

DAILY ONLINE ACTIVITIES SUMMARY

| | | | |
|---|--|---|-------------------|
| Date: | 21-05-2020 | Name: | D Jasmine Joyline |
| Sem & Sec | VI A | USN: | 4AL17CS024 |
| Online Test Summary | | | |
| Subject | Operating System | | |
| Max. Marks | 30 | Score | 30 |
| Certification Course Summary | | | |
| Course | Python Bootcamp 2020:Build 15 working applications and games | | |
| Certificate Provider | Udemy | Duration | 32hr |
| Coding Challenges | | | |
| Problem Statement: 1. Write a java program to implement Round Robin Scheduling. 2. Write a Java Program to Demonstrate a Basic Calculator using Applet. 3. Write a simple code to identify given linked list is palindrome or not by using stack. | | | |
| Status: Completed | | | |
| Uploaded the report in Github | | Yes | |
| If yes Repository name | | https://github.com/alvas-education-foundation/D_Jasmine_Joyline/tree/master/daily_progress | |
| Uploaded the report in slack | | Yes | |

Online Test Details:

OS IA TEST

9:39 4G 0.00 KB/S VO LTE   

  70

Largest Tech Community | Hackathons,...

[Logout](#)

Test Completed!

You have successfully participated in CSE-17CS64-OS-IA1.

Rate this Test

Your Rating: ★★★★★ ◀ Click to Rate

[Results](#)


[Analytics](#)

 MCQ

Your Score


30 / 30


Certification Course Details:









PYTHON
Complete
Bootcamp




Learn by building




Lectures More 

Section 14 - Libraries 

96  Introduction to this module 
 Video - 02:23 mins

97  Libraries 
 Video - 06:35 mins

98  Modules 
 Video - 04:20 mins

99  Json files 
 Video - 07:33 mins

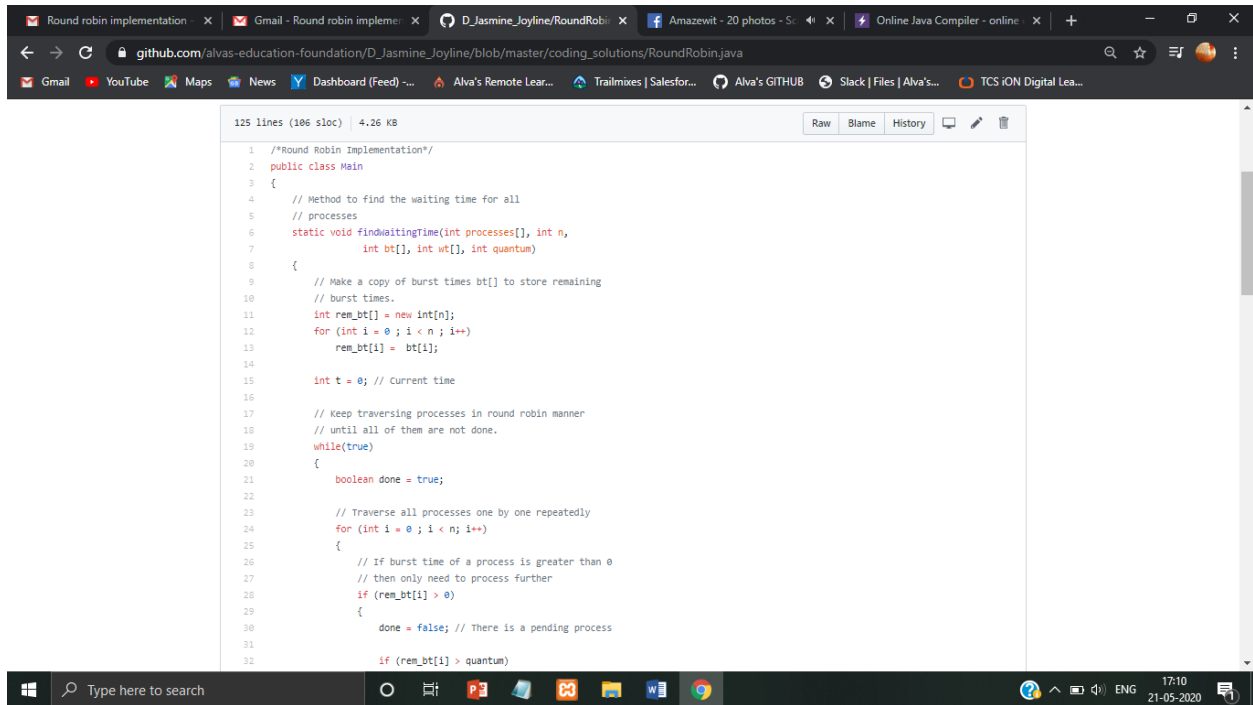
100 Libraries notes
Article

The modules that I have covered today:

- Decision Making Statements
- Loops
- Date and Time
- File handling
- Libraries

Coding Challenges Details:

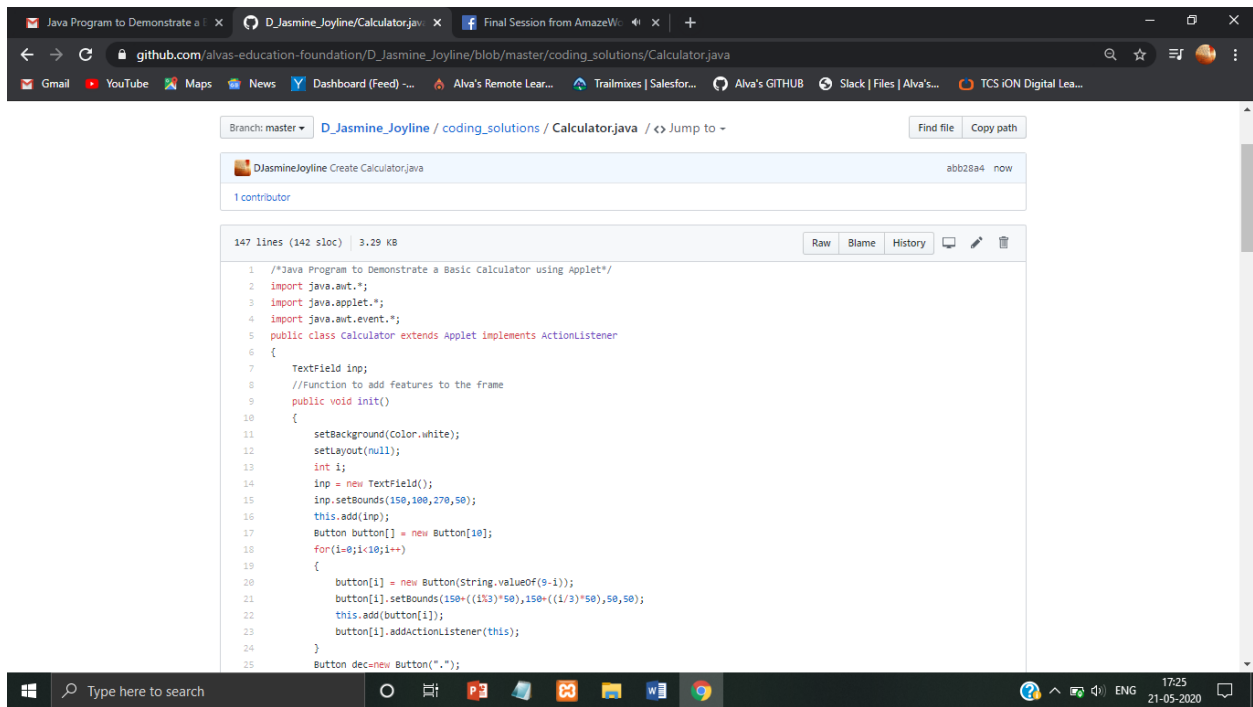
1. Write a java program to implement Round Robin Scheduling.



The screenshot shows a web browser with multiple tabs. The active tab is a GitHub repository page for 'D_Jasmine_Joyline/RoundRobin.java'. The page displays the source code for a Java program implementing Round Robin Scheduling. The code is 125 lines long (106 sloc) and 4.26 KB in size. The code includes a 'Main' class with a 'findWaitingTime' method that calculates the waiting time for processes based on their burst times and a quantum value. The method uses a while loop to traverse processes in a round robin manner until all processes are completed.

```
1  /*Round Robin Implementation*/
2  public class Main
3  {
4      // Method to find the waiting time for all
5      // processes
6      static void findWaitingTime(int processes[], int n,
7          int bt[], int wt[], int quantum)
8      {
9          // Make a copy of burst times bt[] to store remaining
10         // burst times.
11         int rem_bt[] = new int[n];
12         for (int i = 0 ; i < n ; i++)
13             rem_bt[i] = bt[i];
14
15         int t = 0; // Current time
16
17         // Keep traversing processes in round robin manner
18         // until all of them are not done.
19         while(true)
20         {
21             boolean done = true;
22
23             // Traverse all processes one by one repeatedly
24             for (int i = 0 ; i < n ; i++)
25             {
26                 // If burst time of a process is greater than 0
27                 // then only need to process further
28                 if (rem_bt[i] > 0)
29                 {
30                     done = false; // There is a pending process
31
32                     if (rem_bt[i] > quantum)
```

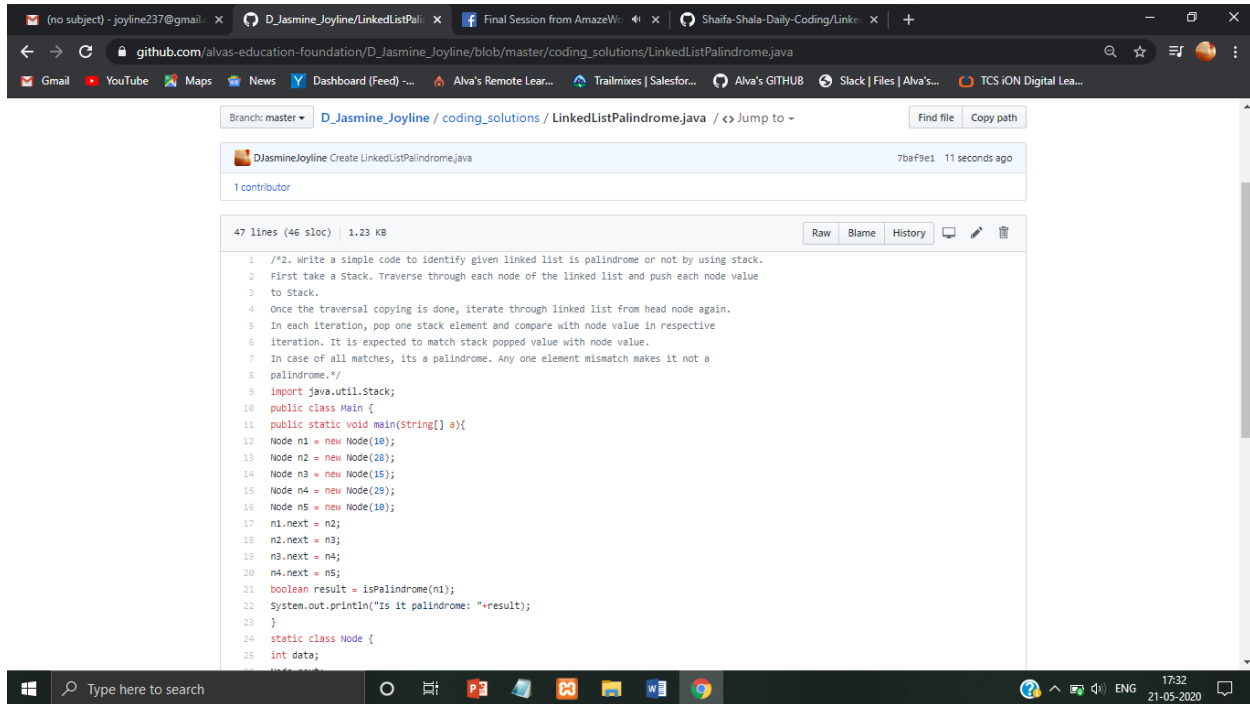
2. Write a Java Program to Demonstrate a Basic Calculator using Applet.



The screenshot shows a web browser with multiple tabs. The active tab is a GitHub repository page for 'D_Jasmine_Joyline/Calculator.java'. The page displays the source code for a Java program demonstrating a basic calculator using an Applet. The code is 147 lines long (142 sloc) and 3.29 KB in size. The code includes a 'Calculator' class that extends 'Applet' and implements 'ActionListener'. The class contains a 'TextField' for input and a 'Button' array for digits and operations. The 'init' method sets up the GUI, and the 'actionPerformed' method handles the calculator logic.

```
1  /*Java Program to Demonstrate a Basic Calculator using Applet*/
2  import java.awt.*;
3  import java.applet.*;
4  import java.awt.event.*;
5  public class Calculator extends Applet implements ActionListener
6  {
7      TextField inp;
8      //Function to add features to the frame
9      public void init()
10     {
11         setBackground(Color.white);
12         setLayout(null);
13         int i;
14         inp = new TextField();
15         inp.setBounds(150,100,270,50);
16         this.add(inp);
17         Button button[] = new Button[10];
18         for(i=0;i<10;i++)
19         {
20             button[i] = new Button(String.valueOf(i));
21             button[i].setBounds(150+((i/3)*50),150+((i/3)*50),50,50);
22             this.add(button[i]);
23             button[i].addActionListener(this);
24         }
25         Button dec=new Button(".");
```

3. Write a simple code to identify given linked list is palindrome or not by using stack.



The screenshot shows a web browser displaying a GitHub repository page for a Java solution. The repository is named "D_Jasmine_Joyline/LinkedListPalindrome.java" and is located at the path "coding_solutions/LinkedListPalindrome.java". The page shows the commit history with one commit by "D_Jasmine_Joyline" titled "Create LinkedListPalindrome.java" with commit hash "7baf9e1" and a timestamp of "11 seconds ago". The code is displayed in a light blue editor with a dark background. The code is a Java program that uses a stack to check if a linked list is a palindrome. It includes a main method that creates a linked list with five nodes containing values 10, 28, 15, 29, and 10. It then calls a method "isPalindrome" to check if the list is a palindrome and prints the result. The code is 47 lines long (46 sloc) and 1.23 KB in size. The code is as follows:

```
1  /*2. Write a simple code to identify given linked list is palindrome or not by using stack.
2  First take a Stack. Traverse through each node of the linked list and push each node value
3  to Stack.
4  Once the traversal copying is done, iterate through linked list from head node again.
5  In each iteration, pop one stack element and compare with node value in respective
6  iteration. It is expected to match stack popped value with node value.
7  In case of all matches, its a palindrome. Any one element mismatch makes it not a
8  palindrome.*/
9  import java.util.Stack;
10 public class Main {
11     public static void main(String[] a){
12         Node n1 = new Node(10);
13         Node n2 = new Node(28);
14         Node n3 = new Node(15);
15         Node n4 = new Node(29);
16         Node n5 = new Node(10);
17         n1.next = n2;
18         n2.next = n3;
19         n3.next = n4;
20         n4.next = n5;
21         boolean result = isPalindrome(n1);
22         System.out.println("Is it palindrome: "+result);
23     }
24     static class Node {
25         int data;
26         Node next;
```