

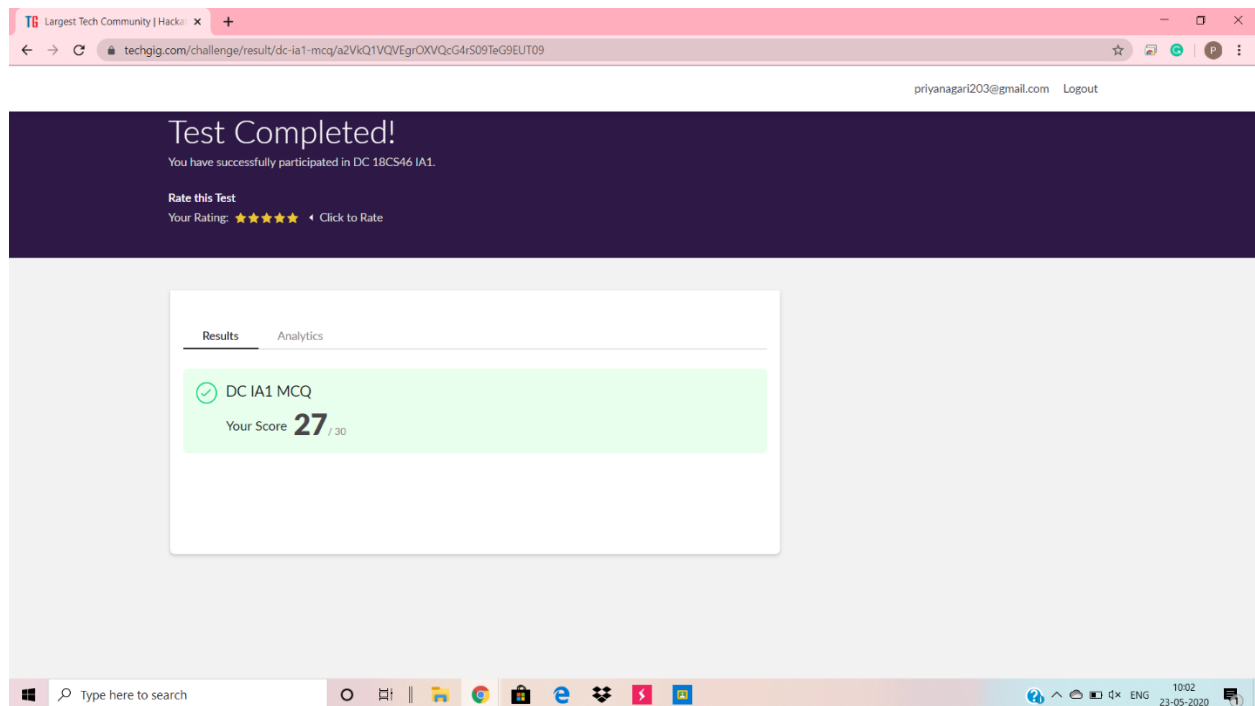
DAILY ONLINE ACTIVITIES SUMMARY

Date:	23/05/2020	Name:	Priya Nagari
Sem & Sec	Fourth SEM section B	USN:	4AL18CS063
Online Test Summary			
Subject	Data Communication		
Max. Marks	30	Score	27
Certification Course Summary			
Course	The complete Android app development Masterclass:Build apps		
Certificate Provider	Udemy	Duration	29 hours
Coding Challenges			
Problem Statement: 1. Write a C or Java program to implement FCFS and SJF process scheduling.			
Status:			
Uploaded the report in Github		YES	
If yes Repository name		Priya_Nagari link: https://github.com/alvas-education-foundation/Priya_Nagari	
Uploaded the report in slack		YES	

Online Test Details:

The online test was about **Introduction to Data Communication, Network Models**(module:1) and **Physical Layer-1, Digital Transmission and analog transmission** (module:2). There were 30 questions and the duration :40 minutes. The questions are multiple choice question type. The score for the test was 27.

Snapshot:



Certification Course Details:

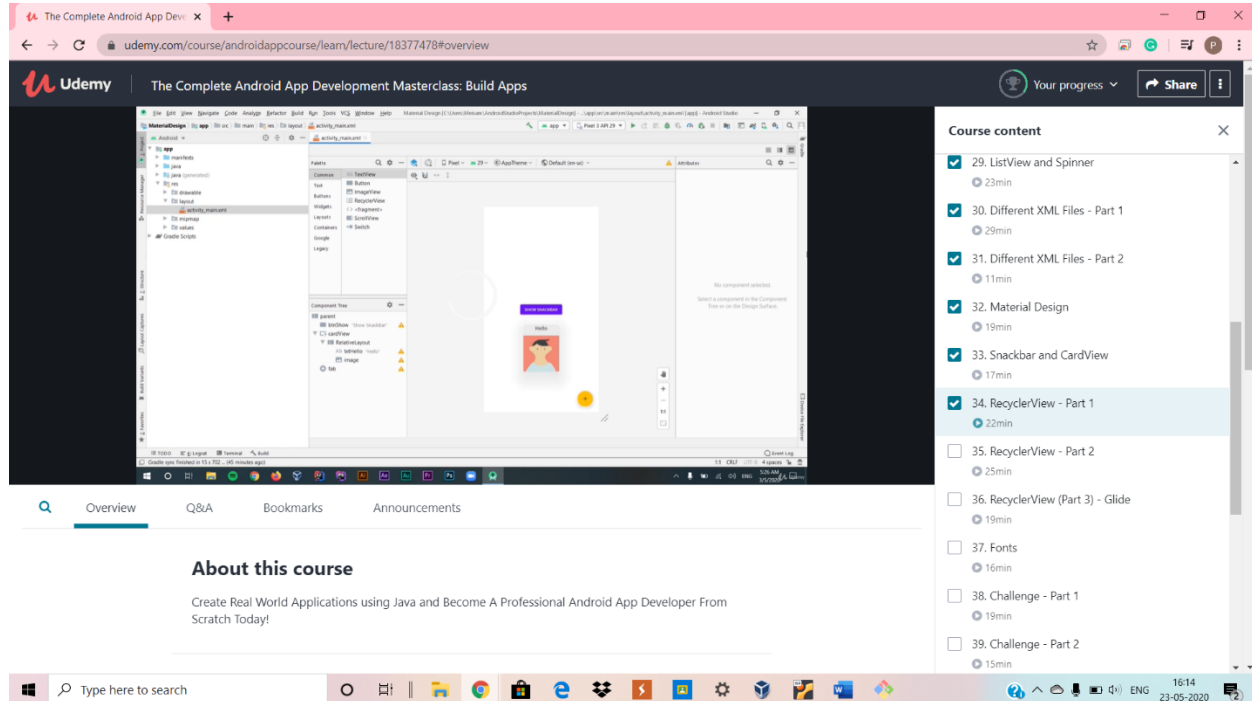
Name of the course: The complete Android app development Masterclass: Build apps

Certificate Provider: Udemy

total duration is 29 hours.

Today I learnt about different types of XML files, material design concepts and initial part of RecyclerView.

Snapshot:

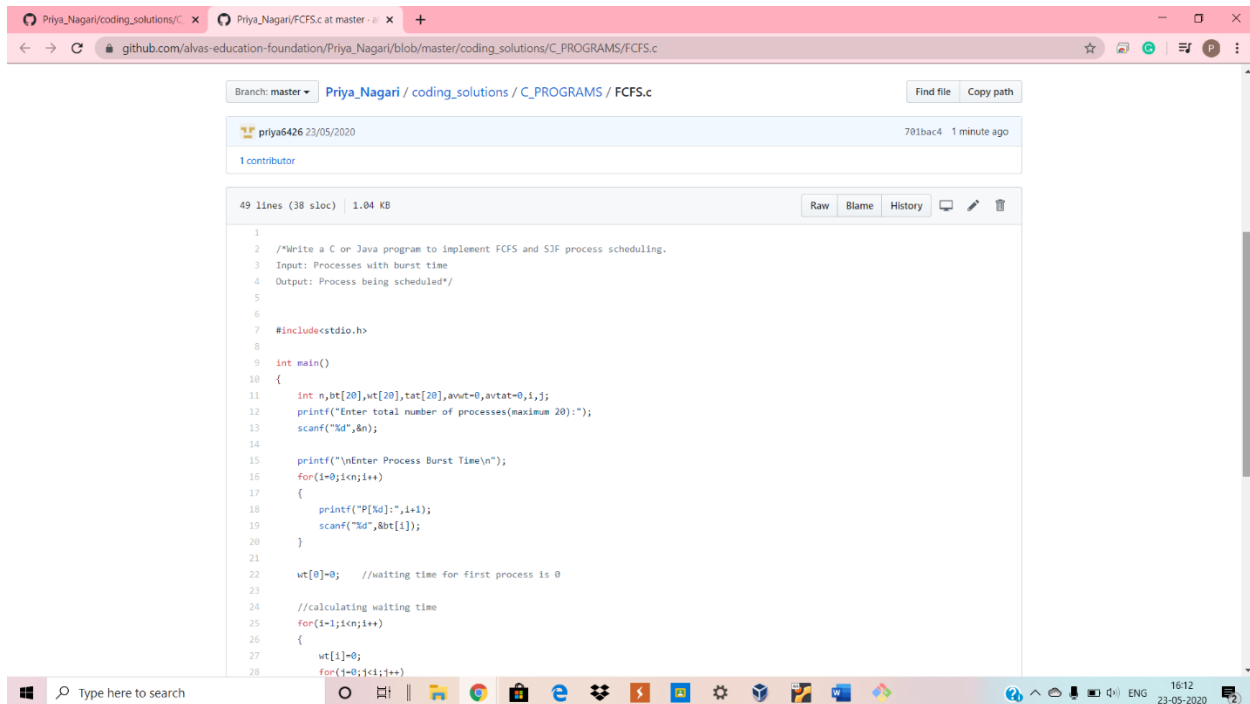


Online Coding Details:

1. Write a C or Java program to implement FCFS and SJF process scheduling.

Solution: uploaded in git hub.

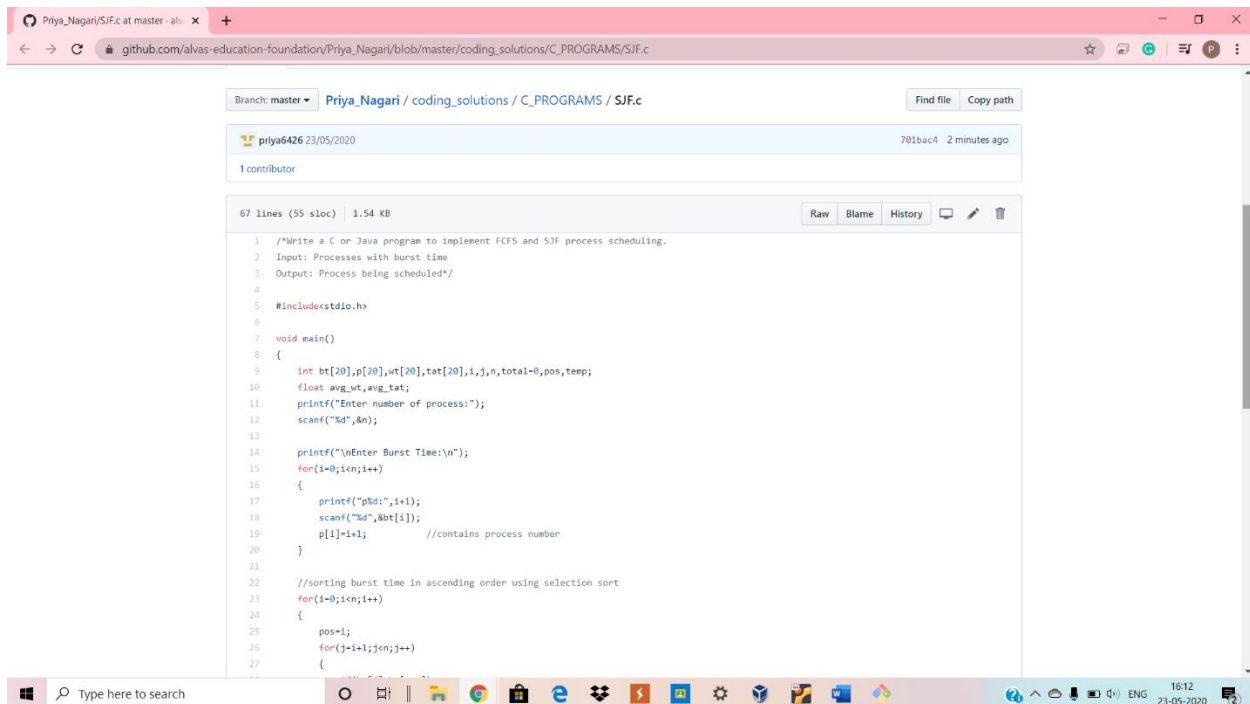
1a.C program to implement FCFS.



The screenshot shows a GitHub web interface for a repository named 'Priya_Nagari / coding_solutions / C_PROGRAMS / FCFS.c'. The file is 1.04 KB and contains 49 lines of C code. The code implements a First-Come-First-Served (FCFS) scheduling algorithm. It starts with a comment: '/*Write a C or Java program to implement FCFS and SJF process scheduling. Input: Processes with burst time Output: Process being scheduled*/'. It includes the standard input/output library and defines a main function. In the main function, it prompts the user to enter the total number of processes (maximum 20) and then the burst time for each process. It then calculates the waiting time for each process, starting from 0 for the first process, and prints the process ID and its burst time. The code is as follows:

```
1
2 /*Write a C or Java program to implement FCFS and SJF process scheduling.
3 Input: Processes with burst time
4 Output: Process being scheduled*/
5
6
7 #include<stdio.h>
8
9 int main()
10 {
11     int n,bt[20],wt[20],tat[20],avwt=0,avtat=0,i,j;
12     printf("Enter total number of processes(maximum 20):");
13     scanf("%d",&n);
14
15     printf("\nEnter Process Burst Time\n");
16     for(i=0;i<n;i++)
17     {
18         printf("P[%d]:",i+1);
19         scanf("%d",&bt[i]);
20     }
21
22     wt[0]=0; //waiting time for first process is 0
23
24     //calculating waiting time
25     for(i=1;i<n;i++)
26     {
27         wt[i]=0;
28         for(j=0;j<i;j++)
```

1b.C program to implement SJF.



The screenshot shows a GitHub web interface for a repository named 'Priya_Nagari / coding_solutions / C_PROGRAMS / SJF.c'. The file is 1.54 KB and contains 67 lines of C code. The code implements a Shortest-Job-First (SJF) scheduling algorithm. It starts with a comment: '/*Write a C or Java program to implement FCFS and SJF process scheduling. Input: Processes with burst time Output: Process being scheduled*/'. It includes the standard input/output library and defines a main function. In the main function, it prompts the user to enter the number of processes and then the burst time for each process. It then sorts the burst times in ascending order using selection sort and prints the process ID and its burst time. The code is as follows:

```
1
2 /*Write a C or Java program to implement FCFS and SJF process scheduling.
3 Input: Processes with burst time
4 Output: Process being scheduled*/
5
6
7 #include<stdio.h>
8
9 void main()
10 {
11     int bt[20],p[20],wt[20],tat[20],i,j,n,total=0,pos,temp;
12     float avg_wt,avg_tat;
13     printf("Enter number of process:");
14     scanf("%d",&n);
15
16     printf("\nEnter Burst Time:\n");
17     for(i=0;i<n;i++)
18     {
19         printf("p[%d]:",i+1);
20         scanf("%d",&bt[i]);
21         p[i]=i+1; //contains process number
22     }
23
24     //sorting burst time in ascending order using selection sort
25     for(i=0;i<n;i++)
26     {
27         pos=i;
28         for(j=i+1;j<n;j++)
29         {
```