1.Bubble sort

**public** **class** Bubble

{

**int**[] a = **new** **int**[5];

Bubble()

{

a[0] = -5;

a[1] = 45;

a[2] = 0;

a[3] = 9;

a[4] = -15;

}

**void** bubbleSort()

{

**int** temp;

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

{

**for**(**int** j=0;j<a.length-1;j++)

{

**if**(a[j+1] < a[j])

{

temp = a[j+1];

a[j+1] = a[j];

a[j] = temp;

}

}

}

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

{

System.***out***.print(a[i]+" ");

}

}

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

{

Bubble obj = **new** Bubble();

obj.bubbleSort();

}

}

2.Selection sort

**public** **class** Selection

{

**int** a[] = **new** **int**[5];

Selection()

{

a[0] = 2;

a[1] = 1;

a[2] = 9;

a[3] = 10;

a[4] = 0;

}

**void** selectionSort()

{

**int** min;

**int** minIndex;

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

{

min = a[i];

minIndex=i;

**for**(**int** k=0;k<a.length;k++)

{

System.***out***.print(a[k] + " ");

}

**for**(**int** j=i+1;j<a.length;j++)

{

**if**(a[j] < min)

{

min = a[j];

minIndex = j;

}

}

a[minIndex] = a[i];

a[i]= min;

System.***out***.println();

}

}

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

{

Selection obj = **new** Selection();

obj.selectionSort();

}

}

3.Insertion sort

**public** **class** Insertion

{

**public** **static** **int**[] sort(**int** a[])

{

**for**(**int** i=1;i<a.length;i++)

{

**int** key = a[i];

**int** j = i-1;

**while**(j>=0 && a[j]>key)

{

a[j+1] = a[j];

j--;

}

a[j+1] = key;

}

**return** a;

}

**public** **static** **void** array(**int** a[])

{

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

{

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

}

}

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

{

**int**[] num = {16, 9, 2, 5, 1};

*sort*(num);

*array*(num);

}

}

4.Quick sort

**public** **class** Quick

{

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

{

**if**(low > high)

{

**return**;

}

**int** mid = low+(high-low)/2;

**int** pivot = a[mid];

**int** i = low;

**int** j = high;

**while**(i <= j)

{

**while**(a[i] < pivot)

i++;

**while**(a[j] > pivot)

j--;

**if**(i <= j)

{

**int** temp = a[i];

a[i] = a[j];

a[j] = temp;

i++;

j--;

}

}

**if**(low < j)

*sort*(a, low, j);

**if**(high > i)

*sort*(a, i, high);

}

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

{

**int** a[] = {30, 50, 20, 10, 40};

*sort*(a, 0, a.length-1);

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

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

}

}

5.Merge sort

**class** Merge

{

**void** merge(**int** arr[], **int** l, **int** m, **int** r)

{

**int** n1 = m - l + 1;

**int** n2 = r - m;

**int** L[] = **new** **int** [n1];

**int** R[] = **new** **int** [n2];

**for** (**int** i=0; i<n1; ++i)

L[i] = arr[l + i];

**for** (**int** j=0; j<n2; ++j)

R[j] = arr[m + 1+ j];

**int** k = l;

**while** (i < n1 && j < n2)

{

**if** (L[i] <= R[j])

{

arr[k] = L[i];

i++;

}

**else**

{

arr[k] = R[j];

j++;

}

k++;

}

**while** (i < n1)

{

arr[k] = L[i];

i++;

k++;

}

**while** (j < n2)

{

arr[k] = R[j];

j++;

k++;

}

}

**void** sort(**int** arr[], **int** l, **int** r)

{

**if** (l < r)

{

**int** m = (l+r)/2;

sort(arr, l, m);

sort(arr , m+1, r);

merge(arr, l, m, r);

}

}

**static** **void** printArray(**int** arr[])

{

**int** n = arr.length;

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

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

System.***out***.println();

}

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

{

**int** arr[] = {12, 11, 13, 5, 6, 7};

System.***out***.println("Given Array");

*printArray*(arr);

Merge ob = **new** Merge();

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

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

*printArray*(arr);

}

}

6.Binary search

**public** **class** Binary

{

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

{

**int** arr[] = {10, 20, 30, 50, 100};

**int** searchElement = 10;

**int** index = *binarySearch*(arr, 0, arr.length-1, searchElement);

System.***out***.println("Element found at index "+index);

}

**private** **static** **int** binarySearch(**int**[] arr, **int** low, **int** high, **int** searchElement)

{

**if**(high >= low)

{

**int** mid = low+((high - low)/2);

**if**(arr[mid]==searchElement)

**return** mid;

**if**(searchElement > arr[mid])

{

//search the element in right side of array

**return** *binarySearch*(arr, mid+1, high, searchElement);

}

**if**(searchElement < arr[mid])

{

//search the element in left side of array

**return** *binarySearch*(arr, low, mid-1, searchElement);

}

}

**return** -1;

}

}

7.Linear search

**public** **class** Linear

{

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

{

**int** a[] = {10, 20, 30, 40, 50};

**int** b = 40;

**for**(**int** i=0;i<a.length-1;i++)

{

**if**(b == a[i])

{

System.***out***.println("Element found at the index of "+ i);

}

}

}

}