int val = A[i];
25:         int hole = i;
26:         while(hole > 0 && A[hole-1] > val) {
27:             A[hole] = A[hole-1];
28:             hole = hole-1;
29:         }
30:         A[hole] = val;
31:     }
32: }
33:
34: int partition(int A[], int start, int end) {
35:     int pivot = A[end];
36:     int partitionIndex = start;
37:     for(int i = start;i < end; i++) {
38:         if(A[i] <= pivot) {
39:             swap(A[i],A[partitionIndex]);
40:             partitionIndex++;
41:         }
42:     }
43:     swap(A[partitionIndex],A[end]);
44:     return partitionIndex;
45: }
46:
```

```
47:
48: void quickSort(int A[], int start, int end) {
49:     if(start < end) {
50:         int partitionIndex = partition(A,start,end);
51:         quickSort(A,start,partitionIndex-1);
52:         quickSort(A,partitionIndex+1,end);
53:     }
54: }
55:
56: void merge(int A[], int start, int mid, int end) {
57:     int p = start, q = mid+1;
58:     int Arr[end-start+1], k = 0;
59:     for(int i = start; i <= end;i++) {
60:         if(p > mid) Arr[k++] = A[q++];
61:         else if(q > end) Arr[k++] = A[p++];
62:         else if(A[p] < A[q]) Arr[k++] = A[p++];
63:         else Arr[k++] = A[q++];
64:     }
65:     for(int p = 0;p < k;p++) A[start++] = Arr[p];
66: }
67:
68: void mergeSort(int A[], int start, int end) {
69:
70:     if(start < end) {
71:         int mid = (start+end)/2;
72:         mergeSort(A,start,mid);
73:         mergeSort(A,mid+1,end);
74:         merge(A,start,mid,end);
75:     }
76: }
77:
```