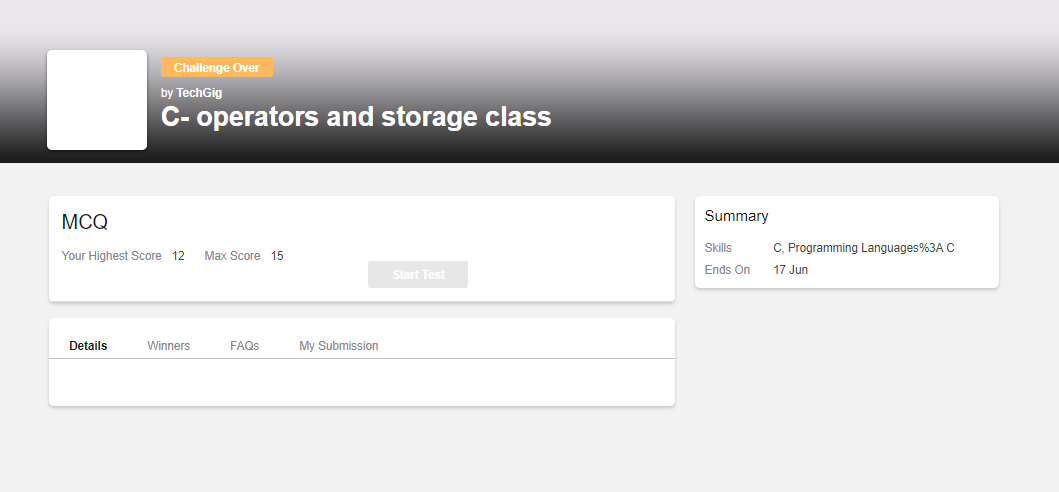
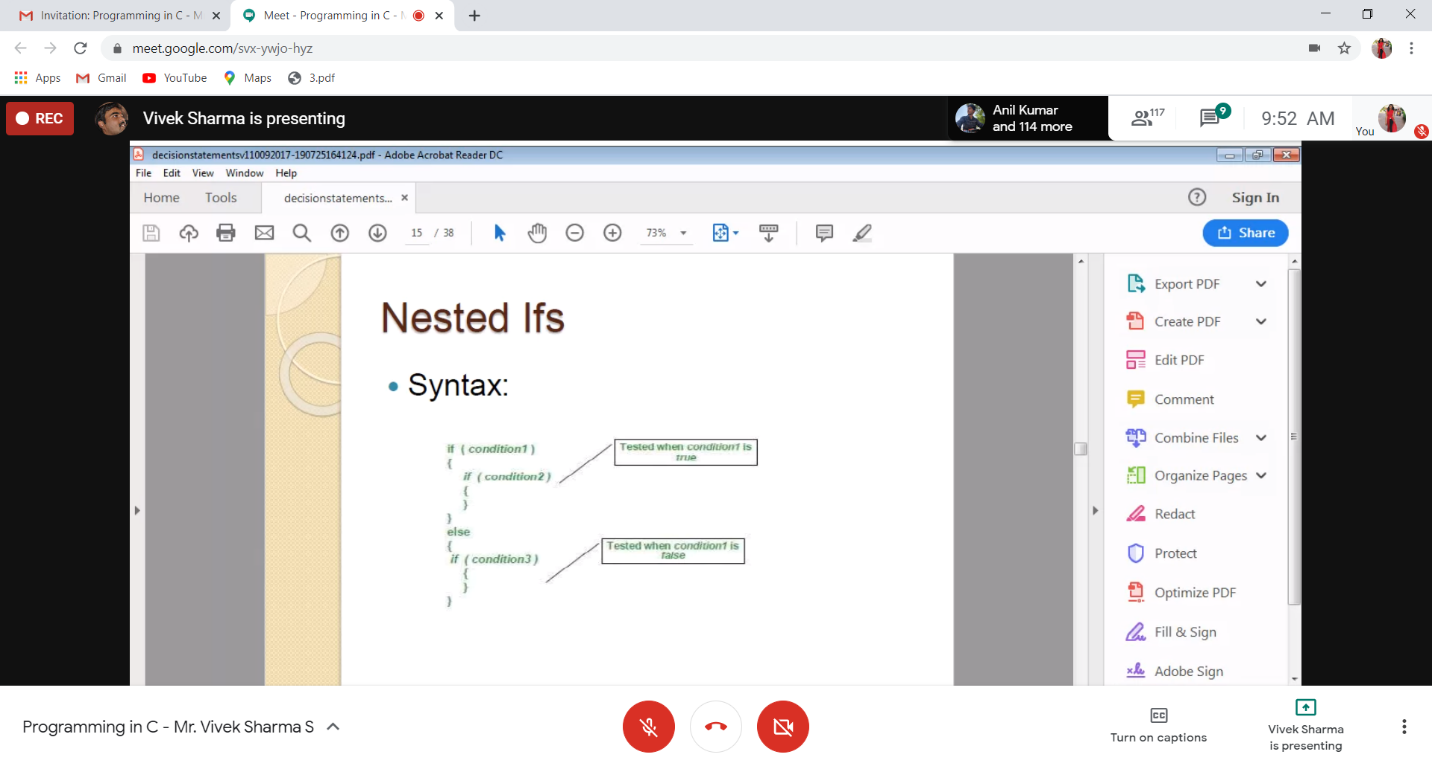
**DAILY ONLINE ACTIVITIES SUMMARY**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Date:** | **17-06-2020** | | | | | **Name:** | **Sneha K Bakale** | |
| **Sem & Sec** | **6th B** | | | | | **USN:** | **4al17cs095** | |
| **Online Test Summary** | | | | | | | | |
| **Subject** | | **C Programming** | | | | | | |
| **Max. Marks** | | **15** | | **Score** | | | **12** | |
| **Certification Course Summary** | | | | | | | | |
| **Course** | **Programming in C** Workshop on Applications of Python Programming in Data Analytics and Machine Learning | | | | | | | |
| **Certificate Provider** | | | **Mr.Vikek sharma**  **Dr.Badhusha** | | **Duration** | | | **4.0 hours** |
| **Coding Challenges** | | | | | | | | |
| **Problem Statement:**  **Python Programming in Data Analytics and Machine Learning exercise.** | | | | | | | | |
| **Status: Completed** | | | | | | | | |
| **Uploaded the report in Github** | | | | | **Yes** | | | |
| **If yes Repository name** | | | | | **Report-**[**https://github.com/Sneha35/OnlineCourse-And-Coding.git**](https://github.com/Sneha35/OnlineCourse-And-Coding.git)  **Python Programming-**  <https://github.com/Sneha35/Applications-of-python-programming-in-DA-and-ML.git> | | | |
| **Uploaded the report in slack** | | | | | **Yes** | | | |

Online Test Details: (Attach the snapshot and briefly write the report for the same) 

Certification Course Details: (Attach the snapshot and briefly write the report for the same) 

Coding Challenges Details: (Attach the snapshot and briefly write the report for the same) **1. Write a Java program to find the row, column position of a specified number (row, column position) in a given 2-dimensional array**

importjava.util.\*;

publicclassabc{

publicstaticvoidmain(String[]args){

intnums[][]={{12,20,30,40},

{15,25,35,45},

{24,29,39,51},

{35,30,39,50},

{50,60,75,72}};

int rows =5;

intsearch\_element=39;

intans[]=Saddleback(nums, rows -1,0,search\_element);

System.out.println("Position of "+search\_element+" in the matrix is ("+ans[0]+","+ans[1]+")");

}

privatestaticint[]Saddleback(intnums[][],int row,int col,intsearch\_element){

//numsay to store the row and column of the searched element

intelement\_pos[]={-1,-1};

if(row <0|| col >=nums[row].length){

returnelement\_pos;

}

if(nums[row][col]==search\_element){

element\_pos[0]= row;

element\_pos[1]= col;

returnelement\_pos;

}

elseif(nums[row][col]>search\_element){

returnSaddleback(nums, row -1, col,search\_element);

}

returnSaddleback(nums, row, col +1,search\_element);

}

}

**Output:**



2. Find the smallest positive integer value that cannot be represented as sum of any subset of a given array sorted in ascending order.

#include <stdio.h>

int findSmallest(int arr[], int n)

{

int res = 1;

for (int i = 0; i < n && arr[i] <= res; i++)

res = res + arr[i];

return res;

}

int main()

{

int arr1[] = {1, 3, 4, 5};

int n1 = sizeof(arr1)/sizeof(arr1[0]);

printf("output1: %d\n", findSmallest(arr1, n1));

int arr2[] = {1, 2, 6, 10, 11, 15};

int n2 = sizeof(arr2)/sizeof(arr2[0]);

printf("output2: %d\n", findSmallest(arr2, n1));

int arr3[] = {1, 1, 1, 1};

int n3 = sizeof(arr3)/sizeof(arr3[0]);

printf("output3: %d\n", findSmallest(arr3, n1));

int arr4[] = {1, 1, 3, 4};

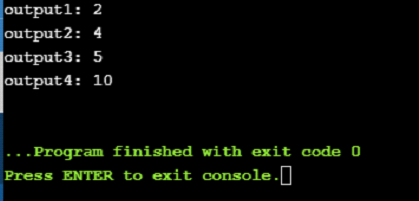
int n4 = sizeof(arr4)/sizeof(arr4[0]);

printf("output4: %d\n", findSmallest(arr4, n1));

return 0;

}

**Output:**



**3. C program for triply linked list**

#include<stdio.h>

struct SLL;

struct TLL {

struct TLL \*top;

struct TLL \*bottom;

struct SLL \*next;

};

typedef struct TLL tnode;

typedef struct SLL {

char ch;

struct SLL \*link;

};

typedef struct SLL snode;

snode \*newnode, \*ptr, \*prev, \*temp;

snode \*first = NULL, \*last = NULL;

tnode \*newt, \*tlast = NULL, \*ttemp;

//--- TLL node---

tnode\* create\_tnode()

{

newt = (tnode \*)malloc(sizeof(tnode));

if (newt == NULL)

{

printf("\nMemory was not allocated");

return 0;

}

else

{

newt->top = NULL;

newt->bottom = NULL;

newt->next = NULL;

return newt;

}

}

//---SLL---

snode\* create\_node(char c)

{

newnode = (snode \*)malloc(sizeof(snode));

if (newnode == NULL)

{

printf("\nMemory was not allocated");

return 0;

}

else

{

newnode->ch = c;

newnode->link = NULL;

return newnode;

}

}

//--- insert SLL---

void insert\_node\_first(char c)

{

newnode = create\_node(c);

if(tlast->next == NULL)

tlast->next = newnode;

if (first == last && first == NULL)

{

first = last = newnode;

first->link = NULL;

last->link = NULL;

}

else

{

temp = first;

first = newnode;

first->link = temp;

}

printf("\n----INSERTED %c TO SLL----", c);

}

//---insert TLL---

void insert\_Tnode()

{

newt = create\_tnode();

if (tlast == NULL)

{

tlast = newt;

tlast->next = NULL;

tlast->top = NULL;

tlast->bottom = NULL;

}

else

{

ttemp = tlast;

tlast = newt;

tlast->next = NULL;

tlast->top = ttemp;

tlast->bottom = NULL;

ttemp->bottom = tlast;

}

printf("\n----CREATED NEW TLL----");

}

void main()

{

char s[100], n;

int i;

scanf("%[^;]s",s);

insert\_Tnode();

for(i = 0; s[i] != '\0'; i++)

{

n = s[i];

if(n == '\n')

insert\_Tnode();

else

insert\_node\_first(n);

}

printf("\n%s\n",s);

}