# 14.11.2024

1. Stock Buy and Sell

CODE :

import java.util.\*;

class Main {

ArrayList<ArrayList<Integer>> stockBuySell(int prices[], int n) {

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

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

if (prices[i + 1] > prices[i]) {

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

buySellPair.add(i);

buySellPair.add(i + 1);

result.add(buySellPair);

}

}

return result;

}

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter the number of days: ");

int n = scanner.nextInt();

int[] prices = new int[n];

System.out.println("Enter the stock prices for each day:");

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

prices[i] = scanner.nextInt();

}

Main solution = new Main();

ArrayList<ArrayList<Integer>> result = solution.stockBuySell(prices, n);

if (result.isEmpty()) {

System.out.println("0");

} else {

System.out.println("1");

}

}

}

Output:

Enter the number of days: 4

Enter the stock prices for each day:

12 29 32 99

1

Time Complexity: O(n)

Space Complexity :O(k)

2 . Coin Change

CODE:

import java.util.Scanner;

class Main {

public long count(int coins[], int sum) {

int n = coins.length;

long prev[] = new long[sum + 1];

for (int s = 0; s <= sum; s++) {

if (s % coins[0] == 0) prev[s] = 1;

}

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

long curr[] = new long[sum + 1];

for (int s = 0; s <= sum; s++) {

long nTake = prev[s];

long take = 0;

if (coins[ind] <= s) take = curr[s - coins[ind]];

curr[s] = take + nTake;

}

prev = curr.clone();

}

return prev[sum];

}

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter the number of coin denominations: ");

int n = scanner.nextInt();

int[] coins = new int[n];

System.out.println("Enter the coin denominations:");

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

coins[i] = scanner.nextInt();

}

System.out.print("Enter the sum to be formed: ");

int sum = scanner.nextInt();

Main solution = new Main();

long ways = solution.count(coins, sum);

System.out.println("Number of ways to form the sum: " + ways);

scanner.close();

}

}

Output:

Enter the number of coin denominations: 3

Enter the coin denominations:

1 2 3

Enter the sum to be formed: 4

Number of ways to form the sum: 4

Time Complexity : O(n \* sum)

Space Complexity : O(sum)

3. First and Last Occurrences

CODE:

import java.util.\*;

class Main {

ArrayList<Integer> find(int arr[], int x) {

int st = -1;

int end = -1;

for (int i = 0; i < arr.length; i++) {

if (arr[i] == x) {

st = i;

break;

}

}

for (int i = arr.length - 1; i >= 0; i--) {

if (arr[i] == x) {

end = i;

break;

}

}

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

res.add(st);

res.add(end);

return res;

}

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter the number of elements in the array: ");

int n = scanner.nextInt();

int[] arr = new int[n];

System.out.println("Enter the elements of the array:");

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

arr[i] = scanner.nextInt();

}

System.out.print("Enter the element to find: ");

int x = scanner.nextInt();

Main solution = new Main();

ArrayList<Integer> result = solution.find(arr, x);

System.out.println("First and last occurrence indices: " + result);

}

}

OUTPUT:

Enter the number of elements in the array: 2

Enter the elements of the array:

1 3

Enter the element to find: 3

First and last occurrence indices: [1, 1]

Time Complexity : O(n)

Space Complexity : O(1)

4. First Transitions

CODE :

import java.util.Scanner;

class Main {

int transitionPoint(int arr[]) {

int st = -1;

// Iterate through the array to find the first occurrence of 1

for (int i = 0; i < arr.length; i++) {

if (arr[i] == 1) {

st = i;

break;

}

}

return st;

}

public static void main(String[] sasta) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter the number of elements in the array: ");

int n = scanner.nextInt();

int[] arr = new int[n];

System.out.println("Enter the elements of the array (only 0's and 1's):");

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

arr[i] = scanner.nextInt();

}

Main solution = new Main();

int result = solution.transitionPoint(arr);

System.out.println("The index of the first occurrence of 1: " + result);

scanner.close();

}

}

Output:

Enter the number of elements in the array: 6

Enter the elements of the array (only 0's and 1's):

0 0 1 1 0 1

The index of the first occurrence of 1: 2

Time Complexity: O(N)

Space Complexity :O(1)

5. First Repeating Element

CODE :

import java.util.\*;

class Main {

public static int firstRepeated(int[] arr) {

Map<Integer, Integer> mp = new HashMap<>();

for (int i = 0; i < arr.length; i++) {

mp.put(arr[i], mp.getOrDefault(arr[i], 0) + 1);

}

for (int i = 0; i < arr.length; i++) {

if (mp.get(arr[i]) >= 2) {

return i + 1;

}

}

return -1;

}

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter the number of elements: ");

int n = scanner.nextInt();

int[] arr = new int[n];

System.out.println("Enter the elements:");

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

arr[i] = scanner.nextInt();

}

int result = firstRepeated(arr);

System.out.println("Position of the first repeating element: " + result);

scanner.close();

}

}

Output:

Enter the number of elements: 3

Enter the elements:

1 2 2

Position of the first repeating element: 2

Time Complexity: O(n)

Space Complexity:O(n)

6. Remove Duplicates Sorted Array

CODE:

import java.util.\*;

class Main {

public int removeDuplicate(int[] arr) {

int i = 0;

int cnt = 0;

while (i < arr.length) {

int j = i;

while (j < arr.length && arr[i] == arr[j]) {

j++;

}

arr[cnt] = arr[i];

cnt++;

i = j;

}

return cnt;

}

public static void main(String[] sasta) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter the number of elements: ");

int n = scanner.nextInt();

int[] arr = new int[n];

System.out.println("Enter the elements:");

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

arr[i] = scanner.nextInt();

}

Main solution = new Main();

int result = solution.removeDuplicate(arr);

System.out.println("The count of unique elements is: " + result);

System.out.println("The array with unique elements is:");

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

System.out.print(arr[i] + " ");

}

}

}

Output:

Enter the number of elements: 4

Enter the elements:

2 2 2 2

The count of unique elements is: 1

The array with unique elements is:

2

Time Complexity : O(n)

Space Complexity : O(1)

7. Maximum Index

CODE:

import java.util.Scanner;

class Solution {

int maxIndexDiff(int[] arr) {

int n = arr.length;

int minLeft[] = new int[n];

int maxRight[] = new int[n];

minLeft[0] = arr[0];

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

minLeft[i] = Math.min(arr[i], minLeft[i - 1]);

}

maxRight[n - 1] = arr[n - 1];

for (int j = n - 2; j >= 0; j--) {

maxRight[j] = Math.max(arr[j], maxRight[j + 1]);

}

int i = 0;

int j = 0;

int maxdiff = -1;

while (i < n && j < n) {

if (minLeft[i] <= maxRight[j]) {

maxdiff = Math.max(maxdiff, j - i);

j++;

} else {

i++;

}

}

return maxdiff;

}

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.println("Enter the number of elements in the array:");

int n = sc.nextInt();

int[] arr = new int[n];

System.out.println("Enter the elements of the array:");

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

arr[i] = sc.nextInt();

}

Solution solution = new Solution();

int result = solution.maxIndexDiff(arr);

System.out.println("The maximum index difference is: " + result);

sc.close();

}

}

Output:

Enter the number of elements in the array:

7

Enter the elements of the array:

1 2 3 4 5 6 7

The maximum index difference is: 6

Time Complexity : O(N)

Space complexity O(N)

8. Wave Array

CODE :

import java.util.Scanner;

class Solution {

public static void convertToWave(int[] arr) {

int n = arr.length;

if (n % 2 == 1) {

n -= 1;

}

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) {

Scanner sc = new Scanner(System.in);

System.out.println("Enter the number of elements in the array:");

int n = sc.nextInt();

int[] arr = new int[n];

System.out.println("Enter the elements of the array:");

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

arr[i] = sc.nextInt();

}

convertToWave(arr);

System.out.println("Array after converting to wave form:");

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

System.out.print(arr[i] + " ");

}

sc.close();

}

}

Output:

Enter the number of elements in the array:

6

Enter the elements of the array:

1 2 3 4 5 6

Array after converting to wave form:

2 1 4 3 6 5

Time Complexity: O(n)

Space Complexity : O(1)