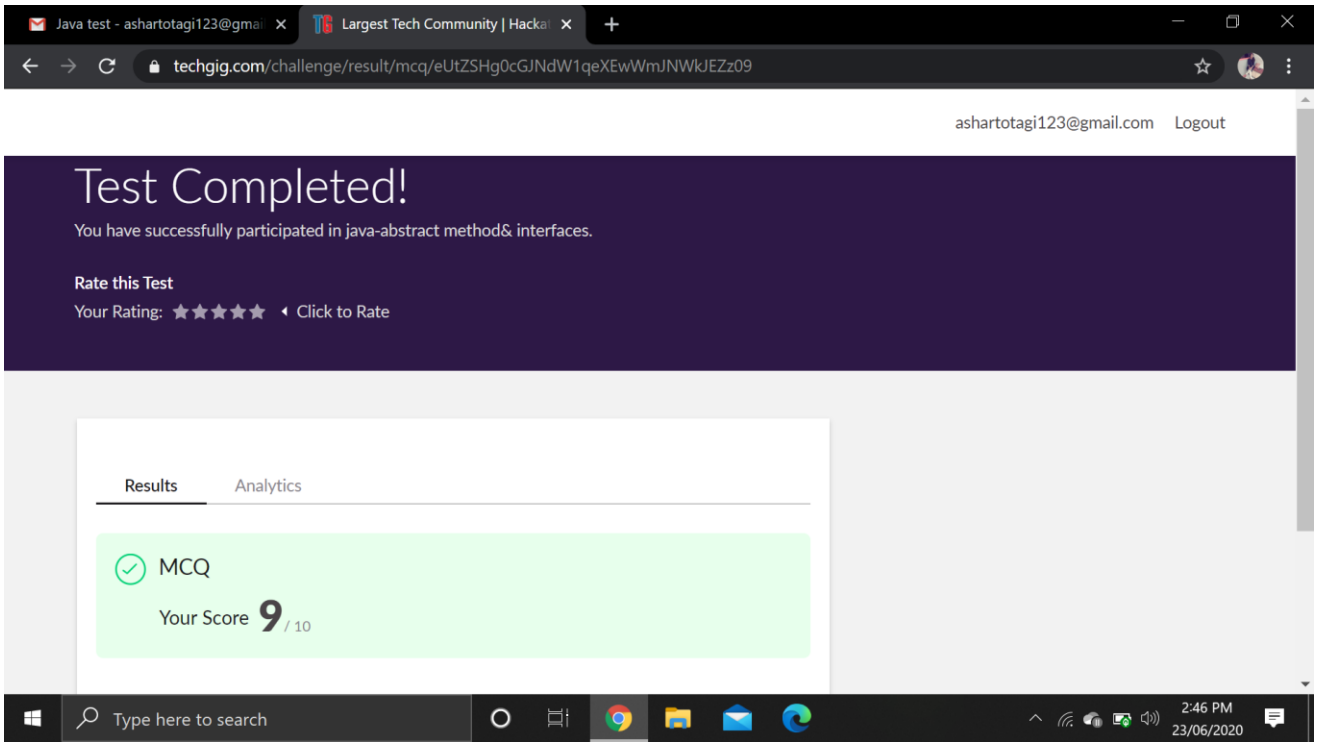
DAILY ONLINE ACTIVITIES SUMMARY

Date:	**23 June 2020**	**Name:**	**Asha Rudrappa Totagi**
Sem& Sec	**6thsem& A sec**	**USN:**	**4AL17CS015**
Pre-Placement Training Summary			
Subject	**9:15 am to 11:15 am – Java and J2EE** **11:15 am to 1:30pm - Programming in C++**		
Faculty	**1. Mr. Sharan** **2. Ms. Merlyn Methais**	**Duration**	**1. 2 hours** **2. 2 hours 15 mins**
Online Coading			
Problem Statement **Program 1:** Write a Java Program to traverse a binary tree using PreOrder traversal without recursion.			
Status: DONE			
Uploaded the report in Github	**YES**		
If yes Repository name	**Daily Status**		
Uploaded the report in slack	**YES**		

Online Pre-placement Training



Coding Challenges Details:

Program 1:

```
import java.util.Stack;
public class Main { public static void main(String[] args)
throws Exception {
// construct the binary tree given in question
BinaryTree bt = BinaryTree.create();
// traversing binary tree in PreOrder without using recursion
System.out .println("printing nodes of a binary tree in preOrder using recursion");
bt.preOrderWithoutRecursion();
}
}
class BinaryTree
{
 static class TreeNode
{
String data;
 TreeNode left, right;
TreeNode(String value)
{
 this.data = value;
left = right = null;
}
 boolean isLeaf()
 {
 return left == null ? right == null : false;
 }
 }
 // root of binary tree
TreeNode root;
/** * Java method to visit tree nodes in PreOrder traversal without recursion. */
public void preOrderWithoutRecursion()
{
 Stack<TreeNode> nodes = new Stack<>();
nodes.push(root);
while (!nodes.isEmpty())
```

```
{
 TreeNode current = nodes.pop();
 System.out.printf("%s ", current.data);
 if (current.right != null)
{
 nodes.push(current.right);
 }
 if (current.left != null)
 {
 nodes.push(current.left);
 }
 }
 }
 /** * Java method to create binary tree with test data * * @return a sample binary tree for testing */
public static BinaryTree create()
 {
 BinaryTree tree = new BinaryTree();
 TreeNode root = new TreeNode("a");
 tree.root = root;
tree.root.left = new TreeNode("b");
 tree.root.left.left = new TreeNode("c");
 tree.root.left.right = new TreeNode("d");
 tree.root.right = new TreeNode("e");
 tree.root.right.right = new TreeNode("f"); return tree;
}
 }
```