**Assignment** 2: Array Manipulation in Java

**Duration: 30 minutes**

**Assignment Description**:

This  **Assignment** focuses on more advanced array operations and problem-solving using arrays. Students are expected to implement and analyze various array manipulation algorithms.

**Questions**:

Maximum Subarray Sum: Write a Java function that finds the maximum subarray sum in a given array of integers. Explain the algorithm and analyze its time complexity.

Solution:

java

public static int maxSubarraySum(int[] arr) {

int maxSum = arr[0];

int currentSum = arr[0];

for (int i = 1; i < arr.length; i++) {

currentSum = Math.max(arr[i], currentSum + arr[i]);

maxSum = Math.max(maxSum, currentSum);

}

return maxSum;

}

Remove Duplicates: Implement a function in Java that removes duplicates from an array of integers. Explain the approach and analyze its efficiency.

Solution:

java

public static int[] removeDuplicates(int[] arr) {

Set<Integer> uniqueSet = new LinkedHashSet<>();

for (int value : arr) {

uniqueSet.add(value);

}

int[] result = new int[uniqueSet.size()];

int index = 0;

for (int value : uniqueSet) {

result[index++] = value;

}

return result;

}

Rotate Array: Design and implement an algorithm in Java to rotate a given array of integers to the right by a specified number of positions. Explain the time complexity of your solution.

Solution:

java

public static void rotateArray(int[] arr, int k) {

int n = arr.length;

k = k % n;

reverseArray(arr, 0, n - 1);

reverseArray(arr, 0, k - 1);

reverseArray(arr, k, n - 1);

}

private static void reverseArray(int[] arr, int left, int right) {

while (left < right) {

int temp = arr[left];

arr[left] = arr[right];

arr[right] = temp;

left++;

right--;

}

}