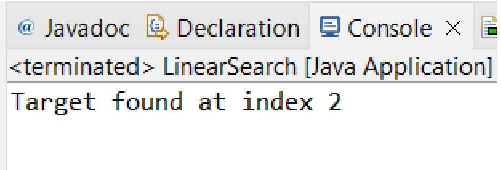
1) WRITING A PROGRAM IN JAVA IMPLEMENTING THE LINEAR SEARCH ALGORITHM

```
package project_4;

public class LinearSearch {

          public static int linearSearch(int[] array, int target) {

              // Iterate through each element in the array
              for (int i = 0; i < array.length; i++) {
                  // If the current element matches the target, return its index

                  if (array[i] == target) {
                      return i;
                  }
              }

              // If the target is not found, return -1
              return -1;
          }

          public static void main(String[] args) {

              int[] array = {9,87,6,32,55,91, 5, 84};
              int target = 6;

              // Call the linearSearch method and store the result
              int index = linearSearch(array, target);

              // Check if the target was found or not
              if (index != -1) {
                  System.out.println("Target found at index " + index);
              } else {
                  System.out.println("Target not found in the array.");
              }
          }
      }
```

Output:

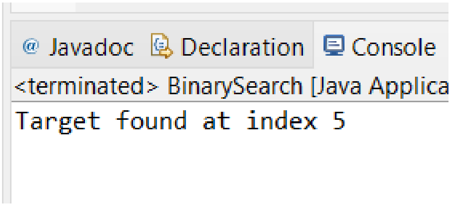


```
<terminated> LinearSearch [Java Application]
Target found at index 2
```

2)WRITING A PROGRAM IN JAVA IMPLEMENTING THE BINARY SEARCH ALGORITHM

```
package project_4;

public class BinarySearch {

          public static int binarySearch(int[] array, int target) {
              int left = 0;
              int right = array.length - 1;

              while (left <= right) {
                  int mid = left + (right - left) / 2;

                  // Check if the target is present at the middle element
                  if (array[mid] == target) {
                      return mid;
                  }

                  // If the target is greater, ignore the left half
                  if (array[mid] < target) {
                      left = mid + 1;
                  }

                  // If the target is smaller, ignore the right half
                  else {
                      right = mid - 1;
                  }
              }

              // If the target is not found, return -1
              return -1;
          }

          public static void main(String[] args) {
              int[] array = {6,9,3,5,7,11,13};
              int target = 11;

              // Call the binarySearch method and store the result
              int index = binarySearch(array, target);

              // Check if the target was found or not
              if (index != -1) {
                  System.out.println("Target found at index " + index);
              } else {
                  System.out.println("Target not found in the array.");
              }
          }
      }
```

Output:

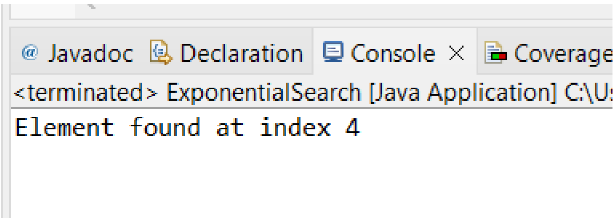


```
<terminated> BinarySearch [Java Applica
Target found at index 5
```

3) WRITING A PROGRAM IN JAVA IMPLEMENTING THE EXPONENTIAL SEARCH ALGORITHM

```java
package project_4;

public class ExponentialSearch {


        public static int exponentialSearch(int[] arr, int target) {
            int length = arr.length;
            if (arr[0] == target) {
                return 0;
            }

            int bound = 1;
            while (bound < length && arr[bound] <= target) {
                bound *= 2;
            }

            int left = bound / 2;
            int right = Math.min(bound, length - 1);

            return binarySearch(arr, target, left, right);
        }

        public static int binarySearch(int[] arr, int target, int left, int right)
{
            while (left <= right) {
                int mid = left + (right - left) / 2;

                if (arr[mid] == target) {
                    return mid;
                }

                if (arr[mid] < target) {
                    left = mid + 1;
                } else {
                    right = mid - 1;
                }
            }

            return -1;
        }

        public static void main(String[] args) {
            int[] arr = {1, 3, 5, 7, 9, 11, 13, 15};
            int target = 9;

            int index = exponentialSearch(arr, target);
            if (index != -1) {
                System.out.println("Element found at index " + index);
            } else {
                System.out.println("Element not found in the array");
            }
        }
    }
```

Output:

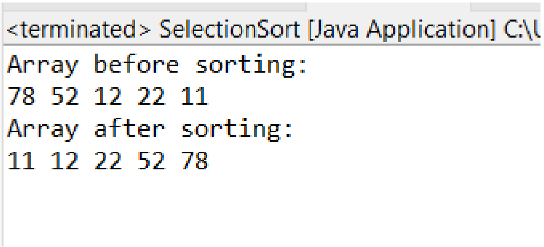


```
<terminated> ExponentialSearch [Java Application] C:\U
Element found at index 4
```

4) WRITING A PROGRAM IN JAVA IMPLEMENTING THE SELECTION SORT ALGORITHM

```java
package project_4;

public class SelectionSort {
          public static void selectionSort(int[] arr) {
              int length = arr.length;

              for (int i = 0; i < length - 1; i++) {
                  int minIndex = i;
                  for (int j = i + 1; j < length; j++) {
                      if (arr[j] < arr[minIndex]) {
                          minIndex = j;
                      }
                  }

                  // Swap the found minimum element with the first element
                  int temp = arr[minIndex];
                  arr[minIndex] = arr[i];
                  arr[i] = temp;
              }
          }

          public static void main(String[] args) {
              int[] arr = {78, 52, 12, 22, 11};
              System.out.println("Array before sorting:");
              printArray(arr);

              selectionSort(arr);

              System.out.println("Array after sorting:");
              printArray(arr);
          }

          public static void printArray(int[] arr) {
              for (int i = 0; i < arr.length; i++) {
                  System.out.print(arr[i] + " ");
              }
              System.out.println();
          }
      }
```

Output:



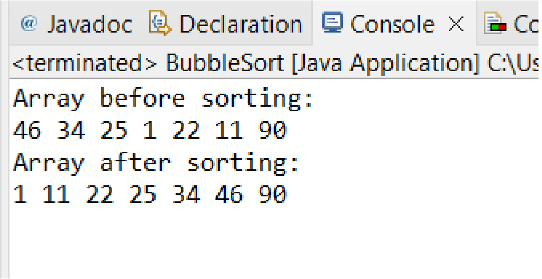
<terminated> SelectionSort [Java Application] C:\U

```
Array before sorting:
78 52 12 22 11
Array after sorting:
11 12 22 52 78
```

5) WRITING A PROGRAM IN JAVA IMPLEMENTING THE BUBBLE SORT ALGORITHM

```java
package project_4;

public class BubbleSort {
        public static void bubbleSort(int[] arr) {
            int length = arr.length;

            for (int i = 0; i < length - 1; i++) {
                for (int j = 0; j < length - i - 1; j++) {
                    if (arr[j] > arr[j + 1]) {
                        // Swap arr[j] and arr[j+1]
                        int temp = arr[j];
                        arr[j] = arr[j + 1];
                        arr[j + 1] = temp;
                    }
                }
            }
        }

        public static void main(String[] args) {
            int[] arr = {46, 34, 25, 1, 22, 11, 90};
            System.out.println("Array before sorting:");
            printArray(arr);

            bubbleSort(arr);

            System.out.println("Array after sorting:");
            printArray(arr);
        }

        public static void printArray(int[] arr) {
            for (int i = 0; i < arr.length; i++) {
                System.out.print(arr[i] + " ");
            }
            System.out.println();
        }
    }
```

Output:

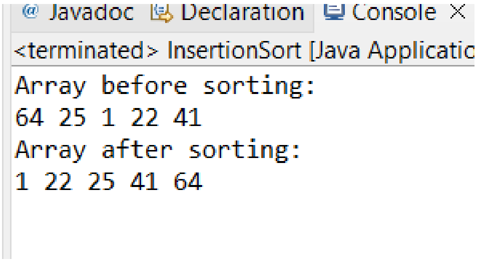


```
<terminated> BubbleSort [Java Application] C:\Us
Array before sorting:
46 34 25 1 22 11 90
Array after sorting:
1 11 22 25 34 46 90
```

6) WRITING A PROGRAM IN JAVA IMPLEMENTING THE INSERTION SORT ALGORITHM

```
package project_4;

public class InsertionSort {

           public static void insertionSort(int[] arr) {
               int length = arr.length;

               for (int i = 1; i < length; i++) {
                   int key = arr[i];
                   int j = i - 1;

                   while (j >= 0 && arr[j] > key) {
                       arr[j + 1] = arr[j];
                       j--;
                   }

                   arr[j + 1] = key;
               }
           }

           public static void main(String[] args) {
               int[] arr = {64, 25, 1, 22, 41};
               System.out.println("Array before sorting:");
               printArray(arr);

               insertionSort(arr);

               System.out.println("Array after sorting:");
               printArray(arr);
           }

           public static void printArray(int[] arr) {
               for (int i = 0; i < arr.length; i++) {
                   System.out.print(arr[i] + " ");
               }
               System.out.println();
           }
       }
```

Output:



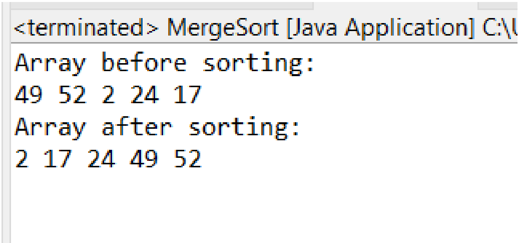
```
<terminated> InsertionSort [Java Applicatic
Array before sorting:
64 25 1 22 41
Array after sorting:
1 22 25 41 64
```

7) WRITING A PROGRAM IN JAVA IMPLEMENTING THE MERGE SORT ALGORITHM

```java
package project_4;

public class MergeSort {
        public static void mergeSort(int[] arr) {
            int length = arr.length;

            if (length < 2) {
                return; // Base case: array is already sorted
            }

            int mid = length / 2;
            int[] left = new int[mid];
            int[] right = new int[length - mid];

            // Fill the left and right subarrays
            for (int i = 0; i < mid; i++) {
                left[i] = arr[i];
            }
            for (int i = mid; i < length; i++) {
                right[i - mid] = arr[i];
            }

            mergeSort(left); // Recursively sort the left subarray
            mergeSort(right); // Recursively sort the right subarray

            merge(arr, left, right); // Merge the sorted subarrays
        }

        public static void merge(int[] arr, int[] left, int[] right) {
            int leftLength = left.length;
            int rightLength = right.length;
            int i = 0, j = 0, k = 0;

            // Merge the left and right subarrays into the original array
            while (i < leftLength && j < rightLength) {
                if (left[i] <= right[j]) {
                    arr[k++] = left[i++];
                } else {
                    arr[k++] = right[j++];
                }
            }

            // Copy the remaining elements of the left subarray, if any
            while (i < leftLength) {
                arr[k++] = left[i++];
            }

            // Copy the remaining elements of the right subarray, if any
            while (j < rightLength) {
                arr[k++] = right[j++];
            }
        }
```

```java
    public static void main(String[] args) {
        int[] arr = {49, 52, 2, 24, 17};
        System.out.println("Array before sorting:");
        printArray(arr);

        mergeSort(arr);

        System.out.println("Array after sorting:");
        printArray(arr);
    }

    public static void printArray(int[] arr) {
        for (int i = 0; i < arr.length; i++) {
            System.out.print(arr[i] + " ");
        }
        System.out.println();
    }
}
```

Output:

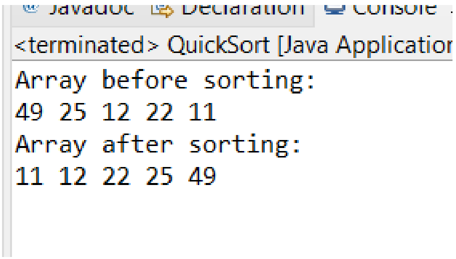


```
<terminated> MergeSort [Java Application] C:\(
Array before sorting:
49 52 2 24 17
Array after sorting:
2 17 24 49 52
```

8) WRITING A PROGRAM IN JAVA IMPLEMENTING THE QUICK SORT ALGORITHM

```java
package project_4;

public class QuickSort {
        public static void quickSort(int[] arr, int low, int high) {
            if (low < high) {
                int pivotIndex = partition(arr, low, high);
                quickSort(arr, low, pivotIndex - 1);
                quickSort(arr, pivotIndex + 1, high);
            }
        }

        public static int partition(int[] arr, int low, int high) {
            int pivot = arr[high];
            int i = low - 1;

            for (int j = low; j < high; j++) {
                if (arr[j] < pivot) {
                    i++;
                    swap(arr, i, j);
                }
            }

            swap(arr, i + 1, high);
            return i + 1;
        }

        public static void swap(int[] arr, int i, int j) {
            int temp = arr[i];
            arr[i] = arr[j];
            arr[j] = temp;
        }

        public static void main(String[] args) {
            int[] arr = {49, 25, 12, 22, 11};
            System.out.println("Array before sorting:");
            printArray(arr);

            quickSort(arr, 0, arr.length - 1);

            System.out.println("Array after sorting:");
            printArray(arr);
        }

        public static void printArray(int[] arr) {
            for (int i = 0; i < arr.length; i++) {
                System.out.print(arr[i] + " ");
            }
            System.out.println();
        }
    }
```

Output:



```
Array before sorting:
49 25 12 22 11
Array after sorting:
11 12 22 25 49
```