DSA PRACTICE PROBLEMS – SET 5

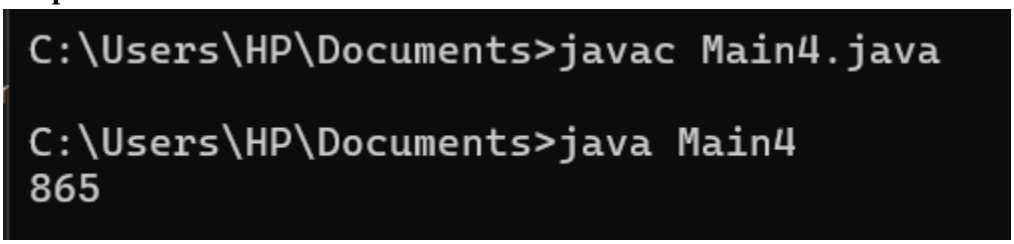
Madhulekha R – AIML - [14/11/2024]

1. **Question:** Stock buy and sell

   Code:

```
import java.util.*;

class Main4{
   public static void main(String[] args) {
      int[] prices = {100, 180, 260, 310, 40, 535, 695};
      System.out.println(maxprofit(prices));
   }

   static int maxprofit(int[] prices) {
      int n = prices.length;
      int lMin = prices[0];
      int lMax = prices[0];
      int res = 0;

      int i = 0;
      while (i < n-1) {
         while (i < n-1 && prices[i] >= prices[i+1]){
           i++;
         }
         lMin = prices[i];
         while (i < n - 1 && prices[i] <= prices[i + 1]){
           i++;
         }
         lMax = prices[i];
         res += (lMax - lMin);
      }
      return res;
   }
}
```

Output:



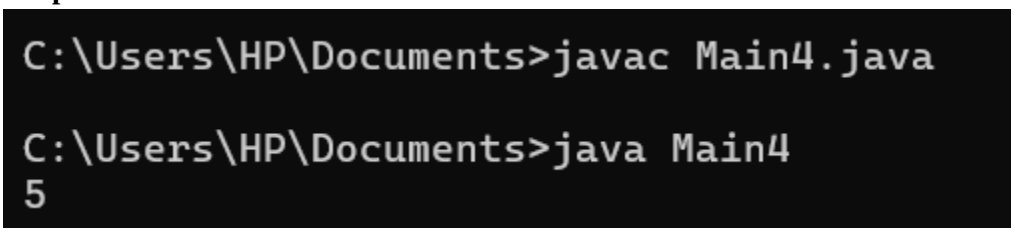
Time Complexity: O(n)
Space Complexity: O(1)

2. **Question:** Coin change

Code:

```
import java.util.*;

public class Main4{
   public static void main(String[] args) {
      List<Integer> coins = Arrays.asList(1, 2, 3);
      int n = 3;
      int sum = 5;
      System.out.println(count(coins, n, sum));
   }

   static int count(List<Integer> coins, int n, int sum){
      int[][] dp = new int[n + 1][sum + 1];
      dp[0][0] = 1;
      for (int i = 1; i <= n; i++){
         for (int j = 0; j <= sum; j++){
            dp[i][j] += dp[i - 1][j];

            if ((j - coins.get(i - 1)) >= 0){
               dp[i][j] += dp[i][j - coins.get(i - 1)];
            }
         }
      }
      return dp[n][sum];
   }
}
```

Output:



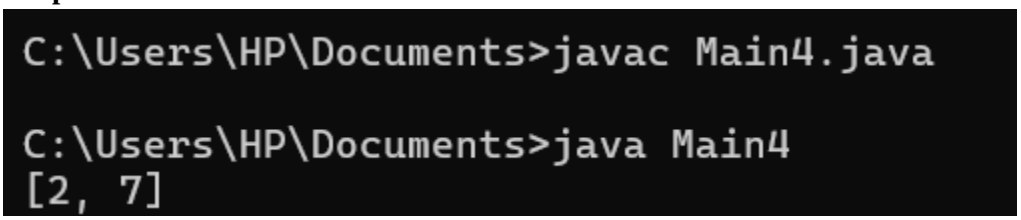
Time Complexity: O(n*sum)
Space Complexity: O(n*sum)

3. **Question:** First and last occurrences of an element

Code:

```
import java.util.*;

public class Main4{
  public static void main(String[] arg){
    int[] arr = {1,2,4,45,15,21,36,4};
    int target=4;
    System.out.println(Arrays.toString(func(arr, target)));
  }

  static int[] func(int[] arr, int target){
    int firstind=-1;
    int lastind=-1;

    for (int i=0; i<arr.length; i++){
      if (arr[i]==target){
        if (firstind==-1){
          firstind=i;
        }
        lastind=i;
      }
    }
    return new int[]{firstind, lastind};
  }
}
```

Output:



Time Complexity: O(N)
Space Complexity: O(1)

4. **Question:** Find transition point

Code:

```
import java.util.*;

class Main4{
  public static void main(String args[]){
    int arr[] = { 0, 0, 0, 0, 1, 1 };
    int point = findTransitionPoint(arr, arr.length);
    System.out.println(point >= 0 ? "Transition point is " + point : "There is no transition point");
  }

  static int findTransitionPoint(int arr[], int n){
    int lb = 0, ub = n - 1;
    while (lb <= ub){
      int mid = (lb + ub) / 2;
          if (arr[mid] == 0){
            lb = mid + 1;
      }
          else if(arr[mid] == 1){
            if (mid == 0 || (mid > 0 && arr[mid - 1] == 0)){
              return mid;
         }
            ub = mid - 1;
      }
    }
    return -1;
  }
}
```

Output:

```
C:\Users\HP\Documents>javac Main4.java

C:\Users\HP\Documents>java Main4
Transition point is 4
```

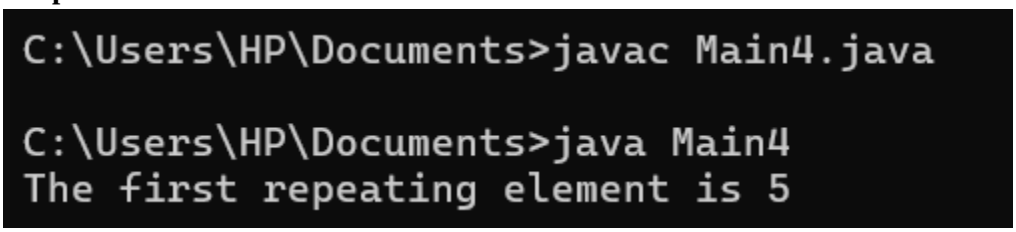
Time Complexity: O(log n)
Space Complexity: O(1)

5. **Question:** First repeating element

Code:

```
import java.util.*;

class Main4{
   public static void main(String[] args) throws java.lang.Exception{
      int arr[] = { 10, 5, 3, 4, 3, 5, 6 };
      printFirstRepeating(arr);
   }

   static void printFirstRepeating(int arr[]){
      int min = -1;
      HashSet<Integer> set = new HashSet<>();
      for (int i = arr.length - 1; i >= 0; i--){
         if (set.contains(arr[i])){
            min = i;
         }
         else{
            set.add(arr[i]);
         }
      }

      if (min != -1){
         System.out.println("The first repeating element is " + arr[min]);
      }
      else {
         System.out.println("There are no repeating elements");
      }
   }
}
```

Output:



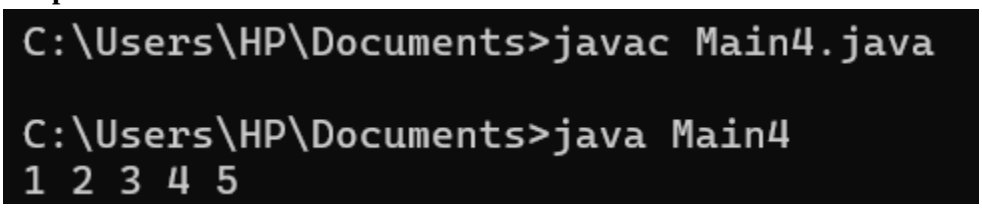
Time Complexity: O(n)
Space Complexity: O(n)

6. **Question:** Remove Duplicates from Sorted array

Code:

```
import java.util.*;

class Main4{
  public static void main(String[] args){
      int[] arr = {1, 2, 2, 3, 4, 4, 4, 5, 5};
      int newSize = removeDuplicates(arr);
      for (int i = 0; i < newSize; i++) {
          System.out.print(arr[i] + " ");
      }
   }

  static int removeDuplicates(int[] arr){
    int n = arr.length;
    if (n <= 1){
      return n;
    }
    int idx = 1;
    for (int i = 1; i < n; i++){
      if (arr[i] != arr[i - 1]){
        arr[idx++] = arr[i];
      }
     }
     return idx;
   }
}
```

Output:



Time Complexity: O(n)
Space Complexity: O(1)

7. **Question:** Maximum Index

Code:

```
import java.util.*;

class Main4{
  public static void main(String[] args){
    int n = 9;
    int[] arr = {34, 8, 10, 3, 2, 80, 30, 33, 1};
    System.out.println(func(arr, n));
  }

  static int func(int[] arr, int n){
    int[] mn = new int[n];
    int[] mx = new int[n];
    mn[0] = arr[0];
    for (int i = 1; i < n; i++){
      mn[i] = Math.min(mn[i - 1], arr[i]);
    }

    mx[n - 1] = arr[n - 1];
    for (int i = n - 2; i >= 0; i--){
      mx[i] = Math.max(mx[i + 1], arr[i]);
    }

    int i = 0, j = 0, ans = -1;
    while (i < n && j < n){
      if (mn[i] <= mx[j]){
        ans = Math.max(ans, j - i);
        j += 1;
      }
      else {
        i += 1;
```

```
        }
      }
      return ans;
    }
}
```

Output:

```
C:\Users\HP\Documents>javac Main4.java

C:\Users\HP\Documents>java Main4
6
```

Time Complexity: O(n)
Space Complexity: O(n)

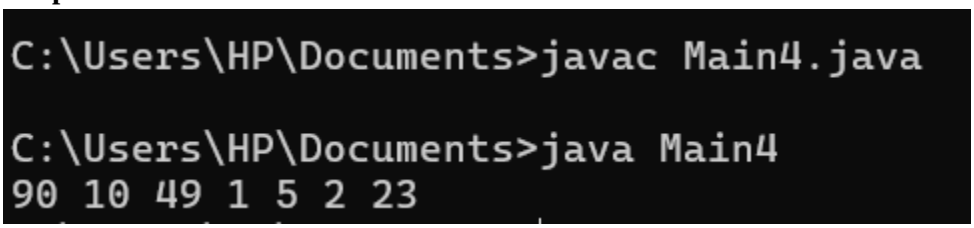
8. **Question:** Wave Array

Code:

```
import java.util.*;

class Main4 {
  public static void main(String args[]) {
     int arr[] = {10, 90, 49, 2, 1, 5, 23};
     int n = arr.length;
     int[] arr1 = sortInWave(arr, n);
     for (int i : arr1) {
        System.out.print(i + " ");
     }
  }

  static void swap(int arr[], int a, int b) {
     int temp = arr[a];
     arr[a] = arr[b];
     arr[b] = temp;
  }

  static int[] sortInWave(int arr[], int n){
     for (int i = 0; i < n; i += 2){
        if (i > 0 && arr[i - 1] > arr[i]){
           swap(arr, i, i - 1);
        }
```

```
            if (i < n - 1 && arr[i + 1] > arr[i]) {
                swap(arr, i, i + 1);
            }
        }
        return arr;
    }
}
```

Output:



Time Complexity: O(n)
Space Complexity: O(n)