1) Write a program to sort an array in descending order using bubble sort.

Ans:

```
public class Sorting {

    public static void bubbleSort(int[] arr) {

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

    }

    public static void main(String[] args) {
        int nums[] = { 3, 5, 1, 6, 0 };

        bubbleSort(nums);

        for (int i : nums) {
            System.out.print(i + " ");
        }

        System.out.println();
    }
}
```

2) WAP to sort an array in descending order using selection sort.

Ans:

```
public class Sorting {

    public static void selectionSort(int[] arr) {
        for (int i = 0; i < arr.length; i++) {
            int minIndex = i;
            for (int j = i + 1; j < arr.length; j++) {
                if (arr[minIndex] < arr[j]) {
                    minIndex = j;
                }
            }
            if (minIndex != i) {
                int temp = arr[minIndex];
                arr[minIndex] = arr[i];
                arr[i] = temp;
            }
        }
    }

    public static void main(String[] args) {
        int nums[] = { 3, 2, 33, 5, 11, 6, 0 };

        selectionSort(nums);

        for (int i : nums) {
            System.out.print(i + " ");
        }

        System.out.println();
    }
}
```

3) WAP to sort an array in descending order using insertion sort.

Ans:

```
public class InsertionSortDescending {
    public static void main(String[] args) {
        int[] arr = {5, 2, 9, 1, 5, 6};

        insertionSortDescending(arr);
```

```
        // Print the sorted array
        System.out.println("Sorted Array in Descending Order:");
        for (int num : arr) {
            System.out.print(num + " ");
        }
    }

    public static void insertionSortDescending(int[] arr) {
        int n = arr.length;
        for (int i = 1; i < n; i++) {
            int key = arr[i];
            int j = i - 1;

            // Move elements of arr[0..i-1] that are greater than key
            // to one position ahead of their current position
            while (j >= 0 && arr[j] < key) {
                arr[j + 1] = arr[j];
                j--;
            }

            arr[j + 1] = key;
        }
    }
}
```

4) Find how many passes would be required to sort the following array in decreasing order using bubble sort . Input array { 3, 5, 1, 6, 0}

Ans:

To find out how many passes are required to sort the given array in decreasing order using the bubble sort algorithm, we can perform the sorting process step by step and count the number of passes it takes until the array is fully sorted. In each pass, the largest element "bubbles up" to its correct position at the beginning of the array.

For the given array {3, 5, 1, 6, 0}:

1. Pass 1: Compare adjacent elements and swap them if they are in the wrong order.

- Initial Array: {3, 5, 1, 6, 0}
- After Pass 1: {5, 3, 6, 1, 0}

2. Pass 2: Continue the process for the remaining unsorted portion.
   - After Pass 2: {5, 6, 3, 1, 0}

3. Pass 3:
   - After Pass 3: {6, 5, 3, 1, 0}

4. Pass 4:
   - After Pass 4: {6, 5, 3, 1, 0}

5. Pass 5:
   - After Pass 5: {6, 5, 3, 1, 0}

After the 5th pass, the array is still not sorted in decreasing order, but you can see that the largest element, 6, has moved to its correct position at the beginning of the array in the first pass. To sort the entire array in descending order, you would need one more pass, but it won't change the order of the elements.

So, it would take a total of 6 passes to sort the given array {3, 5, 1, 6, 0} in decreasing order using the bubble sort algorithm.

5) Find out the number of iterations to sort the array in descending order using selection sort .

Input Array { 3, 5, 1, 6, 0 }

Ans:

To find out the number of iterations required to sort the given array in descending order using the selection sort algorithm, wecan perform the sorting process step by step and count the number of iterations it takes to complete the sorting.

For the given array {3, 5, 1, 6, 0}:

1. Iteration 1: Find the maximum element and swap it with the last element.
   - Initial Array: {3, 5, 1, 6, 0}
   - After Iteration 1: {3, 5, 1, 0, 6}

2. Iteration 2: Find the maximum element from the remaining unsorted portion (excluding the last element) and swap it with the second-to-last element.
   - After Iteration 2: {3, 1, 0, 5, 6}

3. Iteration 3: Find the maximum element from the remaining unsorted portion (excluding the last two elements) and swap it with the third-to-last element.
   - After Iteration 3: {0, 1, 3, 5, 6}

4. Iteration 4: Find the maximum element from the remaining unsorted portion (excluding the last three elements) and swap it with the fourth-to-last element.
   - After Iteration 4: {0, 1, 3, 5, 6}

5. Iteration 5: Find the maximum element from the remaining unsorted portion (excluding the last four elements) and swap it with the fifth-to-last element (the first element).
   - After Iteration 5: {6, 1, 3, 5, 0}

6. Iteration 6: Find the maximum element from the remaining unsorted portion (excluding the last five elements) and swap it with the sixth-to-last element (the second element).
   - After Iteration 6: {6, 5, 3, 1, 0}

The array is now fully sorted in descending order. It took a total of 6 iterations to sort the given array {3, 5, 1, 6, 0} in descending order using the selection sort algorithm.