

Coin Change (Counting ways)

```
class Solution {
public:
    int count(vector<int>& coins, int sum) {
        int n = coins.size();
        vector<int> dp (sum+1,0);
        for (int coin: coins){
            for (int j=1; j<=sum; j++){
                if (j-coin<0) continue;
                else if (j-coin==0) dp[j]+=1;
                else {
                    dp[j] += dp[j-coin];
                }
            }
        }
        return dp[sum];
    }
};
```

2D DP: $dp[i][j] = dp[i-1][j] + dp[i][j - \text{coin}]$

1D DP: Replacing on same for every coin $dp[i] += dp[i - \text{coin}]$

The screenshot shows a coding platform interface with a dark theme. On the left, the 'Output Window' is open, displaying 'Compilation Results' and 'Problem Solved Successfully' with a green checkmark. Below this, statistics are shown: 'Test Cases Passed' as 1111/1111, 'Attempts: Correct / Total' as 1/1, 'Accuracy: 100%', 'Points Scored' as 4/4, and 'Time Taken' as 0.07. At the bottom, it says 'Your Total Score: 14'. On the right, the code editor shows the same C++ solution as in the first block, with line numbers 1 through 24. The code is enclosed in a driver code template.

Time Complexity: $O(n * \text{sum})$

Space Complexity: $O(\text{sum})$

First and Last Occurrences

```
class Solution {
public:
    int binsearch_l(vector<int> &arr, int val){
        int n = arr.size(); int l=0, r=n-1, m=0, res=-1;
        while (l<=r){
            m = (l+r)/2;
```

```

        if (arr[m]>=val) {
            if (arr[m]==val) res=m;
            r = m-1;
        }
        else l=m+1;
    }
    return res;
}

int binsearch_r(vector<int> &arr, int val){
    int n = arr.size(); int l=0, r=n-1, m=0, res=-1;
    while (l<=r){
        m = (l+r)/2;
        if (arr[m]>val) r = m-1;
        else {
            if (arr[m]==val) res=m;
            l=m+1;
        }
    }
    return res;
}

vector<int> find(vector<int>& arr, int x) {
    int first = binsearch_l(arr,x);
    int last = binsearch_r(arr,x);
    return {first,last};
}

};

```

Output Window

Compilation Results Custom Input Y.O.G.I. (AI Bot)

Problem Solved Successfully

Suggest Feedback

Test Cases Passed
1120 / 1120

Attempts : Correct / Total
2 / 8

Accuracy : 25%

Time Taken
0.56

You get marks only for the first correct submission if you solve the problem without viewing the full solution.

```

1 // Driver Code Ends
2 class Solution {
3 public:
4     int binsearch_l(vector<int> &arr, int val){
5         int n = arr.size(); int l=0, r=n-1, m=0, res=-1;
6         while (l<=r){
7             m = (l+r)/2;
8             if (arr[m]>=val) {
9                 if (arr[m]==val) res=m;
10                r = m-1;
11            }
12            else l=m+1;
13        }
14        return res;
15    }
16
17    int binsearch_r(vector<int> &arr, int val){
18        int n = arr.size(); int l=0, r=n-1, m=0, res=-1;
19        while (l<=r){
20            m = (l+r)/2;
21            if (arr[m]>val) r = m-1;
22            else {
23                if (arr[m]==val) res=m;
24                l=m+1;
25            }
26        }
27        return res;
28    }
29
30    vector<int> find(vector<int>& arr, int x) {
31        int first = binsearch_l(arr,x);
32        int last = binsearch_r(arr,x);
33        return {first,last};
34    }
35 }

```

Time Complexity: $O(\log(N))$

Space Complexity: $O(1)$

Find Transition Point

```

class Solution {
public:
    int transitionPoint(vector<int>& arr) {
        int n = arr.size();

```

```

        int l=0, r=n-1, m=0, res=-1;
        while (l<=r){
            m = (l+r)/2;
            if (arr[m]==1) {r=m-1; res=m;}
            else l=m+1;
        }
        return res;
    }
};

```

The screenshot shows a coding platform interface with the following details:

- Problem:** Corporate Hiring Solutions
- Compilation Results:** Problem Solved Successfully (green checkmark)
- Test Cases Passed:** 1115 / 1115
- Attempts:** Correct / Total: 1 / 1
- Accuracy:** 100%
- Points Scored:** 2 / 2
- Time Taken:** 0.11
- Your Total Score:** 20 (with an upward arrow)
- Code Editor:** Shows the C++ code for the 'transitionPoint' function, which is identical to the code in the first block.

Time Complexity: $O(\log(N))$

Space Complexity: $O(1)$

First Repeating Element

```

class Solution {
public:
    // Function to return the position of the first repeating element.
    int firstRepeated(vector<int> &arr) {
        unordered_map<int,int> mp;
        int i=1, mv=INT_MAX, rep_index=1e6+5;
        for (int val: arr){
            if (mp[val]) {
                rep_index=min(rep_index,mp[val]);
            }
            mp[val] = i++;
        }
        return (rep_index==1e6+5?-1:rep_index);
    }
};

```

Output Window

Compilation Results
Custom Input
Y.O.G.I. (AI Bot)

Problem Solved Successfully
Suggest Feedback

Test Cases Passed
1115 / 1115

Attempts : Correct / Total
1 / 1
Accuracy : 100%

Points Scored
2 / 2
Your Total Score: 22

Time Taken
1.2

```

1 // Driver Code Ends
9 // User function template in C++
10
11 class Solution {
12 public:
13     // function to return the position of the first repeating
14     int firstRepeated(vector<int> &arr) {
15         unordered_map<int,int> mp;
16         int i=1, mv=INT_MAX, rep_index=1e6+5;
17         for (int val: arr){
18             if (mp[val]) {
19                 rep_index=min(rep_index,mp[val]);
20             }
21             mp[val] = i++;
22         }
23         return (rep_index==1e6+5?-1:rep_index);
24     }
25 };
26 // Driver Code Ends

```

Time Complexity: $O(N)$

Space Complexity: $O(N)$

Remove Duplicates sorted array

```

class Solution {
public:
    int remove_duplicate(vector<int> &arr) {
        auto i = arr.begin();
        int n = arr.size(), el = -1, rep=0;
        while (i!=arr.end()){
            if (el==arr[i-arr.begin()]) {i=arr.erase(i);rep++;}
            else {el=arr[i-arr.begin()];i++;}
        }
        return n-rep;
    }
};

```

Time Complexity: $O(N)$

Space Complexity: $O(1)$

--- Java

Maximum Index

```

int maxIndexDiff(int[] arr) {

    int[] Lmin = new int[arr.length];
    int[] Lmax = new int[arr.length];

    Lmin[0] = arr[0];
    Lmax[0] = arr[0];
    for(int i = 1; i < arr.length; i++){
        Lmin[i] = Math.min(arr[i], Lmin[i-1]);
    }
}

```

```

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

    int i = 0, j = 0, maxLength = -1;
    while(i < arr.length && j < arr.length){
        if(Lmin[i] <= Lmax[j]){
            maxLength = Math.max(maxLength, j-i);
            j++;
        }
        else{
            i++;
        }
    }

    return maxLength;
}

```

Wave Array

```

public static void convertToWave(int[] arr) {
    int n = arr.length;
    int i = 0, j = 1;
    while(n > 1){
        int temp = arr[i];
        arr[i] = arr[j];
        arr[j] = temp;
        n-=2;
        i+=2;
        j+=2;
    }
}

```