



## **Experiment 1.3**

**Aim:** *Evaluate the complexity of the developed program to find frequency of elements in a given array. (Using hash maps)*

**Objectives:** *To understand the concept of Hash maps and Arrays.*

**Input/Apparatus Used:** VS CODE

**Procedure/Algorithm:**

1. Define a function named *findFrequency* that takes an array (*arr[]*) and its size (*n*) as parameters.
  - Create an *unordered\_map* named *mp* to store element frequencies.
2. Iterate through the array using a loop with index *i* ranging from 0 to *n-1*:
  - Increment the frequency of *arr[i]* in the *mp* map.
3. Iterate through the elements in the *mp* map:
  - Print the element and its corresponding frequency.
4. End of the *findFrequency* function.
5. In the main function:
  - Declare an integer array named *arr* and initialize it with elements.
  - Calculate the size of the array (*n*) using *sizeof(arr) / sizeof(arr[0])*.
  - Call the *findFrequency* function with *arr[]* and *n* as arguments.
6. End of the program.



Course Name: DAA Lab

Course Code: 21ITH-311/21CSH-311

**Code:**

```
#include <bits/stdc++.h>
using namespace std;
void findFrequency(int arr[], int n)
{
    unordered_map<int, int> mp;
    for (int i = 0; i < n; i++) {
        mp[arr[i]]++;
    }
    for (auto i : mp) {
        cout << i.first << " comes " << i.second << " times" << endl;
    }
}
void findFrequencyOrdered(int arr[], int n)
{
    map<int, int> mp;
    for (int i = 0; i < n; i++) {
        mp[arr[i]]++;
    }
    for (auto i : mp) {
        cout << i.first << " comes " << i.second << " times" << endl;
    }
}

int main(){
    int arr[] = { 1, 2, 5, 2, 3, 6, 5, 5, 8, 9, 11, 9, 9, 10 };
    int n = sizeof(arr) / sizeof(arr[0]);
    cout<<"Using Unordered Map:"<<endl;
    findFrequency(arr, n);

    cout<<"Using Ordered Map:"<<endl;
```



Course Name: DAA Lab

Course Code: 21ITH-311/21CSH-311

```
    findFrequencyOrdered(arr, n);  
    return 0;  
}
```

**Observations/Outcome :**

```
ashmap }  
Using Unordered Map:  
10 comes 1 times  
11 comes 1 times  
9 comes 3 times  
8 comes 1 times  
6 comes 1 times  
1 comes 1 times  
2 comes 2 times  
5 comes 3 times  
3 comes 1 times  
Using Ordered Map:  
1 comes 1 times  
2 comes 2 times  
3 comes 1 times  
5 comes 3 times  
6 comes 1 times  
8 comes 1 times  
9 comes 3 times  
10 comes 1 times  
11 comes 1 times  
PS C:\Users\SANJIV\Downloads\CSE-5TH-SEM-WORKS  
HEETS-DAA-AIML-IOT-AP\DAA\Experiment 3> |
```

**Time Complexity:  $O(1)$**