

## DAILY ONLINE ACTIVITIES SUMMARY

Date:	02/06/2020		Name:	M MAHAMMAD ASIF
Sem & Sec	4 <sup>th</sup> Sem & 'A' Sec		USN:	4AL18CS045
<b>Online Test Summary</b>				
Subject	-			
Max. Marks	-		Score	-
<b>Certification Course Summary</b>				
Course	The Complete Android App Development Masterclass:Build Apps.			
Certificate Provider	Udemy		Duration	29 Hours
<b>Coding Challenges</b>				
<b>Problem Statement:</b> 1. C Program to find inversion count of array. 2. Java program to find Perfect Sum Problem.				
<b>Status: Completed</b>				
Uploaded the report in Github			Yes	
If yes Repository name			<a href="https://github.com/alvas-education-foundation/M_MAHAMMAD_ASIF">https://github.com/alvas-education-foundation/M_MAHAMMAD_ASIF</a>	
Uploaded the report in slack			Yes	

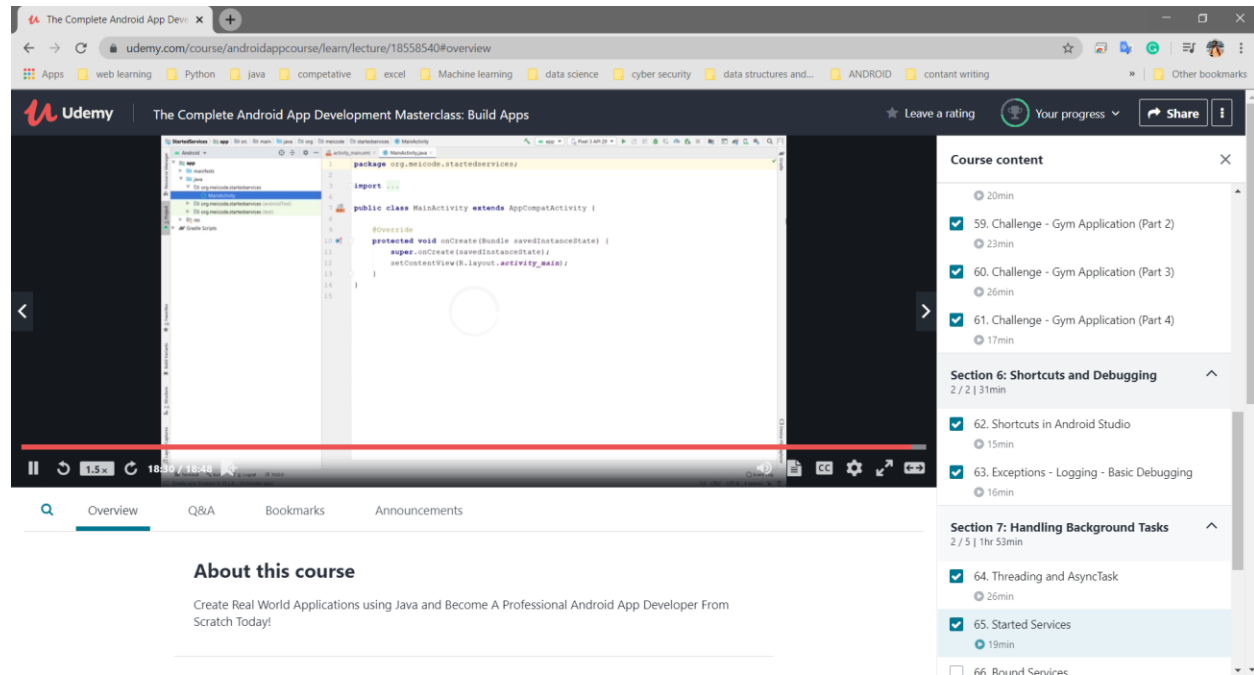
**Online Test Details: Today test was not conducted.**

## **Certification Course Details:**

**I have continued the the course that is “Complete Android App Development Masterclass: Build Apps”, which is about 29 hours of Duration. In that, I had completed Next part of yesterday’s topic, which was about more than an hour. Parallel to that whatever learns in course I’m practicing in Android Studio.**

**In additional to this daily I’m doing some other online courses aswell, as a proof I uploaded some Certificates in my other repository named “Completed course certificates.”**

## **Snapshot:**



The screenshot displays the Udemy interface for the course "The Complete Android App Development Masterclass: Build Apps". The main video player shows a code editor with the following Java code:

```
package org.meicode.startedservices;

import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
}
```

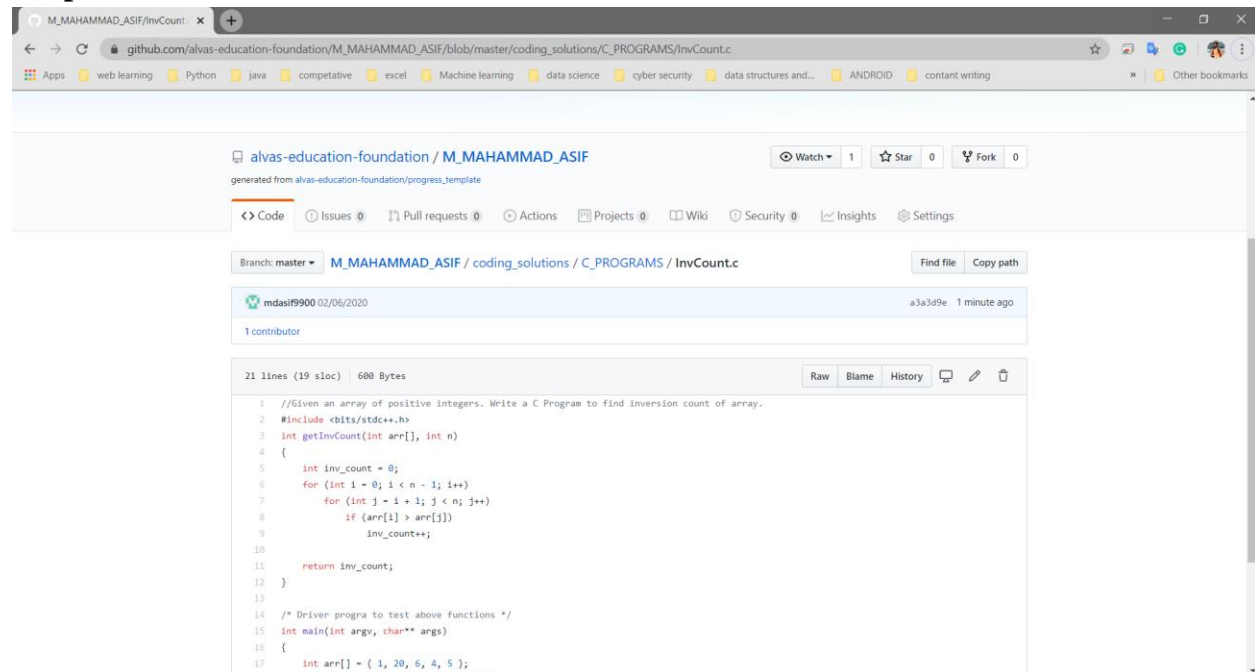
The course content sidebar on the right lists the following items:

- 59. Challenge - Gym Application (Part 2) - 23min
- 60. Challenge - Gym Application (Part 3) - 26min
- 61. Challenge - Gym Application (Part 4) - 17min
- Section 6: Shortcuts and Debugging - 2 / 2 | 31min
  - 62. Shortcuts in Android Studio - 15min
  - 63. Exceptions - Logging - Basic Debugging - 16min
- Section 7: Handling Background Tasks - 2 / 5 | 1hr 53min
  - 64. Threading and AsyncTask - 26min
  - 65. Started Services - 19min
  - 66. Bound Services

**Coding Challenges Details: The Two problems I have solved By Understanding the Concepts through Online and updated the same in Github Repository. The two problem statements were:**

## 1. C Program to find inversion count of array.

### Snapshot:

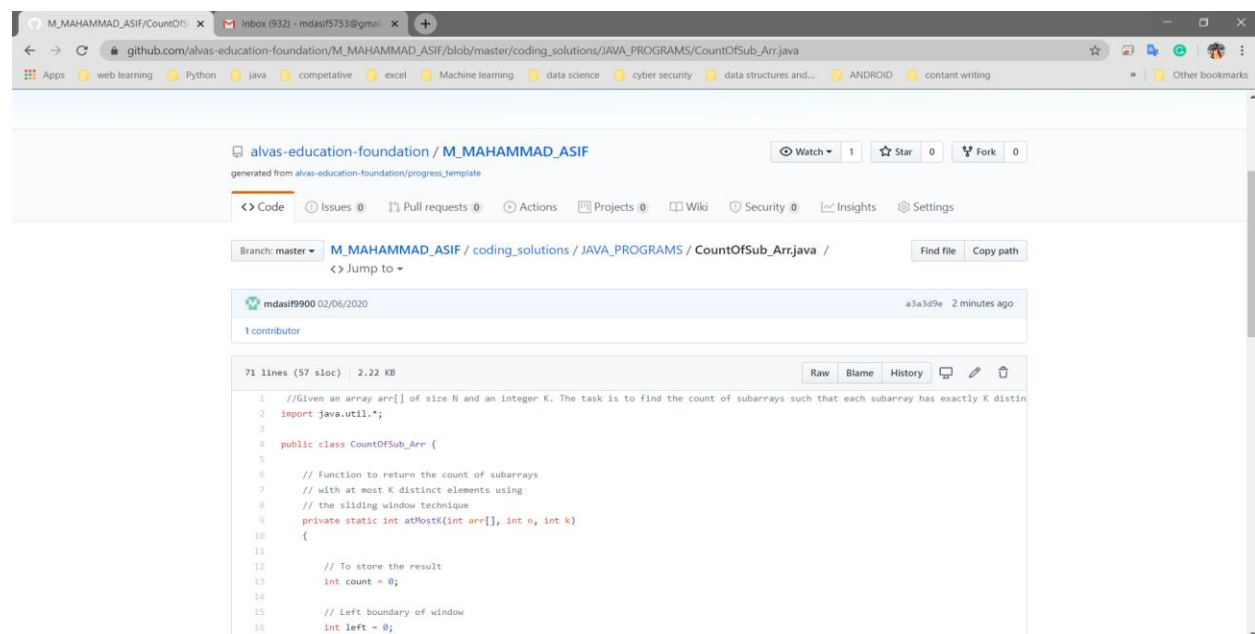


The screenshot shows a GitHub repository for 'alvas-education-foundation / M\_MAHAMMAD\_ASIF'. The file 'InvCount.c' is selected, showing its code. The code is a C program to find the inversion count of an array. It includes a function 'getInvCount' that takes an array and its size, and returns the inversion count. The main function tests the 'getInvCount' function with an array [1, 20, 6, 4, 5].

```
1 //Given an array of positive integers. Write a C Program to find inversion count of array.
2 #include <bits/stdc++.h>
3 int getInvCount(int arr[], int n)
4 {
5     int inv_count = 0;
6     for (int i = 0; i < n - 1; i++)
7         for (int j = i + 1; j < n; j++)
8             if (arr[i] > arr[j])
9                 inv_count++;
10
11     return inv_count;
12 }
13
14 /* Driver program to test above functions */
15 int main(int argc, char** args)
16 {
17     int arr[] = { 1, 20, 6, 4, 5 };
18     ...
```

## 2 Java program to find Perfect Sum Problem

### Snapshot:



The screenshot shows a GitHub repository for 'alvas-education-foundation / M\_MAHAMMAD\_ASIF'. The file 'CountOfSub\_Arr.java' is selected, showing its code. The code is a Java program to find the count of subarrays such that each subarray has exactly K distinct elements. It includes a function 'atMostK' that takes an array, K, and returns the count of subarrays with at most K distinct elements. The main function tests the 'atMostK' function with an array [1, 2, 3, 4, 5] and K=2.

```
1 //Given an array arr[] of size N and an Integer K. The task is to find the count of subarrays such that each subarray has exactly K distinct
2 import java.util.*;
3
4 public class CountOfSub_Arr {
5
6     // Function to return the count of subarrays
7     // with at most K distinct elements using
8     // the sliding window technique
9     private static int atMostK(int arr[], int n, int k)
10     {
11
12         // To store the result
13         int count = 0;
14
15         // Left boundary of window
16         int left = 0;
17     }
```

