WORKSHEET – 3

Name: Ruchika Raj

Section/Group: 615 / B

UID: 20BCS9285

Subject: Design and Analysis of Algorithm Lab

Date of Submission: 02.09.2022

Branch: BE CSE

AIM :

Given an array which may contain duplicates, print all elements and their frequencies.

Task to be done/ Which logistics used:

Task: To find the frequencies of every element of an array.

Logic Used: This can be done through two loops. The first loop will select an element and the second loop will iteration through the array by comparing the selected element with other elements. If a match is found, print the duplicate element.

Platform Used : Online Compiler

Algorithm/Flowchart :

1. Start the program.

2. To check the status of visited elements create an array of size n.

3. Run a loop from index 0 to n and check if (visited[i]==1) then skip that element.

4. Otherwise create a variable count = 1 to keep the count of frequency.

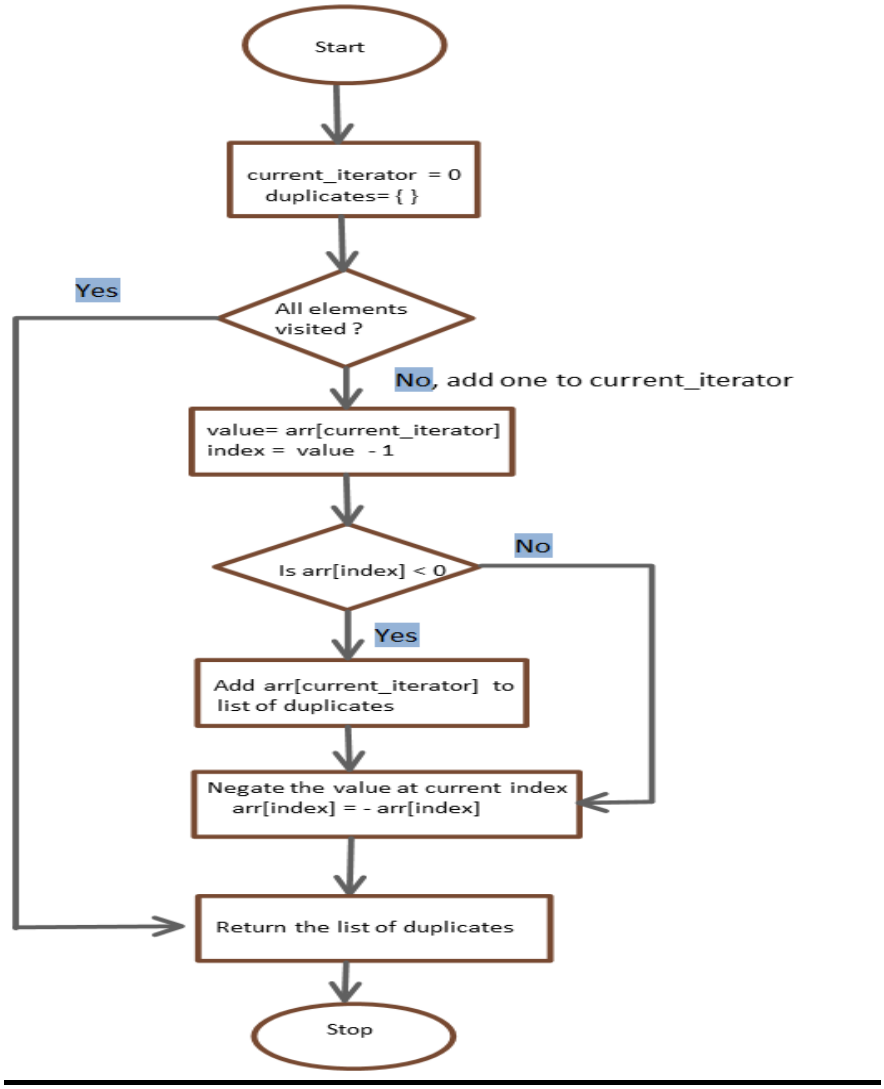
5. Run a loop from index i+1 to n.

6. Check if(arr[i]==arr[j]), then increment the count by 1 and set visited[j]=1.

7. After complete iteration of for loop print element along with value of count.

8. End the program.

Flowchart :



Pseudo Code Of The Algorithm:

1. Begin
2. Read array arr[n].
3. Start a loop for checking a visited element first.
4. If (arr[i]==arr[j]) make a count++.
5. Print count.
6. End

CODE:

```
#include <bits/stdc++.h>
using namespace std;

int main()
{
   int n;
   cin>>n;
   int arr[n];

   for(int i=0; i<n;i++){
         cin>>arr[i];
   }
   int visited[n];
   for(int i=0; i<n; i++){
      if(visited[i]!=1){
        int count = 1;
        for(int j=i+1; j<n; j++){
          if(arr[i]==arr[j]){
             count++;
             visited[j]=1;
          }
        }
        cout<<arr[i]<<" occurs at "<<count<<" times "<<endl;
      }
    }

   return 0;
}
```

Observations/Discussions/

Complexity Analysis:

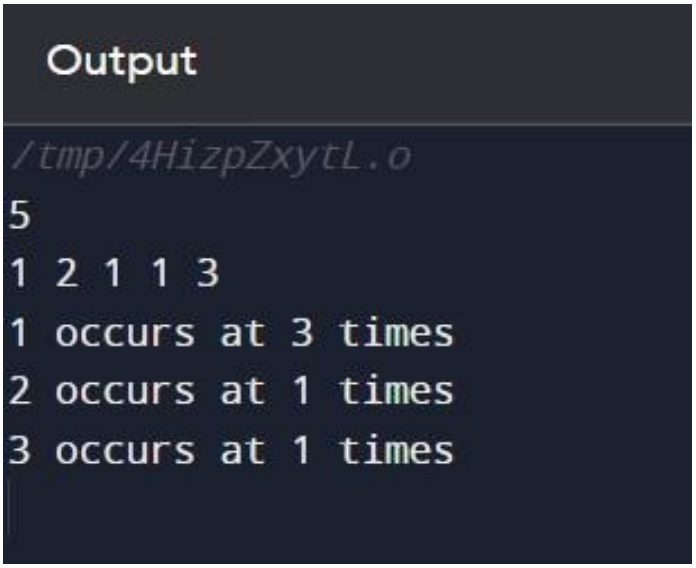
Brute force approach: This is the most basic and easiest approach to find and print duplicate elements of an array. In this approach, we use two for loops (inner and outer loops) to compare an element with each element of an array.

For every ith element run a loop on the given array from (i+1) to n and check if the ith element is present in it or not.

Time Complexity: O(n^2)

Space Complexity: O(n)

Result/Output/Writing Summary:



Learning outcomes (What I have learnt):

1. Learnt about Nested loops.
2. Learnt about working with two Array simultaneously.

3. Learnt how to use Brute force approach.