

DSA- PROBLEMS

14/11/2024

Stock Buy and sell:

```
import java.util.ArrayList;
import java.util.List;
class StockBuySell {
    static class Interval {
        int buy, sell;
        Interval(int buy, int sell) {
            this.buy = buy;
            this.sell = sell;
        }
    }
    public static List<Interval> stockBuySell(int[] prices) {
        List<Interval> result = new ArrayList<>();
        int n = prices.length;
        int i = 0;
        while (i < n - 1) {
            // Find the local minima
            while (i < n - 1 && prices[i + 1] <= prices[i])
                i++;
            if (i == n - 1)
                break;
            int buy = i++;
            while (i < n && prices[i] >= prices[i - 1])
```

```

        i++;

        int sell = i - 1;

        result.add(new Interval(buy, sell));
    }

    return result;
}

public static void main(String[] args) {
    int[] prices = {100, 180, 260, 310, 40, 535, 695};

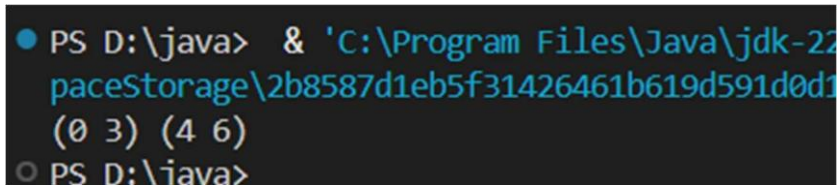
    List<Interval> result = stockBuySell(prices);

    if (result.isEmpty()) {
        System.out.println("No Profit");
    } else {
        for (Interval interval : result) {
            System.out.print("(" + interval.buy + " " + interval.sell + ") ");
        }

        System.out.println();
    }
}
}

```

OUTPUT:



```

PS D:\java> java -cp . StockBuySell
(0 3) (4 6)
PS D:\java>

```

Coin change:

```
import java.util.*;
```

```
public class CoinChange {

    public static int helper(List<Integer> coins, int n, int sum, int[][] memo) {
        if (sum == 0)
            return memo[n][sum] = 1;
        if (n == 0 || sum < 0)
            return 0;
        if (memo[n][sum] != -1)
            return memo[n][sum];
        return memo[n][sum] = helper(coins, n, sum - coins.get(n - 1), memo) + helper(coins, n - 1, sum,
memo);
    }

    public static int count(List<Integer> coins, int n, int sum) {
        int[][] memo = new int[n + 1][sum + 1];
        for (int[] row : memo)
            Arrays.fill(row, -1);
        return helper(coins, n, sum, memo);
    }

    public static void main(String[] args) {
        int n = 4, sum = 10;
        List<Integer> coins = Arrays.asList(2, 5, 3, 6);
        int res = count(coins, n, sum);
        System.out.println(res);
    }
}
```

OUTPUT:

```
PS D:\java> & 'C:\Program Files\Java\jdk-22\bin\java.exe' '-XX:+ShowCodeDetailsInExceptionMessages' '-
paceStorage\2b8587d1eb5f31426461b619d591d0d1\redhat.java\jdt_ws\java_72e8cc1b\bin' 'CoinChange'
5
PS D:\java>
```

First and Last Occurences:

```
class FirstLastOccurrence {

    public static int first(int[] arr, int low, int high, int x, int n) {

        if (high >= low) {

            int mid = low + (high - low) / 2;

            if ((mid == 0 || x > arr[mid - 1]) && arr[mid] == x)

                return mid;

            else if (x > arr[mid])

                return first(arr, mid + 1, high, x, n);

            else

                return first(arr, low, mid - 1, x, n);

        }

        return -1;

    }

}
```

```
public static int last(int[] arr, int low, int high, int x, int n) {

    if (high >= low) {

        int mid = low + (high - low) / 2;

        if ((mid == n - 1 || x < arr[mid + 1]) && arr[mid] == x)

            return mid;

        else if (x < arr[mid])

            return last(arr, low, mid - 1, x, n);

        else

            return last(arr, mid + 1, high, x, n);

    }

}
```

```

    }

    return -1;
}

public static void main(String[] args) {
    int[] arr = {1, 2, 2, 2, 2, 3, 4, 7, 8, 8};
    int n = arr.length;
    int x = 8;

    System.out.println("First Occurrence = " + first(arr, 0, n - 1, x, n));
    System.out.println("Last Occurrence = " + last(arr, 0, n - 1, x, n));
}
}

```

OUTPUT:



```

● PS D:\java> & 'C:\Program Files\Java\jdk-22\bin\java.exe' '-XX:+ShowCodeDetails
paceStorage\2b8587d1eb5f31426461b619d591d0d1\redhat.java\jdt_ws\java_72e8cc1b\bi
First Occurrence = 8
Last Occurrence = 9
PS D:\java>

```

Find Transition Point:

```

import java.util.List;

class TransitionPoint {

    public static int transitionPoint(List<Integer> arr) {
        if (arr.get(arr.size() - 1) == 0) return -1;
        if (arr.get(0) == 1) return 0;
        int left = 0, right = arr.size() - 1, mid = 0;

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

```

```

        if (mid > 0 && arr.get(mid) == 1 && arr.get(mid - 1) == 0) return mid;

        else if (arr.get(mid) == 1) right = mid - 1;

        else left = mid + 1;

    }

    return -1;
}

```

```

public static void main(String[] args) {

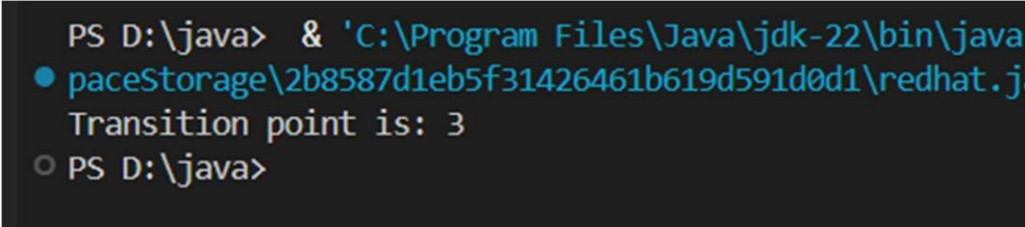
    List<Integer> arr = List.of(0, 0, 0, 1, 1);

    System.out.println("Transition point is: " + transitionPoint(arr));

}
}

```

OUTPUT:



```

PS D:\java> & 'C:\Program Files\Java\jdk-22\bin\java
● -cp 'C:\Program Files\Java\jdk-22\bin\java\classes;C:\Program Files\Java\jdk-22\bin\java\lib\redhat.jar' com.example.Redhat
Transition point is: 3
○ PS D:\java>

```

First Repeating element:

```

import java.util.HashMap;

import java.util.Map;

import java.util.List;

import java.util.Arrays;

public class FirstRepeated {

    public static int firstRepeated(List<Integer> arr) {

        int minn = Integer.MAX_VALUE;
    }
}

```


Remove Duplicates - Sorted Array:

```
import java.util.ArrayList;

public class RemoveDuplicates {

    public static int removeDuplicates(ArrayList<Integer> arr) {

        int n = arr.size();

        if (n <= 1) return n;

        int ind = 1;

        for (int i = 1; i < n; i++) {

            if (!arr.get(i).equals(arr.get(i - 1))) {

                arr.set(ind++, arr.get(i));

            }

        }

        return ind;

    }

    public static void main(String[] args) {

        ArrayList<Integer> arr = new ArrayList<>();

        arr.add(1);

        arr.add(2);

        arr.add(2);

        arr.add(3);

        arr.add(4);

        arr.add(4);

        arr.add(4);

        arr.add(5);

        arr.add(5);

    }

}
```



```

int n = removeDuplicates(arr);

for (int i = 0; i < n; i++) {

    System.out.print(arr.get(i) + " ");

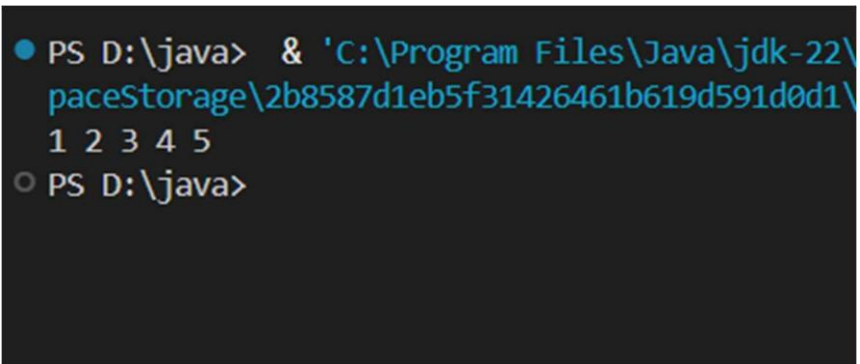
}

}

}

```

OUTPUT:



```

PS D:\java> & 'C:\Program Files\Java\jdk-22\bin\java.exe' -cp 'D:\java\src\MaxIndexDiff.class' -Djava.class.path='D:\java\src\MaxIndexDiff.class' 1 2 3 4 5
1 2 3 4 5
PS D:\java>

```

Maximum Index:

```

import java.util.*;

class MaxIndexDiff {

    public static int maxIndexDiff(int[] arr, int n) {

        Map<Integer, List<Integer>> map = new HashMap<>();

        for (int i = 0; i < n; i++) {

            map.putIfAbsent(arr[i], new ArrayList<>());

            map.get(arr[i]).add(i);

        }

        Arrays.sort(arr);

        int maxDiff = Integer.MIN_VALUE;

        int temp = n;
    }
}

```

```

for (int i = 0; i < n; i++) {
    if (temp > map.get(arr[i]).get(0)) {
        temp = map.get(arr[i]).get(0);
    }
    maxDiff = Math.max(maxDiff, map.get(arr[i]).get(map.get(arr[i]).size() - 1) - temp);
}
return maxDiff;
}

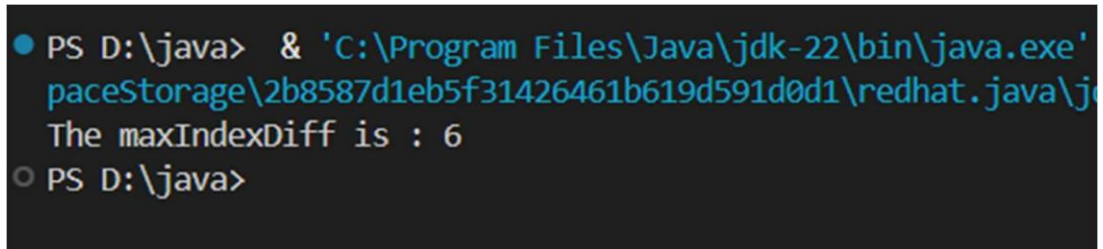
```

```

public static void main(String[] args) {
    int n = 9;
    int[] arr = {34, 8, 10, 3, 2, 80, 30, 33, 1};
    int ans = maxIndexDiff(arr, n);
    System.out.println("The maxIndexDiff is : " + ans);
}
}

```

OUTPUT:



```

PS D:\java> & 'C:\Program Files\Java\jdk-22\bin\java.exe' -cp 'D:\java\paceStorage\2b8587d1eb5f31426461b619d591d0d1\redhat.java\j...'
The maxIndexDiff is : 6
PS D:\java>

```

Wave array:

```

import java.util.Arrays;

public class WaveSort {

    static void sortInWave(int[] arr, int n) {

        Arrays.sort(arr);

        for (int i = 0; i < n - 1; i += 2) {

```

```

    int temp = arr[i];
    arr[i] = arr[i + 1];
    arr[i + 1] = temp;
}
}

```

```
public static void main(String[] args) {  
    int[] arr = {10, 90, 49, 2, 1, 5, 23};  
  
    int n = arr.length;  
  
    sortInWave(arr, n);  
  
    for (int i = 0; i < n; i++)  
        System.out.print(arr[i] + " ");  
  
}
```

OUTPUT:

```
PS D:\java> & 'C:\Program Files\Java\jdk-22\bin\
paceStorage\2b8587d1eb5f31426461b619d591d0d1\redh
2 1 10 5 49 23 90
PS D:\java>
```