Name: Ganath Avinash G.R

CSE – B

CH.SC.U4CSE24118

Week –1(27/11/2025)

1.Write a program to find the sum of first N natural number

CODE:

```c
//Write a program to sum up n terms

#include <stdio.h>

int sum(int n){
  int s=0;
  for(int i=1;i<=n;i++){
      s+=i;
  }
  return s;
}

void main(){
    int n=0;
    scanf("%d",&n);
    printf("sum: %d",sum(n));
}
```

OUTPUT:

```
C:\Users\Ganath Avinash\OneDrive\ドキュメント\Back-end\DAA>gcc 1.c

C:\Users\Ganath Avinash\OneDrive\ドキュメント\Back-end\DAA>a
5
sum: 15
C:\Users\Ganath Avinash\OneDrive\ドキュメント\Back-end\DAA>
```

JUSTIFICATION:

Space Complexity O(1) since the number of variable created is not changing on based on input so it is O(1)

All the variables created does not depend on the input, It will be the same for any input so no change on input.

2.Write a program to find the sum of Square first N natural number

CODE:

```c
//Write a program to sum of sq. up n terms

#include <stdio.h>

int sum(int n){
  int s=0;
  for(int i=1;i<=n;i++){
      s+=(i*i);
  }
  return s;
}

void main(){
    int n=0;
    scanf("%d",&n);
    printf("sum: %d",sum(n));
}
```

OUTPUT:

```
C:\Users\Ganath Avinash\OneDrive\ドキュメント\Back-end\DAA>gcc 2.c

C:\Users\Ganath Avinash\OneDrive\ドキュメント\Back-end\DAA>a
3
sum: 14
C:\Users\Ganath Avinash\OneDrive\ドキュメント\Back-end\DAA>
```

JUSTIFICATION:

Same as the above question, Space Complexity O(1) since the number of variable created is not changing on based on input so it is O(1)

All the variables created does not depend on the input

It will be the same for any input so no change on input.

3.Write a program to find the sum of Cube first N natural number

CODE:

```c
//Write a program to sum of cube up n terms

#include <stdio.h>

int sum(int n){
  int s=0;
  for(int i=1;i<=n;i++){
      s+=(i*i*i);
  }
  return s;
}

void main(){
    int n=0;
    scanf("%d",&n);
    printf("sum: %d",sum(n));
}
```

OUTPUT:

```
C:\Users\Ganath Avinash\OneDrive\ドキュメント\Back-end\DAA>gcc 3.c

C:\Users\Ganath Avinash\OneDrive\ドキュメント\Back-end\DAA>a
3
sum: 36
C:\Users\Ganath Avinash\OneDrive\ドキュメント\Back-end\DAA>
```

JUSTIFICATION:

Space Complexity O(1) since the number of variable created is not changing on based on input so it is O(1)

All the variables created does not depend on the input it will be the same for any input so no change on input. (Same reason as qns above)

4.Write a program to find factorial using recursion

CODE:

```c
//Write a program for factorial of a number

#include <stdio.h>

int fact(int n){
  if(n==1){
    return 1;
  }
  return n*fact(n-1);
}

void main(){
    int n=0;
    scanf("%d",&n);
    printf("Fact: %d",fact(n));
}
```

OUTPUT:

```
C:\Users\Ganath Avinash\OneDrive\ドキュメント\Back-end\DAA>gcc 4.c

C:\Users\Ganath Avinash\OneDrive\ドキュメント\Back-end\DAA>a
5
Fact: 120
C:\Users\Ganath Avinash\OneDrive\ドキュメント\Back-end\DAA>
```

JUSTIFICATION:

Space Complexity O(n) Since it is recursively calling the function for n times so the variable n will be called n times.

In this example, above it creates 5 instances of the same variable during recursion and our input was 5 (this proves complexity analysis)

5.Write a program to find transpose of a 3*3 Matrix

CODE:

```c
//Transpose
#include <stdio.h>

void main(){
  int arr[3][3];
  for(int i=0;i<3;i++){
    for(int j=0;j<3;j++){
      scanf("%d",&arr[i][j]);
    }
  }
  for(int i=0;i<3;i++){
    for(int j=0;j<3;j++){
      if(i<j){
        int tem=arr[i][j];
        arr[i][j]=arr[j][i];
        arr[j][i]=tem;
      }
    }
  }
  printf("Transpose: \n");
  for(int i=0;i<3;i++){
    for(int j=0;j<3;j++){
      printf(" %d ",arr[i][j]);
    }
    printf("\n");
  }
}
```

OUTPUT:

```
C:\Users\Ganath Avinash\OneDrive\ドキュメント\Back-end\DAA>gcc 5.c

C:\Users\Ganath Avinash\OneDrive\ドキュメント\Back-end\DAA>a
1 2 3
4 5 6
7 8 9
Transpose:
 1  4  7
 2  5  8
 3  6  9
```

JUSTIFICATION:

Space Complexity O(1) since the number of variable created is not changing on based on input so it is O(1)....if it was mentioned user defined dimensions then it would be row x column

Suppose the matrix's dimensions were not fixed and where arr[m][n] then the complexity would have been O(n*m).

6.Write a program to find Fibonacci series

CODE:

```c
//fibbonacci
#include <stdio.h>

int fibbo(int n){
  int s=0;int s1=1;int num=0;
  for(int i=0;i<n-2;i++){
      num=s+s1;
      s=s1;
      s1=num;
  }
  return num;
}

void main(){
    int n=0;
    scanf("%d",&n);
    printf("Fact: %d\n",fibbo(n));
}
```

OUTPUT:

```
C:\Users\Ganath Avinash\OneDrive\ドキュメント\Back-end\DAA>gcc 6.c

C:\Users\Ganath Avinash\OneDrive\ドキュメント\Back-end\DAA>a
5
Fact: 3
```

JUSTIFICATION:

Space Complexity O(1) since the number of variable created is not changing on based on input so it is O(1). All the variables created does not depend on the input it will be the same for any input so no change on input.

If it was recursive function then it would have O(N) as it is iterative it is O(1).