DAA BASIC SPACE COMPLEXITY

NAME: U VENKATA ACHYUTH KRISHNA

ROLL NO: CH.SC.U4CSE24148

CLASS: CSE-B

1. Write a program to find sum of first n natural numbers using user defined function.

Code:

```c
#include <stdio.h>
int sumOfNaturals(int n) {
int sum = (n * (n + 1) / 2);
return sum;
}
int main() {
int n;
printf("Enter a number: ");
scanf("%d", &n);
printf("Sum of first %d natural numbers is: %d\n", n, sumOfNaturals(n));
return 0;
}
```

Output:

```
amma@amma:~$ gedit sumOfNaturals.c
amma@amma:~$ gcc -o sumOfNaturals.c
gcc: fatal error: no input files
compilation terminated.
amma@amma:~$ gcc -o sumOfNaturals sumOfNaturals.c
amma@amma:~$ ./sumOfNaturals
Enter a number: 5
Sum of first 5 natural numbers is: 15
```

Space complexity:

The space complexity for this code is O(1).

Justification:

Space complexity is O(1) because the user-defined function uses only a constant amount of memory for its parameter and one local variable.

2. Write a program to find the Sum of Square of first N natural numbers.

CODE:

```c
#include <stdio.h>
int sumOfSquares(int n) {
return (n * (n + 1) * (2 * n + 1)) / 6;
}
int main() {
int n;
printf("Enter a number: ");
scanf("%d", &n);
printf("Sum of squares of first %d natural numbers is: %d\n", n, sumOfSquares(n));
return 0;
}
```

OUTPUT:

```
amma@amma:~$ gedit sumOfSquares.c
amma@amma:~$ gcc -o sumOfSquares sumOfSquares.c
amma@amma:~$ ./sumOfSquares
Enter a number: 5
Sum of squares of first 5 natural numbers is: 55
```

Space complexity:

O(1).

Justification:

Space complexity is O(1) because the user-defined function uses only a constant amount of memory for its parameter and local variables.

3. Write a program to find Sum of Cubes of first N natural numbers.

Code:

```c
#include <stdio.h>
int sumOfCubes(int n) {
int sum = (n * (n + 1) / 2);
return sum * sum;
}
int main() {
int n;
printf("Enter a number: ");
scanf("%d", &n);
printf("Sum of cubes of first %d natural numbers is: %d\n", n, sumOfCubes(n));
return 0;
}
```

Output:

```
amma@amma:~$ gedit sumOfCubes.c
amma@amma:~$ gcc -o sumOfCubes sumOfCubes.c
amma@amma:~$ ./sumOfCubes
Enter a number: 5
Sum of cubes of first 5 natural numbers is: 225
```

Space complexity:

The space complexity for this code is O(1).

Justification:

Space complexity is O(1) because the user-defined function uses only a constant amount of memory for its parameter and one local variable.

4. Write a program to find factorial of a number using recursion.

Code:

```c
#include <stdio.h>
int factorial(int n) {
if (n == 0 || n == 1)
return 1;
else
return n * factorial(n - 1);
}
int main() {
int num;
printf("Enter a number: ");
scanf("%d", &num);
if (num < 0)
printf("Factorial is not defined for negative numbers.\n");
else
printf("Factorial of %d is %d\n", num, factorial(num));
return 0;
}
```

Output:

```
amma@amma:~$ gedit factorial.c
amma@amma:~$ gcc -o factorial factorial.c
amma@amma:~$ ./factorial
Enter a number: 5
Factorial of 5 is 120
```

Space complexity:

O(n).

Justification:

Space complexity is O(n) because each recursive call adds a new stack frame, creating n nested calls for input n.

5.**Writea program to transpose a 3x3 matrix.**

Code:

```c
#include <stdio.h>

int main() {
    int matrix[3][3], transpose[3][3];

    printf("Enter elements of 3x3 matrix:\n");
    for(int i = 0; i < 3; i++) {
        for(int j = 0; j < 3; j++) {
            scanf("%d", &matrix[i][j]);
        }
    }
    for(int i = 0; i < 3; i++) {
        for(int j = 0; j < 3; j++) {
            transpose[j][i] = matrix[i][j];
        }
    }
printf("\nTranspose of the matrix:\n");
    for(int i = 0; i < 3; i++) {
        for(int j = 0; j < 3; j++) {
            printf("%d ", transpose[i][j]);
        }
        printf("\n");
    }

return 0;
}
```

Output:

```
amma@amma:~$ gedit transpose.c
amma@amma:~$ gcc -o transpose transpose.c
amma@amma:~$ ./transpose
Enter elements of 3x3 matrix:
1 2 3 4 5 6 7 8 9

Transpose of the matrix:
1 4 7
2 5 8
3 6 9
```

Space complexity:

O(1).

Justification:

Space complexity is O(1) because a 3×3 matrix uses fixed constant memory regardless of the input values.

6. Write a program to find Fibonacci numbers of a given number.

Code:

```c
#include <stdio.h>
int main(){
int a,b,c,n;
printf("enter the a number n\n");
scanf("%d",&n);
a=0;
b=1;
printf("the fibonacci numbers are \n");
while(a<n){
printf("%d ",a);
 c=a+b;
 a=b;
 b=c;
 }
 }
```

Output:

```
ammamma25:$ gcc -o fibonacci.c
ammamma25:  ./fibonacci
ener the a number n
4

the fibonacci numbers are
0 1 1 2 3
ammaamma25:$
```

Space complexity:

O(1).

Justification:

O(1) only a fixed number of variables (a, b, c, n) are used.