DAILY ONLINE ACTIVITIES SUMMARY

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Date:** | **11/6/2020** | | | | **Name:** | **Sushmitha Shet** | |
| **Sem & Sec** | **8 B** | | | | **USN:** | **4al16cs110** | |
| Online Test Summary | | | | | | | |
| **Subject** | | **SMS** | | | | | |
| **Max. Marks** | | **60** | | **Score** | | **60** | |
| Certification Course Summary | | | | | | | |
| **Course** | **Introduction to Ethical Hacking** | | | | | | |
| **Certificate Provider** | | | **Great Learning** | **Duration** | | | **6 hrs** |
| Coding Challenges | | | | | | | |
| **Problem Statement: Write a C Program to create random graph using random edge generation.** | | | | | | | |
| **Status:-solved** | | | | | | | |
| **Uploaded the report in Github** | | | | **Yes** | | | |
| **If yes Repository name** | | | | **sushmithashet** | | | |
| **Uploaded the report in slack** | | | | **Yes** | | | |

Online coding:

C program to create random graph using random edge generation.

C Program to Create a Random Graph Using Random Edge Generation

#include<stdio.h>

#include<stdlib.h>

#include <time.h>

#define MAX\_VERTICES 30

#define MAX\_EDGES 10

typedef unsigned char vertex;

int main(){

srand ( time(NULL) );

int numberOfVertices = rand() % MAX\_VERTICES;

srand ( time(NULL) );

int maxNumberOfEdges = rand() % MAX\_EDGES;

if( numberOfVertices == 0)

numberOfVertices++;

vertex \*\*\*graph;

printf("Total Vertices = %d, Max # of Edges = %d\n",numberOfVertices, maxNumberOfEdges);

if ((graph = (vertex \*\*\*) malloc(sizeof(vertex \*\*) \* numberOfVertices)) == NULL){

printf("Could not allocate memory for graph\n");

exit(1);

}

int vertexCounter = 0;

int edgeCounter = 0;

for (vertexCounter = 0; vertexCounter < numberOfVertices; vertexCounter++){

if ((graph[vertexCounter] = (vertex \*\*) malloc(sizeof(vertex \*) \* maxNumberOfEdges)) == NULL){

printf("Could not allocate memory for edges\n");

exit(1);

}

for (edgeCounter = 0; edgeCounter < maxNumberOfEdges; edgeCounter++){

if ((graph[vertexCounter][edgeCounter] = (vertex \*) malloc(sizeof(vertex))) == NULL){

printf("Could not allocate memory for vertex\n");

exit(1);

}

}

}

vertexCounter = 0;edgeCounter = 0;

for (vertexCounter = 0; vertexCounter < numberOfVertices; vertexCounter++){

printf("%d:\t",vertexCounter);

for (edgeCounter=0; edgeCounter < maxNumberOfEdges; edgeCounter++){

if (rand()%2 == 1){ /\*link the vertices\*/

int linkedVertex = rand() % numberOfVertices;

graph[vertexCounter][edgeCounter] = graph[linkedVertex];

printf("%d, ", linkedVertex);

}

else{

graph[vertexCounter][edgeCounter] = NULL;

}

}

printf("\n");

}

return 1;

}