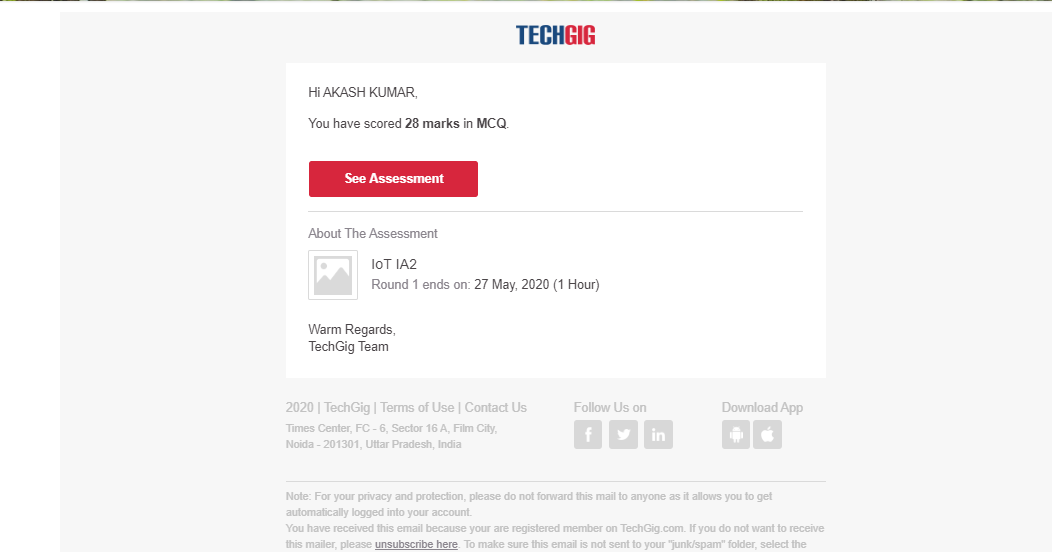
**DAILY ONLINE ACTIVITIES SUMMARY**

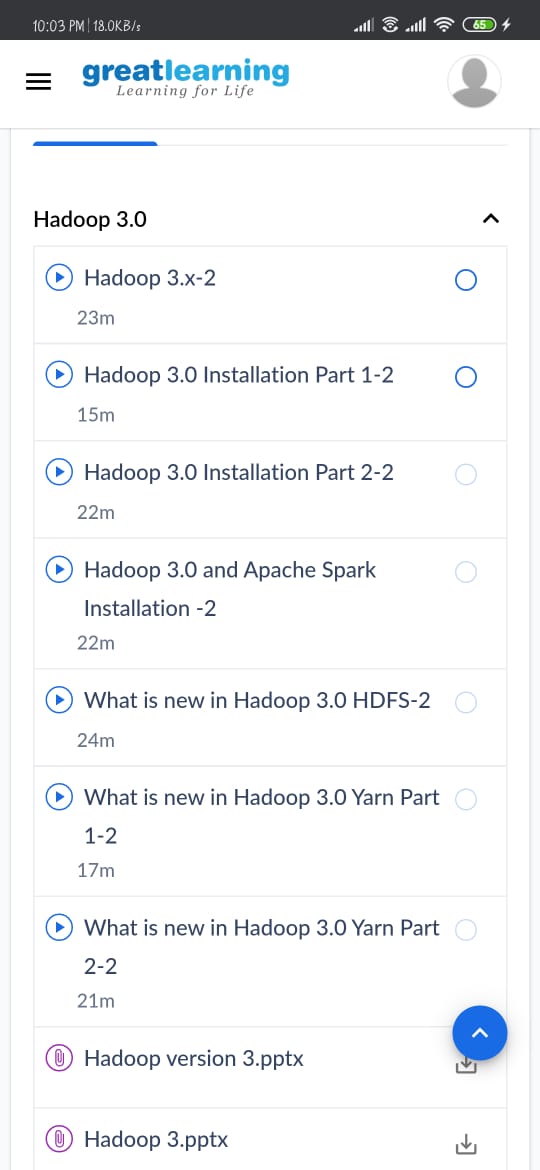
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Date:** | **27/05/2020** | | | | **Name:** | **AKASH KUMAR S** | |
| **Sem & Sec** | **8thA** | | | | **USN:** | **4AL16CS006** | |
| **Online Test Summary** | | | | | | | |
| **Subject** | | **IOT** | | | | | |
| **Max. Marks** | | **30** | | **Score** | | **28** | |
| **Certification Course Summary** | | | | | | | |
| **Course** | Introduction to Hadoop 3.0 | | | | | | |
| **Certificate Provider** | | | **Great learning**  **Academy** | **Duration** | | | **6 hours** |
| **Coding Challenges** | | | | | | | |
| **Problem Statement:** Given an array arr[] of the positive integers of size N, the task is to find the largest element on the left side of each index which is smaller than the element present at that index. | | | | | | | |
| **Status: COMPLETED** | | | | | | | |
| **Uploaded the report in Github** | | | | **YES** | | | |
| **If yes Repository name** | | | | **Akash\_Daily\_Progress** | | | |
| **Uploaded the report in slack** | | | | **YES** | | | |

Online Test Details:

Snapshot of test



Certification Course Details:



#### Introduction to Hadoop 3.0

Coding Challenges Details

Program 1...Given an array arr[] of the positive integers of size N, the task is to find the largest element on the left side of each index which is smaller than the element present at that index.

import java.util.\*;

class GFG{

// Function to find the

// Largest element before

// every element of an array

static void findMaximumBefore(int arr[],

int n){

// Loop to iterate over every

// element of the array

for (int i = 0; i < n; i++) {

int currAns = -1;

// Loop to find the maximum smallest

// number before the element arr[i]

for (int j = i - 1; j >= 0; j--) {

if (arr[j] > currAns &&

arr[j] < arr[i]) {

currAns = arr[j];

}

}

System.out.print(currAns+ " ");

}

}

// Driver Code

public static void main(String[] args)

{

int arr[] = { 4, 7, 6, 8, 5 };

int n = arr.length;

// Function Call

findMaximumBefore(arr, n);

}

}