DAILY ONLINE ACTIVITIES SUMMARY

Date:	**22-05-2020**	**Name:**	**Nayan. P. Joshi**
Sem & Sec	**8th Sem A**	**USN:**	**4AL16CS058**

Online Test Summary

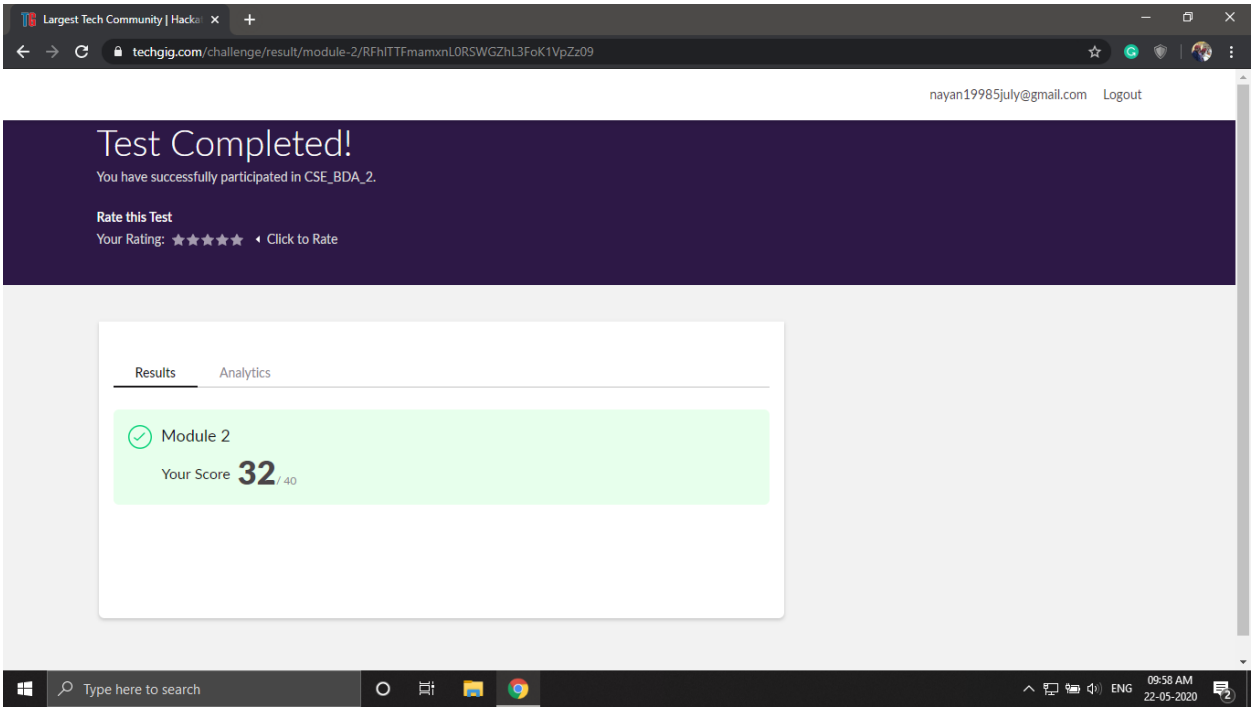
Subject	**Big Data Analytics**		
Max. Marks	**40**	**Score**	**32**

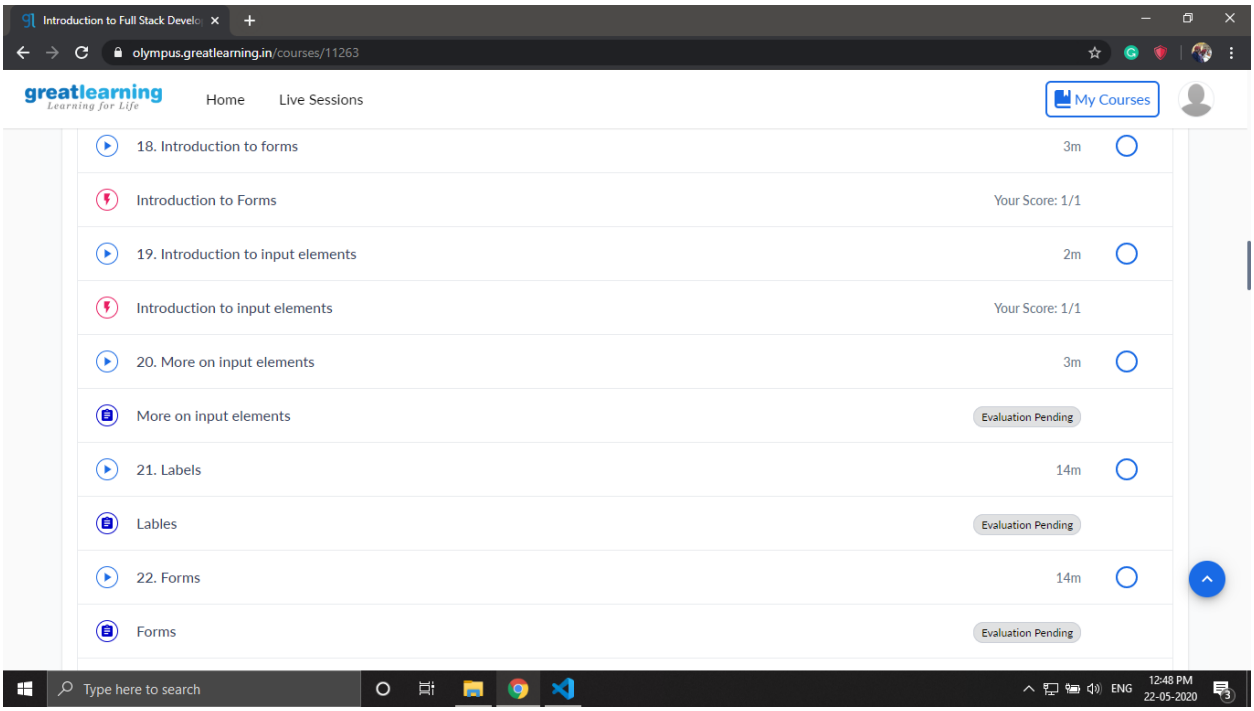
Certification Course Summary

Course	**Introduction to Full Stack Development**		
Certificate Provider	**Great learning academy**	**Duration**	**60hrs**

Coding Challenges

Problem Statement: C Program to implement various operations of Singly Linked List Stack	
Status: Solved	
Uploaded the report in GitHub	**yes**
If yes Repository name	**nayan1998**
Uploaded the report in slack	**yes**


Largest Tech Community | Hacke
techgig.com/challenge/result/module-2/RFhITTFmamxnL0RSWGZhL3FoK1VpZz09
nayan19985july@gmail.com Logout
Test Completed!
You have successfully participated in CSE_BDA_2.
Rate this Test
Your Rating: Click to Rate
Results
Analytics
Module 2
Your Score 32 / 40
Type here to search
ENG
09:58 AM
22-05-2020


Introduction to Full Stack Develo
olympus.greatlearning.in/courses/11263
greatlearning
Learning for Life
Home
Live Sessions
My Courses
18. Introduction to forms 3m
Introduction to Forms Your Score: 1/1
19. Introduction to input elements 2m
Introduction to input elements Your Score: 1/1
20. More on input elements 3m
More on input elements Evaluation Pending
21. Labels 14m
Lables Evaluation Pending
22. Forms 14m
Forms Evaluation Pending
Type here to search
ENG
12:48 PM
22-05-2020

C Program to implement various operations of Singly Linked List Stack

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

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

void push(int data);
void pop();
void display();
void create();

int count = 0;

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

   printf("\n 1 - Push");
   printf("\n 2 - Pop");
```

```
printf("\n 3 - Display");
printf("\n 4 - Destroy");
printf("\n 5 - Exit");

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:
      display();
```

```
            break;
        case 4:
                    destroy();
                        break;
        case 5:
            exit(0);
        default :
            printf("Invalid Input");
            break;
        }
    }
}


void create()
{
    top = NULL;
}

void push(int data)
{
    if (top == NULL)
    {
```

```
        top =(struct node *)malloc(1*sizeof(struct node));
        top->ptr = NULL;
        top->info = data;
    }
    else
    {
        temp =(struct node *)malloc(1*sizeof(struct node));
        temp->ptr = top;
        temp->info = data;
        top = temp;
    }
    count++;
}


void display()
{
    top1 = top;

    if (top1 == NULL)
    {
        printf("Stack is empty");
        return;
```

```
    }

    while (top1 != NULL)
    {
        printf("%d \n", top1->info);
        top1 = top1->ptr;
    }
 }

void pop()
{
    top1 = top;

    if (top1 == NULL)
    {
        printf("\n Error : Not Able to pop from empty stack");
        return;
    }
    else
        top1 = top1->ptr;
    printf("\n Popped value : %d", top->info);
    free(top);
    top = top1;
```

```
    count--;
}


int topelement()
{
    return(top->info);
}


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


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;
}
```