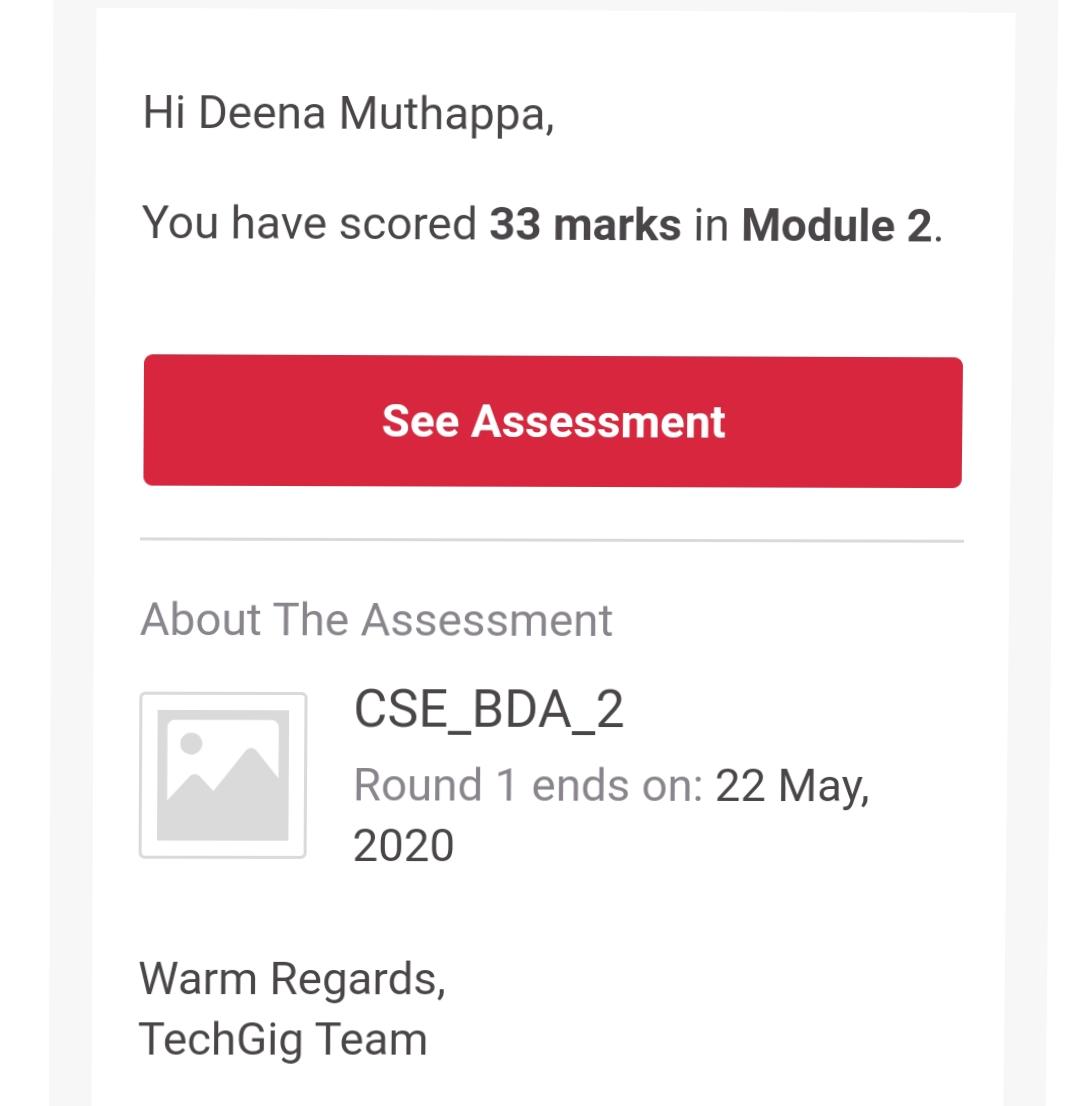
**DAILY ONLINE ACTIVITIES SUMMARY**

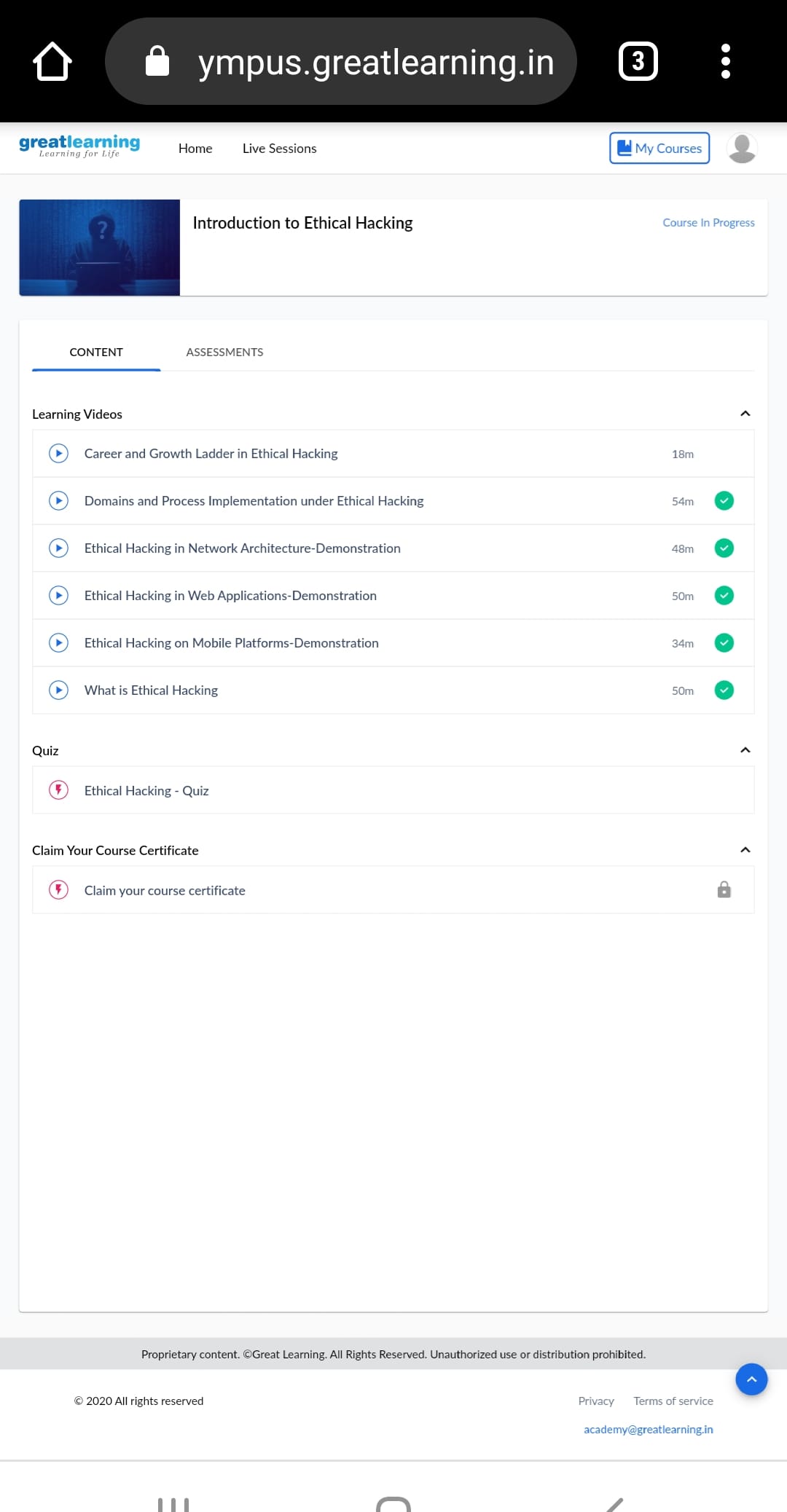
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Date:** | **22/05/2020** | | | | **Name:** | **Deena Muthappa** | |
| **Sem & Sec** | **8th sem, A sec** | | | | **USN:** | **4AL16CS028** | |
| **Online Test Summary** | | | | | | | |
| **Subject** | | **BDA 2** | | | | | |
| **Max. Marks** | | **40** | | **Score** | | **33** | |
| **Certification Course Summary** | | | | | | | |
| **Course** | **Introduction to Ethical Hacking** | | | | | | |
| **Certificate Provider** | | | **Great Learning** | **Duration** | | | **254 mins** |
| **Coding Challenges** | | | | | | | |
| **Write a C Program to implement various operations of Singly Linked List Stack** | | | | | | | |
| **Status: Completed** | | | | | | | |
| **Uploaded the report in Github** | | | | **Yes** | | | |
| **If yes Repository name** | | | | **deenamuthappa/Coding-Challenges** | | | |
| **Uploaded the report in slack** | | | | **Yes** | | | |

Online Test Details:



Online examination is conducting a test online to measure the knowledge of the participants on a given topic. With online examination students can do the exam online. The test was on the subject Big Data Analytics under Module 2.

Certification Course Details:



The ethical hacker plays a key role in the security process. The methodology used to secure an organization can be broken down into five key steps. Ethical hacking is addressed in the first:

Assessment—Ethical hacking, penetration testing, and hands-on security tests.

Policy Development—Development of policy based on the organization’s goals and mission. The focus should be on the organization’s critical assets.

Implementation—The building of technical, operational, and managerial controls to secure key assets and data.

Training—Employees need to be trained as to how to follow policy and how to configure key security controls, such as Intrusion Detection Systems (IDS) and firewalls.

Audit—Auditing involves periodic reviews of the controls that have been put in place to provide good security.

Coding Challenges Details:

#include <stdio.h>  
#include <stdlib.h>

struct node  
{  
int info;  
struct node \*ptr;  
}\*top,\*top1,\*temp;

int topelement();  
void push(int data);  
void pop();  
void empty();  
void display();  
void destroy();  
void stack\_count();  
void create();

int count = 0;

void main()  
{  
int no, ch, e;

printf("\n 1 - Push");

printf("\n 2 - Pop");

printf("\n 3 - Top");

printf("\n 4 - Empty");

printf("\n 5 - Exit");

printf("\n 6 - Dipslay");

printf("\n 7 - Stack Count");

printf("\n 8 - Destroy stack");

create();

while (1)

{

printf("\n Enter choice : ");

scanf("%d", &ch);

switch (ch)

{

case 1:

printf("Enter data : ");

scanf("%d", &no);

push(no);

break;

case 2:

pop();

break;

case 3:

if (top == NULL)

printf("No elements in stack");

else

{

e = topelement();

printf("\n Top element : %d", e);

}

break;

case 4:

empty();

break;

case 5:

exit(0);

case 6:

display();

break;

case 7:

stack\_count();

break;

case 8:

destroy();

break;

default :

printf(" Wrong choice, Please enter correct choice ");

break;

}

}

}

/\* Create empty stack \*/  
void create()  
{  
top = NULL;  
}

/\* Count stack elements \*/  
void stack\_count()  
{  
printf("\n No. of elements in stack : %d", count);  
}

/\* Push data into stack \*/  
void push(int data)  
{  
if (top == NULL)  
{  
top =(struct node )malloc(1sizeof(struct node));  
top->ptr = NULL;  
top->info = data;  
}  
else  
{  
temp =(struct node )malloc(1sizeof(struct node));  
temp->ptr = top;  
temp->info = data;  
top = temp;  
}  
count++;  
}

/\* Display stack elements \*/  
void display()  
{  
top1 = top;

if (top1 == NULL)

{

printf("Stack is empty");

return;

}

while (top1 != NULL)

{

printf("%d ", top1->info);

top1 = top1->ptr;

}

}

/\* Pop Operation on stack \*/  
void pop()  
{  
top1 = top;

if (top1 == NULL)

{

printf("\n Error : Trying to pop from empty stack");

return;

}

else

top1 = top1->ptr;

printf("\n Popped value : %d", top->info);

free(top);

top = top1;

count--;

}

/\* Return top element \*/  
int topelement()  
{  
return(top->info);  
}

/\* Check if stack is empty or not \*/  
void empty()  
{  
if (top == NULL)  
printf("\n Stack is empty");  
else  
printf("\n Stack is not empty with %d elements", count);  
}

/\* Destroy entire stack \*/  
void destroy()  
{  
top1 = top;

while (top1 != NULL)

{

top1 = top->ptr;

free(top);

top = top1;

top1 = top1->ptr;

}

free(top1);

top = NULL;

printf("\n All stack elements destroyed");

count = 0;