**DAILY ONLINE ACTIVITIES SUMMARY**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Date:** | **08-06-2020** | | | | | **Name:** | **Huda Sultana** | |
| **Sem & Sec** | **8 A** | | | | | **USN:** | **4AL16CS039** | |
| **Online Test Summary** | | | | | | | | |
| **Subject** | | **SMS** | | | | | | |
| **Max. Marks** | | **60** | | **Score** | | | **60** | |
| **Certification Course Summary** | | | | | | | | |
| **Course** | **Introdution to Amazon CloudFront** | | | | | | | |
| **Certificate Provider** | | | **AWS** | | **Duration** | | | **10 mins** |
| **Coding Challenges** | | | | | | | | |
| **Problem Statement:**  Program to partition an integer into numbers which sum up to form the original number. It generates partitions of a set of numbers for a given range. | | | | | | | | |
| **Status: Solved** | | | | | | | | |
| **Uploaded the report in Github** | | | | | **Yes** | | | |
| **If yes Repository name** | | | | | **Hudasulltana/online\_coding** | | | |
| **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)

**PROGRAM 1 .**

**//** Program to partition an integer into numbers which sum up to form the original number. It generates partitions of a set of numbers for a given range.

**#include <stdio.h>**

**#include <stdlib.h>**

**typedef struct {**

**int first;**

**int n;**

**int level;**

**} Call;**

**void print(int n, int \* a) {**

**int i ;**

**for (i = 0; i <= n; i++) {**

**printf("%d", a[i]);**

**}**

**printf("\n");**

**}**

**void integerPartition(int n, int \* a){**

**int first;**

**int i;**

**int top = 0;**

**int level = 0;**

**Call \* stack = (Call \* ) malloc (sizeof(Call) \* 1000);**

**stack[0].first = -1;**

**stack[0].n = n;**

**stack[0].level = level;**

**while (top >= 0){**

**first = stack[top].first;**

**n = stack[top].n;**

**level = stack[top].level;**

**if (n >= 1) {**

**if (first == - 1) {**

**a[level] = n;**

**print(level, a);**

**first = (level == 0) ? 1 : a[level-1];**

**i = first;**

**} else {**

**i = first;**

**i++;**

**}**

**if (i <= n / 2) {**

**a[level] = i;**

**stack[top].first = i;**

**top++;**

**stack[top].first = -1;**

**stack[top].n = n - i;**

**stack[top].level = level + 1;**

**} else {**

**top--;**

**}**

**} else {**

**top --;**

**}**

**}**

**}**

**int main(){**

**int N = 1;**

**int \* a = (int \* ) malloc(sizeof(int) \* N);**

**int i;**

**printf("\nEnter a number N to generate all set partition from 1 to N: ");**

**scanf("%d", &N);**

**for ( i = 1; i <= N; i++)**

**{**

**printf("\nInteger partition for %d is: \n", i);**

**integerPartition (i, a);**

**}**

**return(0);**

**}**