

DAILY ONLINE ACTIVITIES SUMMARY

Date:	12/6/2020	Name:	Vleena Mascarenhas
Sem & Sec	8 th & B	USN:	4AL16CS121
Online Test Summary			
Subject	Big Data Analytics		
Max. Marks	30	Score	24
Certification Course Summary			
Course	Introduction to Information Security		
Certificate Provider	Great learning Academy	Duration	5.5hrs
Coding Challenges			
Problem Statement: 1.C Program to generate all the set partitions of n numbers beginning from 1 and so on.			
Status: Solved			
Uploaded the report in Github		yes	
If yes Repository name		vleena	
Uploaded the report in slack		yes	

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

Test Completed!

You have successfully participated in CSE_BDA_7.

Rate this Test

Your Rating: ★★★★★ ◀ Click to Rate

Results

Analytics



Round 1

Your Score

24 / 30

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

Learning Material



Stanford and Computer Security



Intro to Stanford and Computer
Security Field



Computer Security - Its applications
and its future



Innovations in Cybersecurity -
Quantum Computing



What is the future of cryptography?



4m

Coding Challenges Details: (Attach the snapshot and briefly write the report for the same)

```
#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);

}
```

OUTPUT:

Enter a number N to generate all set partition from 1 to N: 5

Integer partition for 1 is:

1

Integer partition for 2 is:

2

11

Integer partition for 3 is:

3

12

111

Integer partition for 4 is:

4

13

112

1111

22

Integer partition for 5 is:

5

14

113

1112

11111

122

23