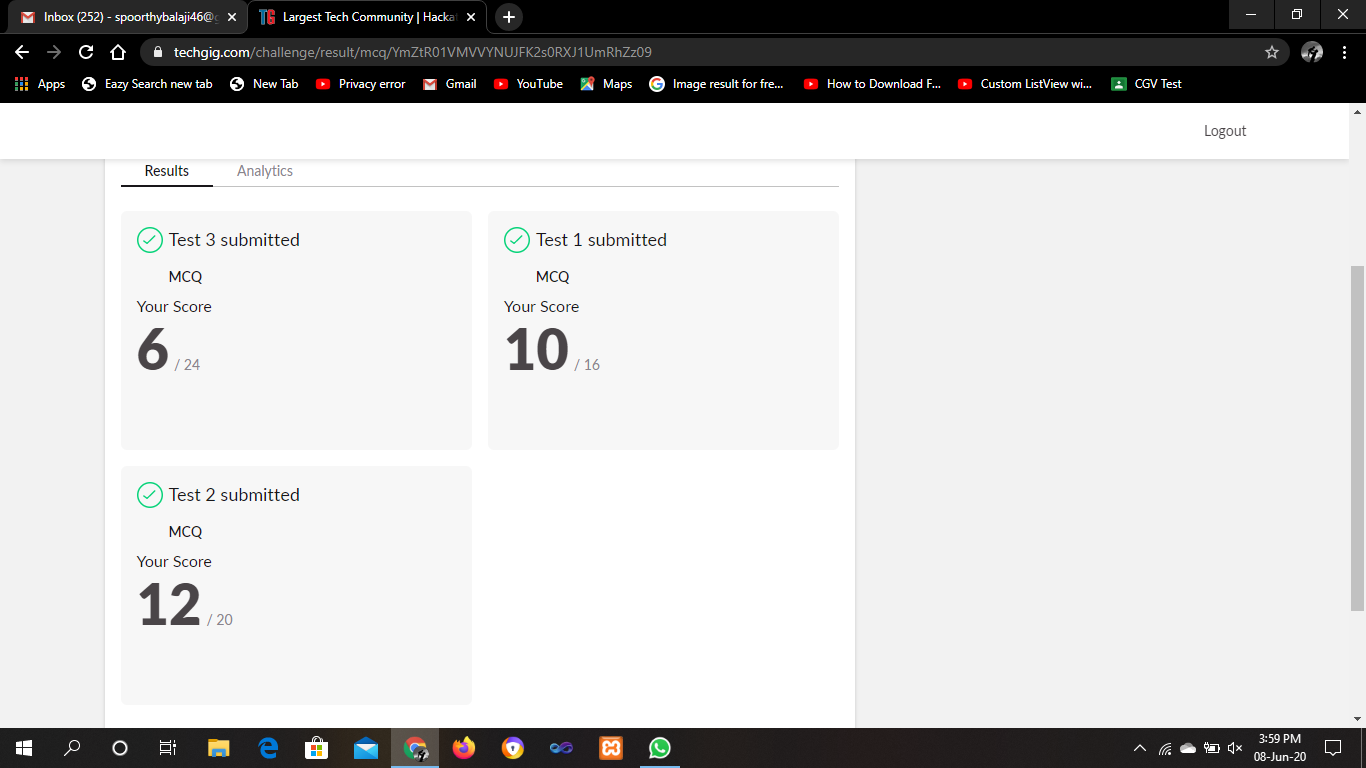
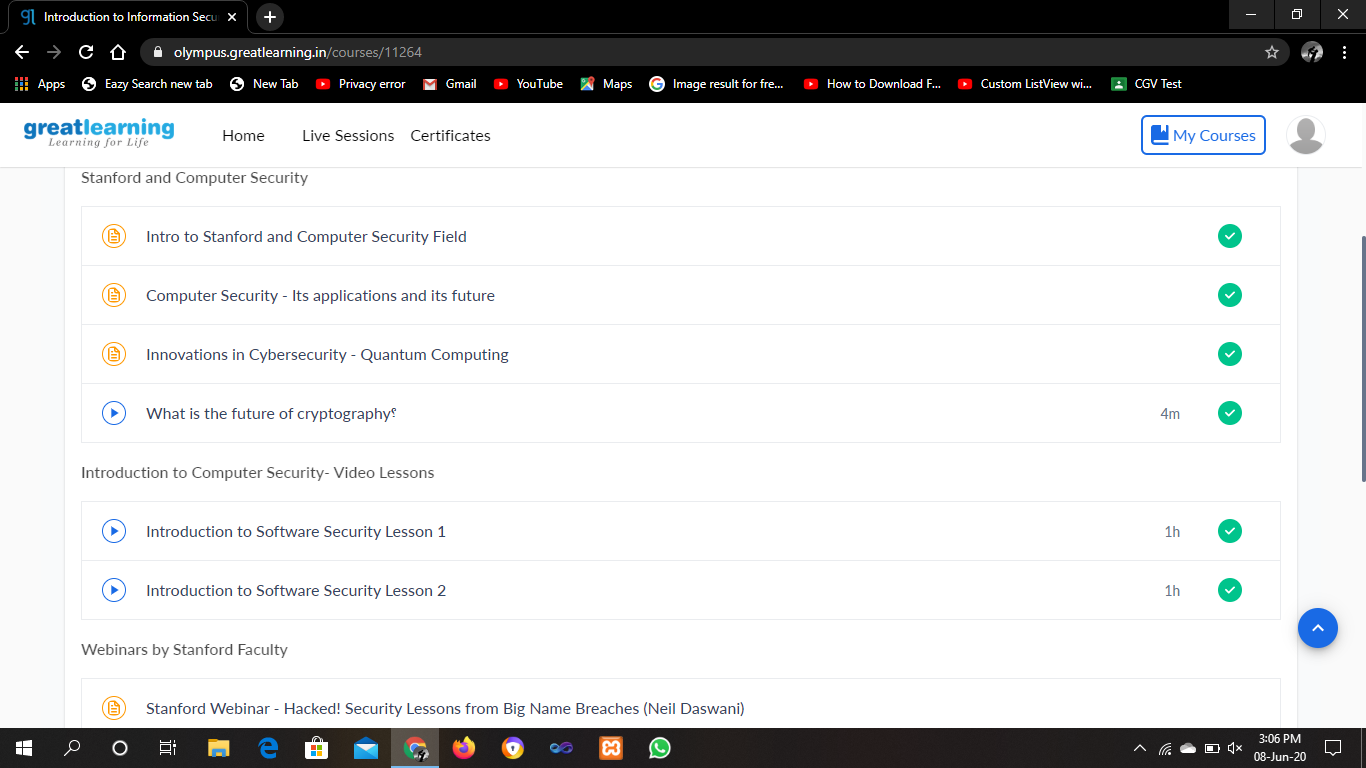
**DAILY ONLINE ACTIVITIES SUMMARY**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Date:** | **08/06/2020** | | | | | **Name:** | **Spoorthy Balaji** | |
| **Sem & Sec** | **6th & B** | | | | | **USN:** | **4al17cs098** | |
| **Online Test Summary** | | | | | | | | |
| **Subject** | | **CNSC** | | | | | | |
| **Max. Marks** | | **60** | | **Score** | | | **28** | |
| **Certification Course Summary** | | | | | | | | |
| **Course** | **Introduction to Information Security** | | | | | | | |
| **Certificate Provider** | | | **Great Learning Academy** | | **Duration** | | | **6hours** |
| **Coding Challenges** | | | | | | | | |
| **Problem Statement: 5** Programs | | | | | | | | |
| **Status: Solved** | | | | | | | | |
| **Uploaded the report in Github** | | | | | **yes** | | | |
| **If yes Repository name** | | | | | <https://github.com/spoorthybalaji/Daily_Status> | | | |
| **Uploaded the report in slack** | | | | | **yes** | | | |

IA TEST



CERTIFICATION COURSE



ONLINE CODING

**1**. **Java program to delete a node from the middle of the singly linked list**

public class deleteMid{

//Represent a node of the singly linked list

class Node{

int data;

Node next;

public Node(int data)

{

this.data = data;

this.next = null;

}

}

//Represent the head and tail of the singly linked list

public Node head = null;

public Node tail = null;

public int size;

//addNode() will add a new node to the list

public void addNode(int data) {

//Create a new node

Node newNode = new Node(data);

//Checks if the list is empty

if(head == null) {

//If list is empty, both head and tail will point to new node

head = newNode;

tail = newNode;

}

else {

//newNode will be added after tail such that tail's next will point to newNode

tail.next = newNode;

//newNode will become new tail of the list

tail = newNode;

}

size++;

}

//deleteFromMid() will delete a node from the middle of the list

void deleteFromMid() {

Node temp, current;

//Checks if the list is empty

if(head == null) {

System.out.println("List is empty");

return;

}

else {

//Store the mid position of the list

int count = (size % 2 == 0) ? (size/2) : ((size+1)/2);

//Checks whether the head is equal to the tail or not, if yes then the list has only one node.

if( head != tail ) {

//Initially, temp will point to head

temp = head;

current = null;

//Current will point to node previous to temp

//If temp is pointing to node 2 then current will point to node 1.

for(int i = 0; i< count-1; i++){

current = temp;

temp = temp.next;

}

if(current != null) {

//temp is the middle that needs to be removed.

//So, current node will point to node next to temp by skipping temp.

current.next = temp.next;

//Delete temp

temp = null;

}

//If current points to NULL then, head and tail will point to node next to temp.

else {

head = tail = temp.next;

//Delete temp

temp = null;

}

}

//If the list contains only one element

//then it will remove it and both head and tail will point to NULL

else {

head = tail = null;

}

}

size--;

}

//display() will display all the nodes present in the list

public void display() {

//Node current will point to head

Node current = head;

if(head == null) {

System.out.println("List is empty");

return;

}

while(current != null) {

//Prints each node by incrementing pointer

System.out.print(current.data + " ");

current = current.next;

}

System.out.println();

}

public static void main(String[] args) {

deleteMidsList = new deleteMid();

//Adds data to the list

sList.addNode(1);

sList.addNode(2);

sList.addNode(3);

sList.addNode(4);

//Printing original list

System.out.println("Original List: ");

sList.display();

while(sList.head != null) {

sList.deleteFromMid();

//Printing updated list

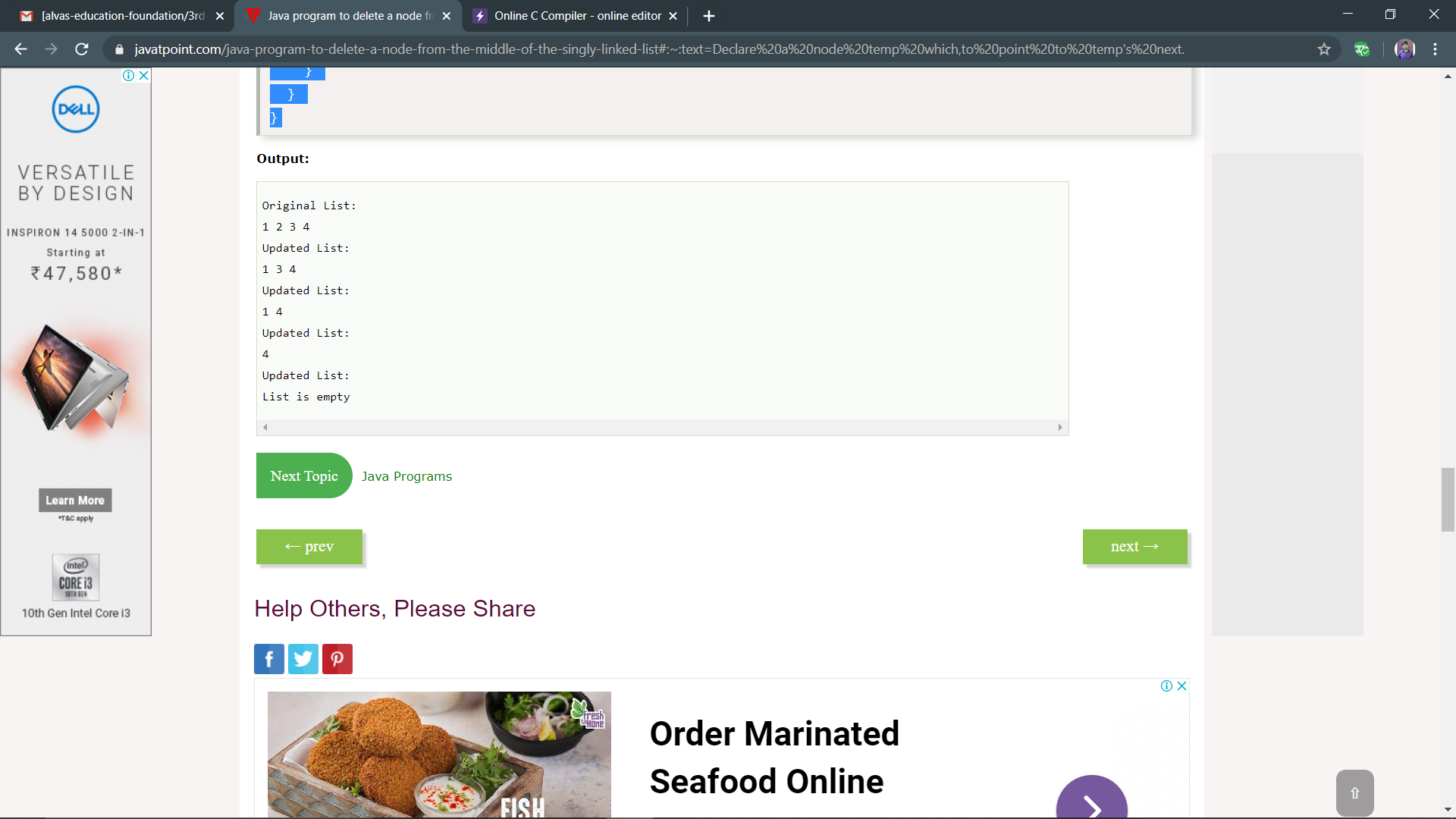
System.out.println("Updated List: ");

sList.display();

}

}

}



**2. Write C++ program to Check whether a number can be represented as difference of two squares**

#include <bits/stdc++.h>

using namespace std;

bool difSquare(int n)

{

// Checking if n % 4 = 2 or not

if (n % 4 != 2) {

return true;

}

return false;

}

int main()

{

int n;

std::cout<<"enter the number: ";

std::cin>>n;

if (difSquare(n)) {

cout<< "Yes\n";

}

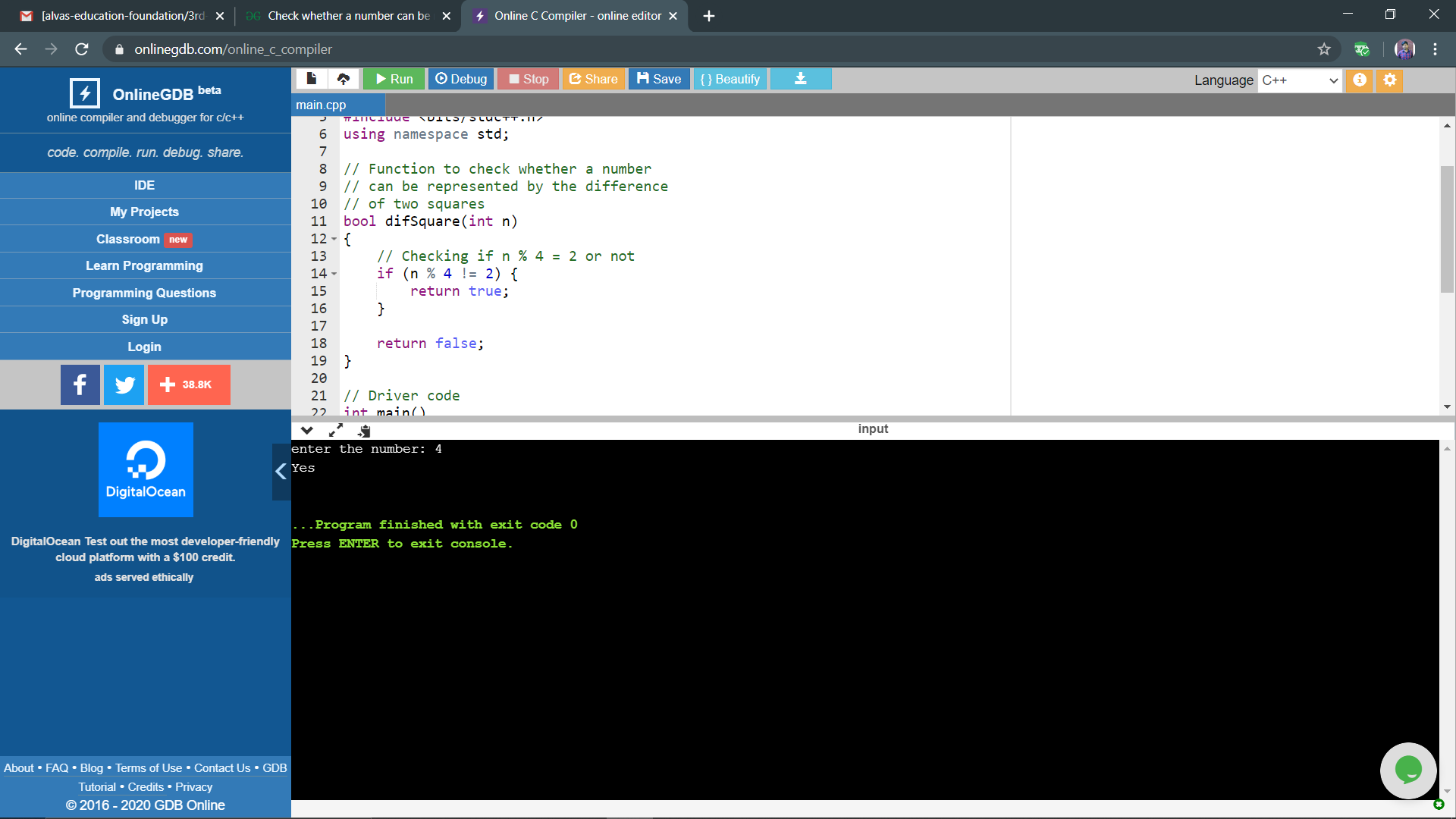
else {

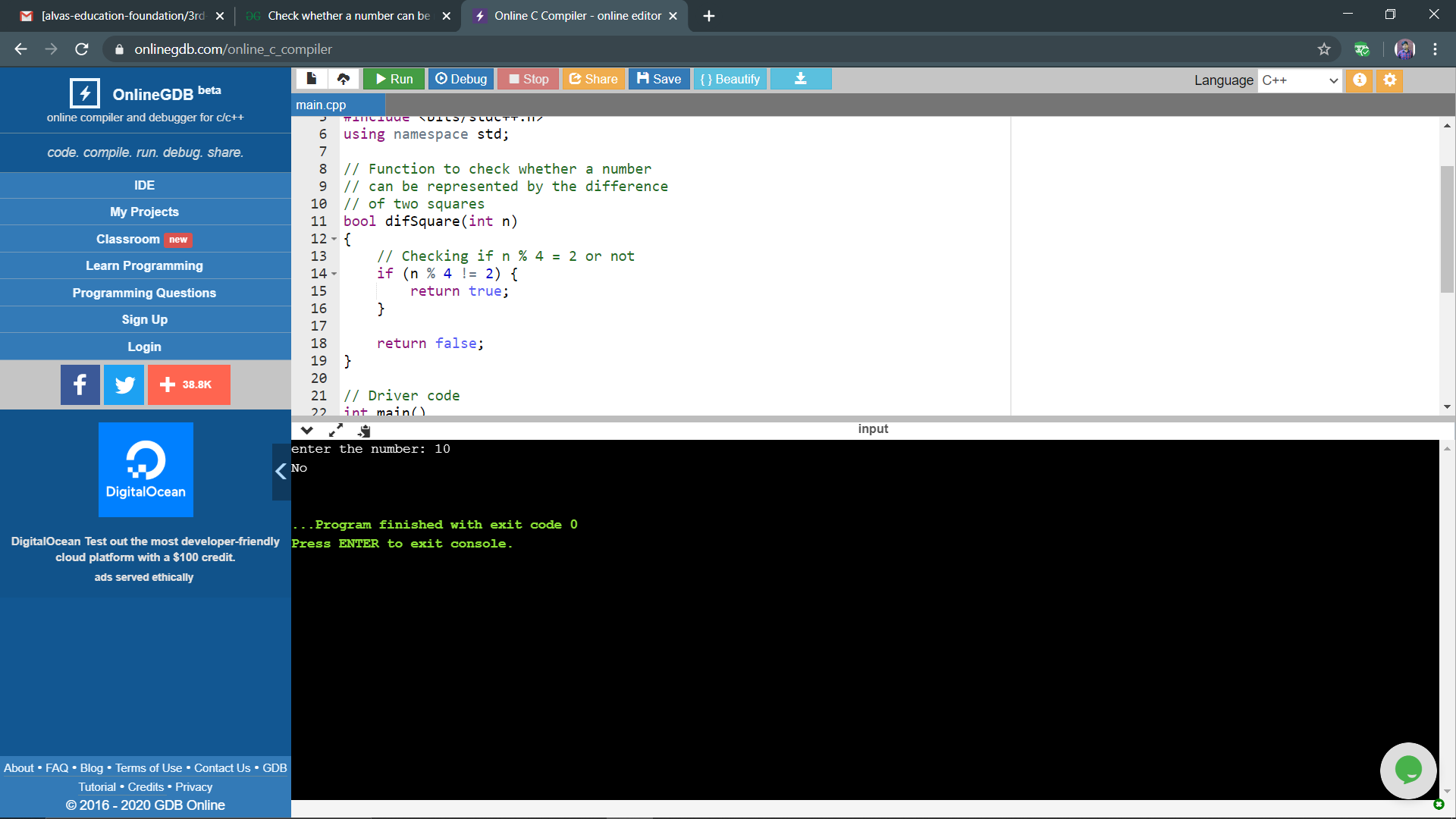
cout<< "No\n";

}

return 0;

}

****

****

**3. C Program to Generate All the Set Partitions of n Numbers Beginning from 1 and so on**

#include <stdio.h>

#include <stdlib.h>

typedef struct {

int first;

int n;

int level;

} Call;

void print(int n, int \* a) {

int i ;

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

printf("%d", a[i]);

}

printf("\n");

}

void integerPartition(int n, int \* a){

int first;

int i;

int top = 0;

int level = 0;

Call \* stack = (Call \* ) malloc (sizeof(Call) \* 1000);

stack[0].first = -1;

stack[0].n = n;

stack[0].level = level;

while (top >= 0){

first = stack[top].first;

n = stack[top].n;

level = stack[top].level;

if (n >= 1) {

if (first == - 1) {

a[level] = n;

print(level, a);

first = (level == 0) ? 1 : a[level-1];

i = first;

} else {

i = first;

i++;

}

if (i<= n / 2) {

a[level] = i;

stack[top].first = i;

top++;

stack[top].first = -1;

stack[top].n = n - i;

stack[top].level = level + 1;

} else {

top--;

}

} else {

top --;

}

}

}

int main(){

int N = 1;

int \* a = (int \* ) malloc(sizeof(int) \* N);

int i;

printf("\nEnter a number N to generate all set partition from 1 to N: ");

scanf("%d", &N);

for ( i = 1; i<= N; i++)

{

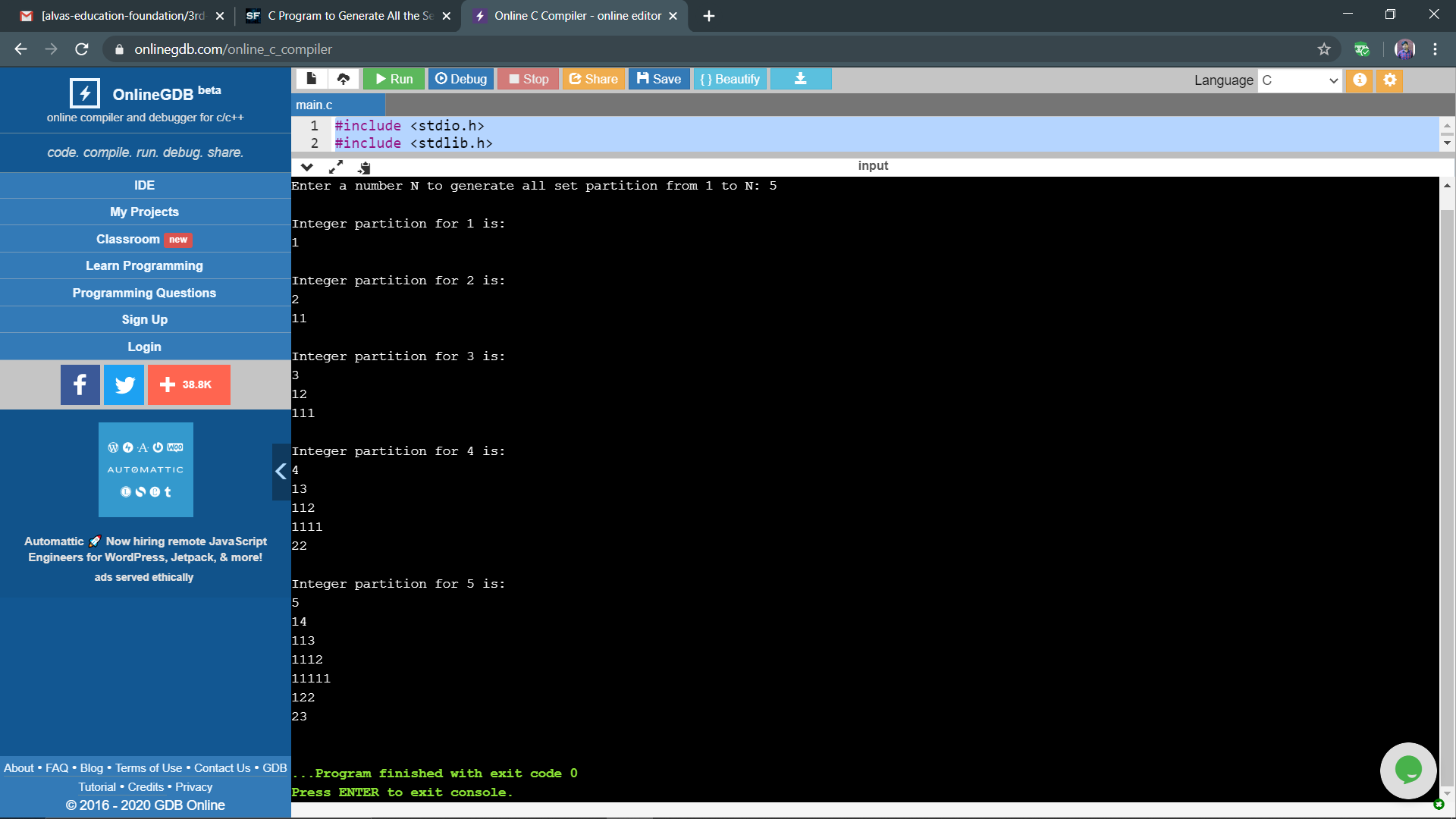
printf("\nInteger partition for %d is: \n", i);

integerPartition (i, a);

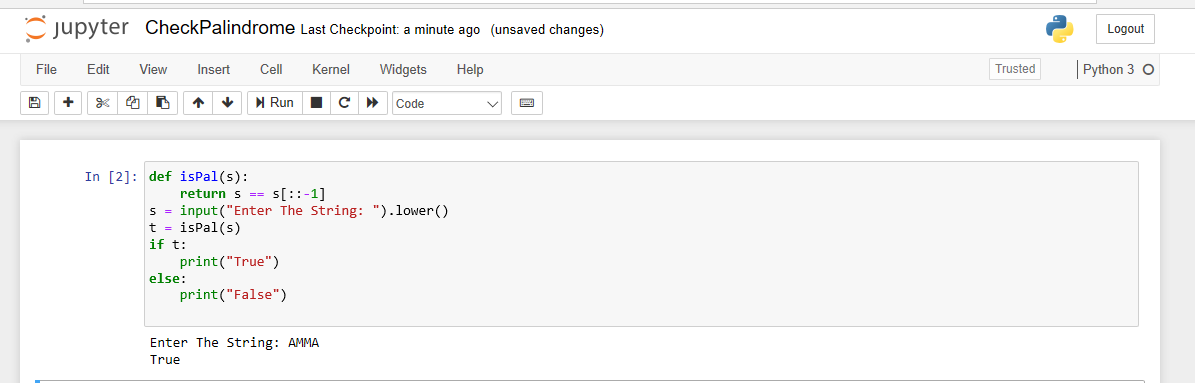
}

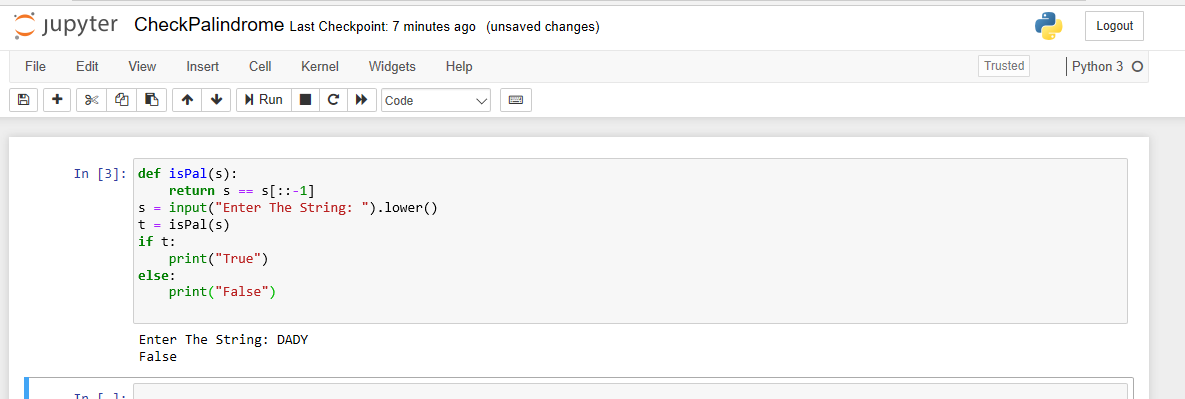
return(0);

}



**4. Write a python function that will take a string and checks whether it is a palindrome or not. Return If it a palindrome, print true else print false.**

****



5. **Python Program to Check Whether a String is a Palindrome or not Using Recursion**

