**DAILY ONLINE ACTIVITIES SUMMARY**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Date:** | **15-06-2020** | | | | | **Name:** | **Deril Quadras** | |
| **Sem & Sec** | **8 A** | | | | | **USN:** | **4AL16CS031** | |
| **Online Test Summary** | | | | | | | | |
| **Subject** | | **SMS** | | | | | | |
| **Max. Marks** | | **60** | | **Score** | | | **-** | |
| **Certification Course Summary** | | | | | | | | |
| **Course** | **IoT Foundation:Command and Control** | | | | | | | |
| **Certificate Provider** | | | **AWS** | | **Duration** | | | **120mins** |
| **Coding Challenges** | | | | | | | | |
| **Problem Statement:**   1. Write a C Program to perform the following operations on Triply Linked List. | | | | | | | | |
| **Status: Solved** | | | | | | | | |
| **Uploaded the report in Github** | | | | | **Yes** | | | |
| **If yes Repository name** | | | | | **Derilquadras/online\_c\_coding\_repository** | | | |
| **Uploaded the report in slack** | | | | | **Yes** | | | |

Online Test Details:

Marks not yet received in mail.

Certification Course Details:



Coding Challenges Details:

**PROGRAM 1 .**

**//Write a C Program to perform the following operations on 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);**

**}**