1. Writing a program in Java implementing the linear search algorithm.

Source code:

```
package SortingAndSearching;

import java.util.Scanner;

public class LinearSearch {

	public static void main(String[] args) {
		int arr[]={55,25,76,43,29,89,77,19};
		Scanner scan=new Scanner(System.in);
		System.out.println("Enter a value to be searched");
		int key=scan.nextInt();
		//search for key
		boolean found=false;
		for(int val : arr){
			if(val==key){
				found=true;
				break;
			}

		}

		if(found==true)
		System.out.println("Value is present");

		else
			System.out.println("Value is not present");

	}


}
```

2. Writing a program in Java implementing the binary search algorithm.

Source code:

```
package SortingAndSearching;

import java.util.Scanner;

public class BinarySearch {

	public static void main(String[] args) {
		int arr[] = { 45, 67, 23, 89, 9 };
		SelectionSort.sort(arr);

		for(int i=0;i<arr.length;i++){
			System.out.print(arr[i]+" ");
```

```
		}


		Scanner scan=new Scanner(System.in);
		System.out.println("\nEnter a value to be searched");
		int key=scan.nextInt();


		boolean ans=binarySearching(arr,key);
		if(ans)
			System.out.println("Value present");
		else
			System.out.println("Value not present");

	}

	public static boolean binarySearching(int arr[],int key){
		boolean ans=false;
		int st=0;
		int end=arr.length-1;
		int mid;
		while(st<=end){
			mid=(st+end)/2;
			if(arr[mid]==key){
				ans=true; break;
			}
			else
				if(arr[mid]<key)
					st=mid+1;
				else
					end=mid-1;

		}

	return ans;
	}

}
```

3. Writing a program in Java implementing the exponential search algorithm.

Source code:

```
package SortingAndSearching;

import java.util.Arrays;

class ExponentialSearch
{

	static int exponentialSearch(int arr[],
							int n, int x)
	{
```

```
            if (arr[0] == x)
                  return 0;


            int i = 1;
            while (i < n && arr[i] <= x)
                  i = i*2;


            return Arrays.binarySearch(arr, i/2,
                                             Math.min(i, n-1), x);
      }


      public static void main(String args[])
      {
            int arr[] = {2, 3, 4, 10, 40};
            int x = 10;
            int result = exponentialSearch(arr,
                                                   arr.length, x);

            System.out.println((result < 0) ?
            "Element is not present in array" :
            "Element is present at index " +
                                            result);
      }
}
```

4. Writing a program in Java implementing the selection sort algorithm.

Source code:

```
package SortingAndSearching;
import java.util.Scanner;
public class SelectionSort {
      public static void selectionSort(int[] arr) {
            for (int i = 0; i < arr.length - 1; i++) {
                  int index = i;
                  for (int j = i + 1; j < arr.length; j++) {
                        if (arr[j] < arr[index]) {
                              index = j;
                        }
                  }
                  int smallerNumber = arr[index];
                  arr[index] = arr[i];
                  arr[i] = smallerNumber;
      }
}
      public static void main(String a[]) {
            Scanner sc = new Scanner(System.in);
                  System.out.println("Input size of array");
                  int size = sc.nextInt();
                  int ar[] = new int[size];
                        System.out.println("Input numbers in Array");
```

```
                                for (int i = 0; i < ar.length; i++) {
                                        ar[i] = sc.nextInt();
                                }
                                System.out.println("Before Selection Sort");
                                for (int i = 0; i < ar.length; i++) {
                                        System.out.print(i + " ");
                                }
                                System.out.println();
                                selectionSort(ar);
                        System.out.println("After Selection Sort");
                        for (int i = 0; i < ar.length; i++) {
                                System.out.print(i + " ");
                        }
                        sc.close();
                }
}
```

5. Writing a program in Java implementing the bubble sort algorithm.

Source code:

```
package SortingAndSearching;

public class BubbleSort {
        public static void sort(int arr[]) {
                int temp;
                        for(int i=0;i<arr.length;i++){
                                for(int j=0;j<arr.length-i-1;j++){
                                        if(arr[j]>arr[j+1]){
                                                temp=arr[j];
                                                arr[j]=arr[j+1];
                                                arr[j+1]=temp;
                                        }
                                }
                        }
        }

        public static void main(String[] args) {
                int arr[] = { 33, 78, 19, 66, 3 };
                sort(arr);
                for(int i=0;i<arr.length;i++){
                        System.out.print(arr[i]+" ");
                }
        }

}
```

6. Writing a program in Java implementing the insertion sort algorithm.

Source code:

```
package SortingAndSearching;
```

```
public class InsertionSort {

	public static void main(String[] args) {
		int arr[]= {9,12,3,21,44};
		insertionsort(arr);
		for(int i=0;i<arr.length;i++)
			System.out.println(arr[i]);
	}

	private static void insertionsort(int[] arr) {
		for(int j=1;j<arr.length;j++) {
			int key=arr[j];
			int i=j-1;//-1
			while(i>-1 && arr[i]>key) {
				arr[i+1]=arr[i];
				i--;
			}
			arr[i+1]=key;
		}

	}

}
```

7. Writing a program in Java implementing the merge sort algorithm.

Source code:

```
package SortingAndSearching;

import java.util.Arrays;

public class MergeSort {

	 void print(int[] arr){
	  for(int i=0;i<arr.length;i++){
	      System.out.print(arr[i]+" ");
	  }
	  System.out.println();
	}
	int[] merge(int arr1[],int arr2[],int n,int m){
	    int[] result=new int[n+m];
	    int i=0;
	    int j=0;
	    int k=0;
	    while(i<n && j<m){
	        if(arr1[i]<arr2[j]){
	            result[k]=arr1[i];
	            i++;
	        }
	        else if(arr1[i]>arr2[j]){
	            result[k]=arr2[j];
	            j++;
	        }
```

```
            else{
             result[k]=arr1[i];
             i++;
             j++;
            }
            k++;
        }
        while(i<n){
            result[k]=arr1[i];
            i++;k++;
        }
        while(j<m){
            result[k]=arr2[j];
            j++;k++;
        }
        return result;
    }

    public static void main(String []args){
        MergeSort msa=new MergeSort();
        int[] arr1={1,3,5,6};
        int[] arr2={2,4,8,10,20};
        Arrays.sort(arr1);
        Arrays.sort(arr2);
        msa.print(arr1);
        msa.print(arr2);
        int[] result=msa.merge(arr1,arr2,arr1.length,arr2.length);
        msa.print(result);


    }
}
```

8. Writing a program in Java implementing the quick sort algorithm.

Source code:

```
package SortingAndSearching;

public class QuickSort {

	public static void main(String[] args) {
		int data[]= {55,4,23,12,90,78,6};
		quickSort(data,0,data.length-1);
		for(int i=0;i<data.length;i++)
			System.out.print(data[i]+" ");

	}

	private static void quickSort(int[] data, int low, int high) {
		if(low<high) {

			int pivot=partition(data,low,high);
			quickSort(data,low,pivot-1);
```

```
                quickSort(data,pivot+1,high);
        }

    }

    private static int partition(int[] data, int low, int high) {

        int pivot=data[high];

        int i=low-1;

        for(int j=low;j<high;j++) {
            if(data[j]<=pivot) {

                i++;

                int temp=data[i];
                data[i]=data[j];
                data[j]=temp;
            }

        }

        int temp=data[i+1];
        data[i+1]=data[high];
        data[high]=temp;
        return i+1;
    }

}
```