TECHNICAL TRAINING DSA - CODING PRACTICE PROBLEMS

Name: Rishi Kumar S
Dept: CSBS
Date: 14-11-2024

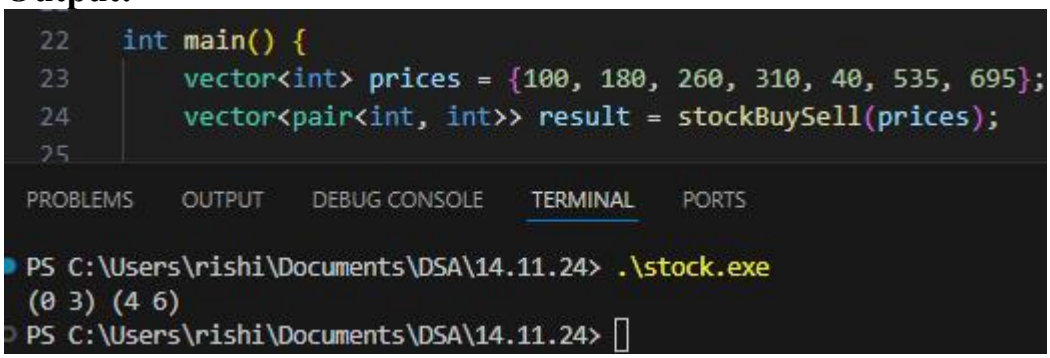
Question 1:
Stock Buy and sell

Code:

```
#include <iostream>
#include <vector>
using namespace std;

vector<pair<int, int>> stockBuySell(vector<int> &prices) {
    vector<pair<int, int>> result;
    int n = prices.size();
    for (int i = 0; i < n - 1;) {
        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.push_back({buy, sell});
    }
    return result;
}

int main() {
    vector<int> prices = {100, 180, 260, 310, 40, 535, 695};
    vector<pair<int, int>> result = stockBuySell(prices);

    if (result.empty()) {
        cout << "No Profit" << endl;
    } else {
        for (auto &p : result) {
            cout << "(" << p.first << " " << p.second << ") ";
        }
        cout << endl;
```

```
    }

    return 0;
}
```

Output:



Time Complexity: O(n^2)
Space Complexity: O(1)

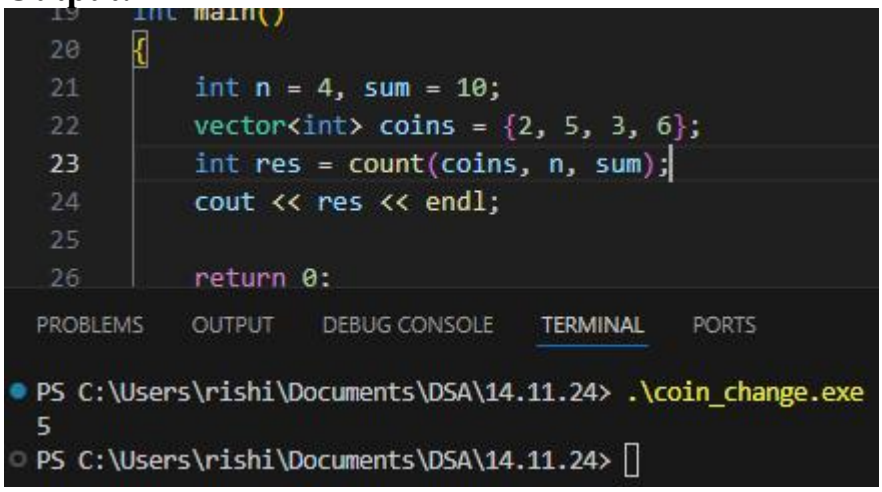
Question 2:
Coin change (Count ways)

Code:

```
#include <vector>
#include <iostream>
using namespace std;

int helper(vector<int> &coins, int n, int sum, vector<vector<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[n-1], memo) +
helper(coins, n - 1, sum, memo);
}
int count(vector<int> &coins, int n, int sum) {
    vector<vector<int>> memo(n + 1, vector<int>(sum + 1, -1));
    return helper(coins, n, sum, memo);
}
```

```
int main()
{
    int n = 4, sum = 10;
    vector<int> coins = {2, 5, 3, 6};
    int res = count(coins, n, sum);
    cout << res << endl;

    return 0;
}
```

Output:



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

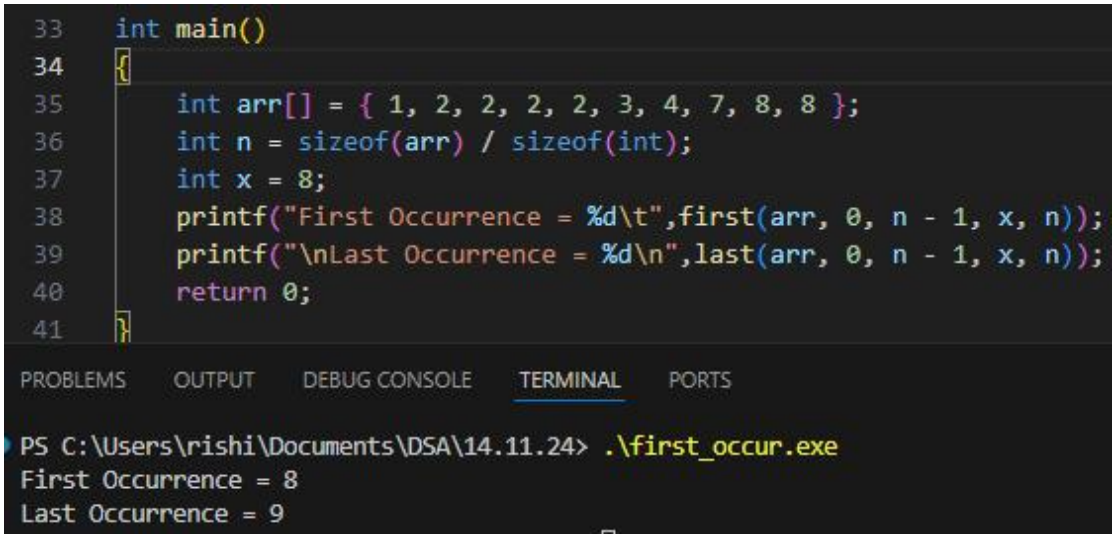
Question 3:
First and Last Occurences

Code:

```
#include <vector>
#include <iostream>
using namespace std;

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;
}
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;
}

int main()
{
    int arr[] = { 1, 2, 2, 2, 2, 3, 4, 7, 8, 8 };
    int n = sizeof(arr) / sizeof(int);
    int x = 8;
    printf("First Occurrence = %d\t",first(arr, 0, n - 1, x, n));
    printf("\nLast Occurrence = %d\n",last(arr, 0, n - 1, x, n));
    return 0;
}
```

Output:



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

Question 4:
Find Transition Point

CODE:

```
#include <vector>
#include <iostream>
using namespace std;

int transitionPoint(vector<int>& arr) {
        if(arr[arr.size()-1]==0) return -1;
        if(arr[0]==1) return 0;
        int left=0,right=arr.size()-1;
        int mid=0;
        while(left<=right){
            mid=(left+right)/2;
            if(mid>0 && arr[mid]==1 && arr[mid-1]==0) return mid;
            else if(arr[mid]==1) right=mid-1;
            else left=mid+1;
        }
        return -1;
}

int main(){
    vector<int> arr= {0, 0, 0, 1, 1};
    int n = sizeof(arr) / sizeof(int);
    cout<<"Transtiton point is: "<<transitionPoint(arr);
    return 0;
```

}

Output:

```
19  int main(){
20      vector<int> arr= {0, 0, 0, 1, 1};
21      int n = sizeof(arr) / sizeof(int);
22      cout<<"Transtiton point is: "<<transitionPoint(arr);
23      return 0;
24  }
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\rishi\Documents\DSA\14.11.24> .\trans_pt.exe
Transtiton point is: 3
PS C:\Users\rishi\Documents\DSA\14.11.24>
```

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

Question 5:
First Repeating element

CODE:

```
#include <vector>
#include <iostream>
#include <unordered_map>
using namespace std;

int firstRepeated(vector<int> &arr) {
        int minn=INT_MAX;
        unordered_map<int,int> map;
        int n=arr.size();
        for(int i=0;i<n;i++){
            if(map.find(arr[i])!=map.end()){
                minn=min(minn,map[arr[i]]);
            }
            else map[arr[i]]=i;
        }
        if(minn==INT_MAX) return -1;
        return minn+1;
}

int main(){
    vector<int> arr= {0, 0, 0, 1, 1};
    int n = sizeof(arr) / sizeof(int);
```

```
    cout<<"First Repeated index: "<<firstRepeated(arr);
    return 0;
}
```

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

Output:

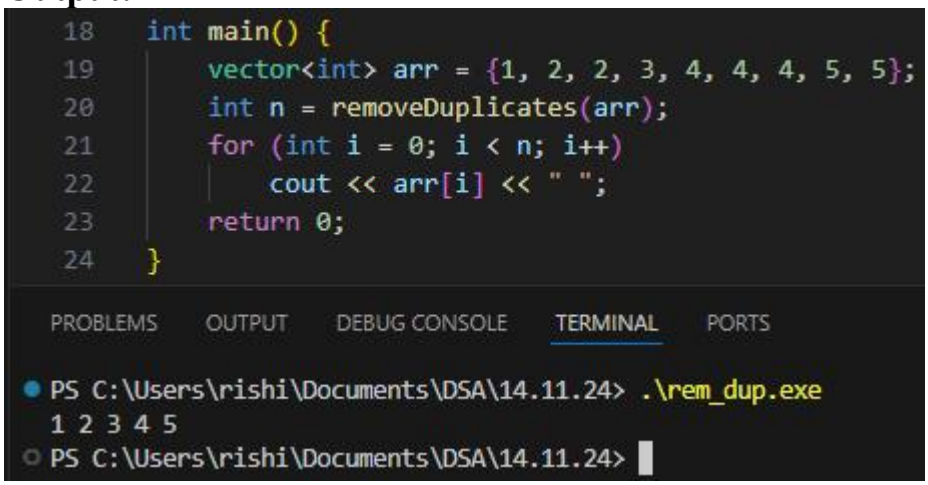
Question 6:
Remove Duplicates - Sorted Array

Code:

```
#include <vector>
#include <iostream>
using namespace std;

int removeDuplicates(vector<int>& arr) {
    int n = arr.size();
    if (n <= 1)
        return n;
    int ind = 1;
    for (int i = 1; i < n; i++) {
        if (arr[i] != arr[i - 1]) {
            arr[ind++] = arr[i];
        }
    }
    return ind;
}

int main() {
```

```
    vector<int> arr = {1, 2, 2, 3, 4, 4, 4, 5, 5};
    int n = removeDuplicates(arr);
    for (int i = 0; i < n; i++)
        cout << arr[i] << " ";
    return 0;
}
```

Output:



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

Question 7:
Maximum Index

CODE:

```
#include <vector>
#include <unordered_map>
#include <algorithm>
#include <iostream>
using namespace std;

int maxIndexDiff(vector<int>& arr, int n){
    unordered_map<int, vector<int> > map;
    for (int i = 0; i < n; i++) {
        map[arr[i]].push_back(i);
    }
    sort(arr.begin(), arr.end());
    int maxDiff = INT_MIN;
```

```
    int temp = n;
    for (int i = 0; i < n; i++) {
        if (temp > map[arr[i]][0]){
            temp = map[arr[i]][0];
        }
        maxDiff = max(maxDiff,map[arr[i]][map[arr[i]].size() - 1]-temp);
    }
    return maxDiff;
}
int main()
{
    int n = 9;
    vector<int> arr{ 34, 8, 10, 3, 2, 80, 30, 33, 1 };
    int ans = maxIndexDiff(arr, n);
    cout << "The maxIndexDiff is : " << ans << endl;
    return 1;
}
```

Output:

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