DESIGN AND ANALYSIS OF ALGORITHMS

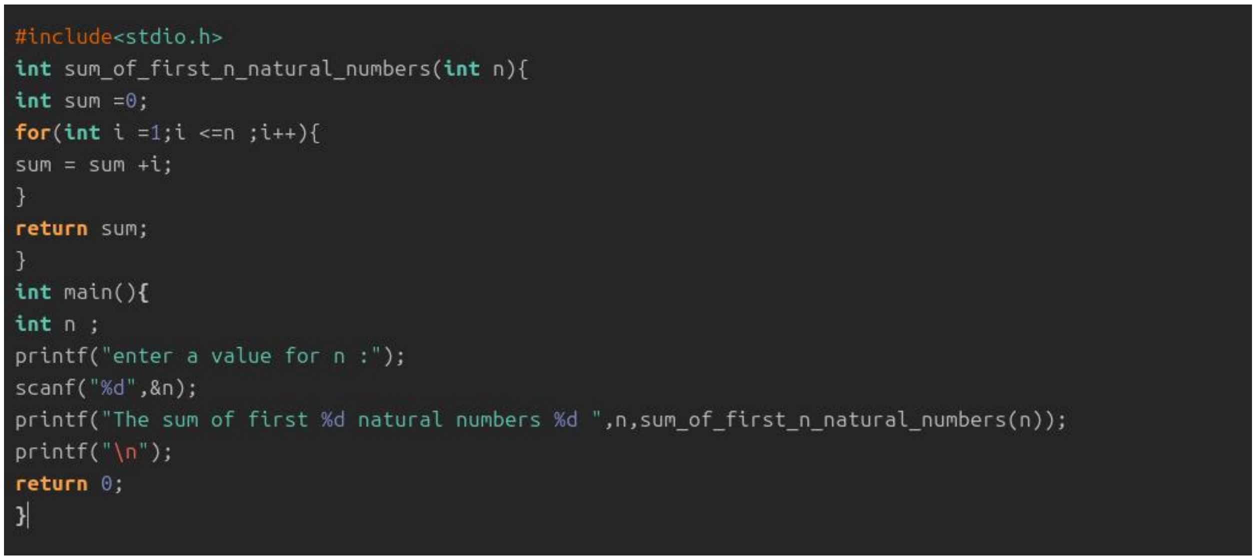
LAB –1

NAME : UPPU MANJU SRI NATH

ROLL NO:CH.SC.U4CSE24249

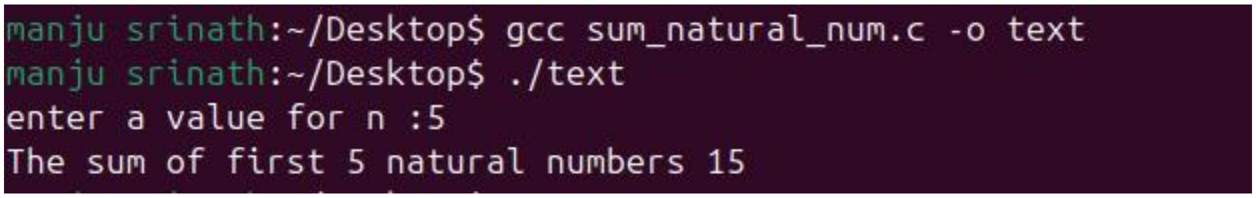
1) WRITE A PROGRAM TO FIND THE SUM OF FIRST N NATURAL NUMBERS ,USING USER DEFINED FUNCTION ?

CODE:



```
#include<stdio.h>
int sum_of_first_n_natural_numbers(int n){
int sum =0;
for(int i =1;i <=n ;i++){
sum = sum +i;
}
return sum;
}
int main(){
int n ;
printf("enter a value for n :");
scanf("%d",&n);
printf("The sum of first %d natural numbers %d ",n,sum_of_first_n_natural_numbers(n));
printf("\n");
return 0;
}
```

OUTPUT :



```
manju srinath:~/Desktop$ gcc sum_natural_num.c -o text
manju srinath:~/Desktop$ ./text
enter a value for n :5
The sum of first 5 natural numbers 15
```

SPACE COMPLEXITY :

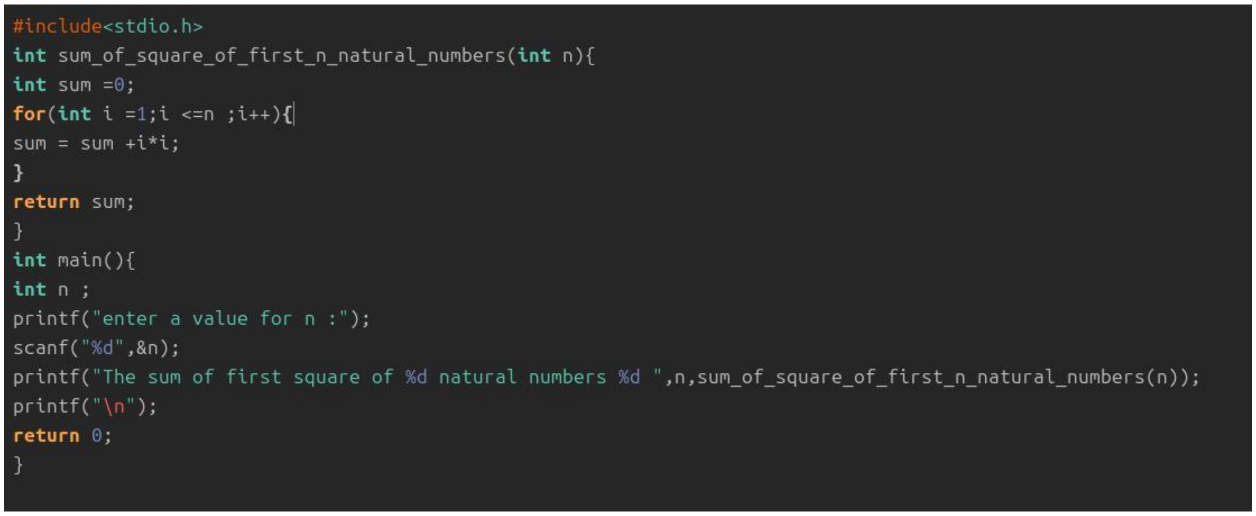
-The loop uses only two variables (s and i) , and it does’t allocate any extra memory as n increases.

-so,the memory usage stays constant.

-therefore,the space complexity is O(1).

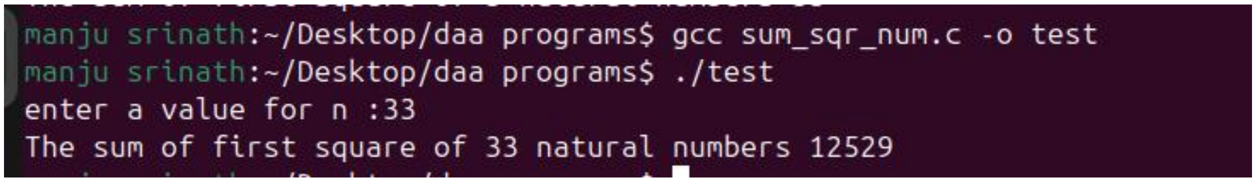
2) WRITE A PROGRAM TO FIND SUM OF SQUARES OF FIRST N NATURAL NUMBERS USING USER DEFINED FUNCTION ?

CODE :



```c
#include<stdio.h>
int sum_of_square_of_first_n_natural_numbers(int n){
int sum =0;
for(int i =1;i <=n ;i++){
sum = sum +i*i;
}
return sum;
}
int main(){
int n ;
printf("enter a value for n :");
scanf("%d",&n);
printf("The sum of first square of %d natural numbers %d ",n,sum_of_square_of_first_n_natural_numbers(n));
printf("\n");
return 0;
}
```

OUTPUT :



```
manju srinath:~/Desktop/daa programs$ gcc sum_sqr_num.c -o test
manju srinath:~/Desktop/daa programs$ ./test
enter a value for n :33
The sum of first square of 33 natural numbers 12529
```

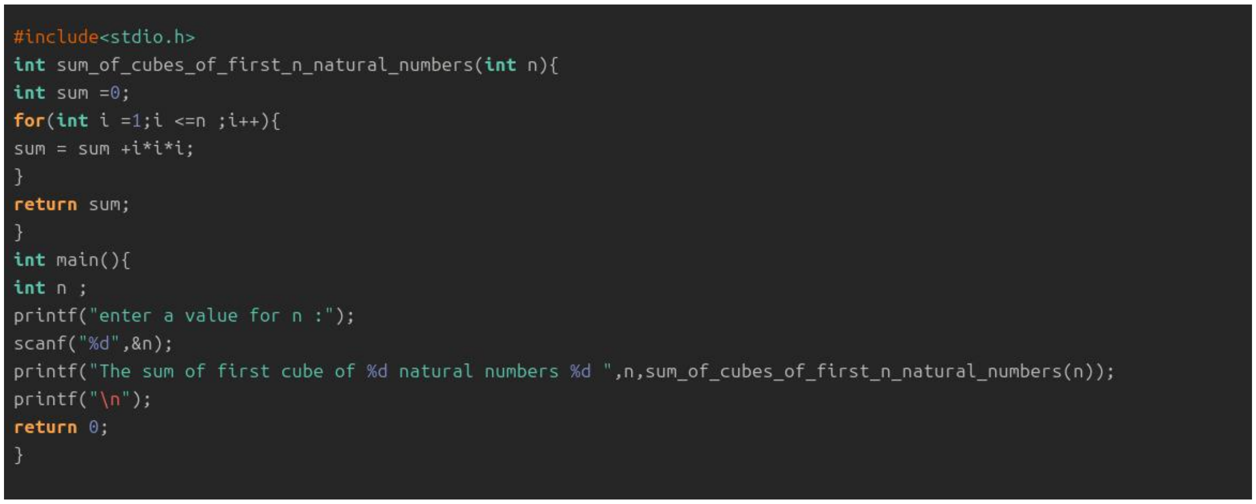
SPACE COMPLEXITY :

- The program uses only a fixed number of variables,and this count does not grow as n increases.

-since the memory usage remains constant ,the space complexity is O(1).

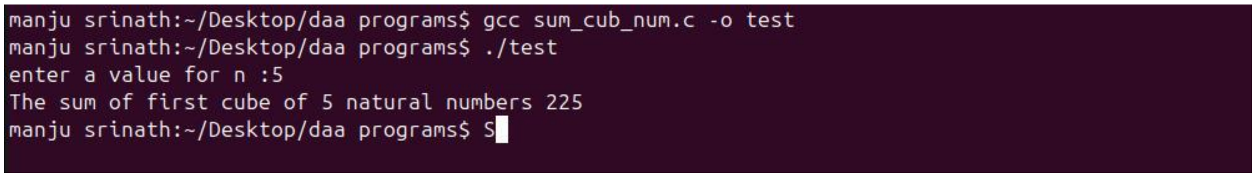
3) WRITE A PROGRAM TO FIND SUM CUBES OF FIRST N NATURAL NUMBERS ?

CODE:



```c
#include<stdio.h>
int sum_of_cubes_of_first_n_natural_numbers(int n){
int sum =0;
for(int i =1;i <=n ;i++){
sum = sum +i*i*i;
}
return sum;
}
int main(){
int n ;
printf("enter a value for n :");
scanf("%d",&n);
printf("The sum of first cube of %d natural numbers %d ",n,sum_of_cubes_of_first_n_natural_numbers(n));
printf("\n");
return 0;
}
```

OUTPUT:



```
manju srinath:~/Desktop/daa programs$ gcc sum_cub_num.c -o test
manju srinath:~/Desktop/daa programs$ ./test
enter a value for n :5
The sum of first cube of 5 natural numbers 225
manju srinath:~/Desktop/daa programs$ S
```

SPACE COMPLEXITY:

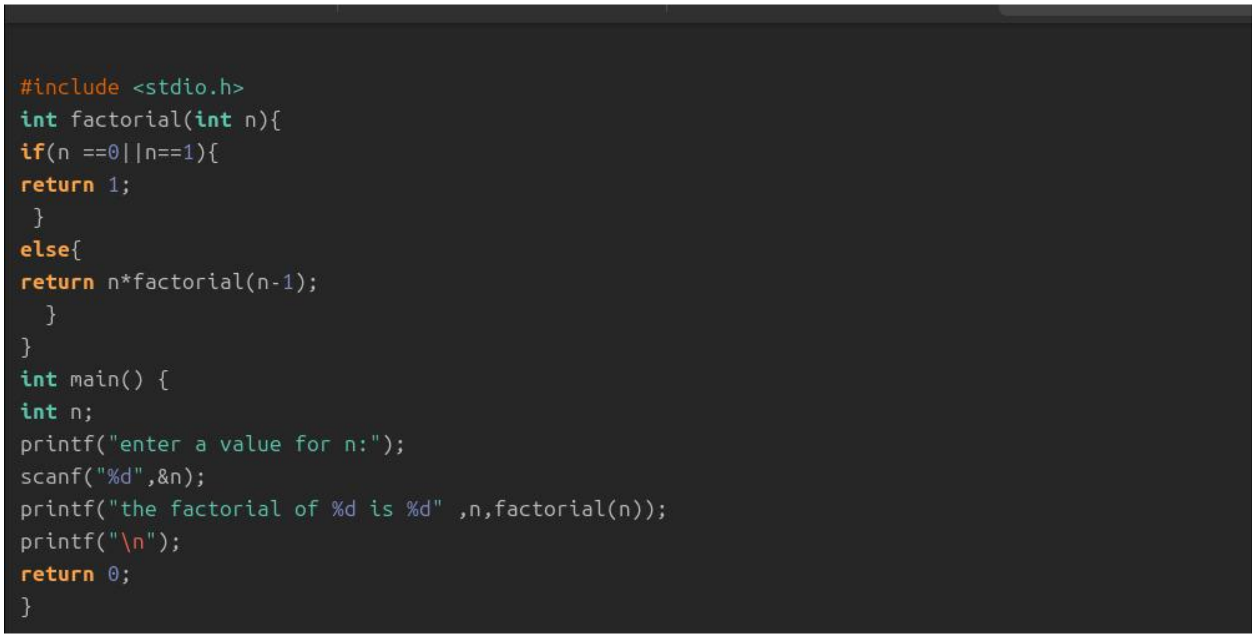
-the loop does not allocate new memory repeatedly

-it only reuses the same variables.

-therefore , the space complexity is O(1).

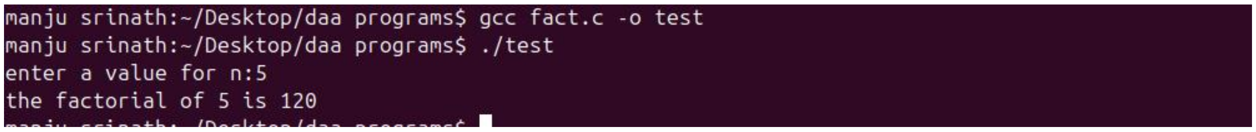
4) WRITE A PROGRAM TO FIND FACTORIAL OF NATURAL NUMBER USING RECURSSIVE FUNCTION ?

CODE:



```
#include <stdio.h>
int factorial(int n){
if(n ==0||n==1){
return 1;
 }
else{
return n*factorial(n-1);
  }
}
int main() {
int n;
printf("enter a value for n:");
scanf("%d",&n);
printf("the factorial of %d is %d" ,n,factorial(n));
printf("\n");
return 0;
}
```

OUTPUT:



```
manju srinath:~/Desktop/daa programs$ gcc fact.c -o test
manju srinath:~/Desktop/daa programs$ ./test
enter a value for n:5
the factorial of 5 is 120
```

SPACE COMPLEXITY :

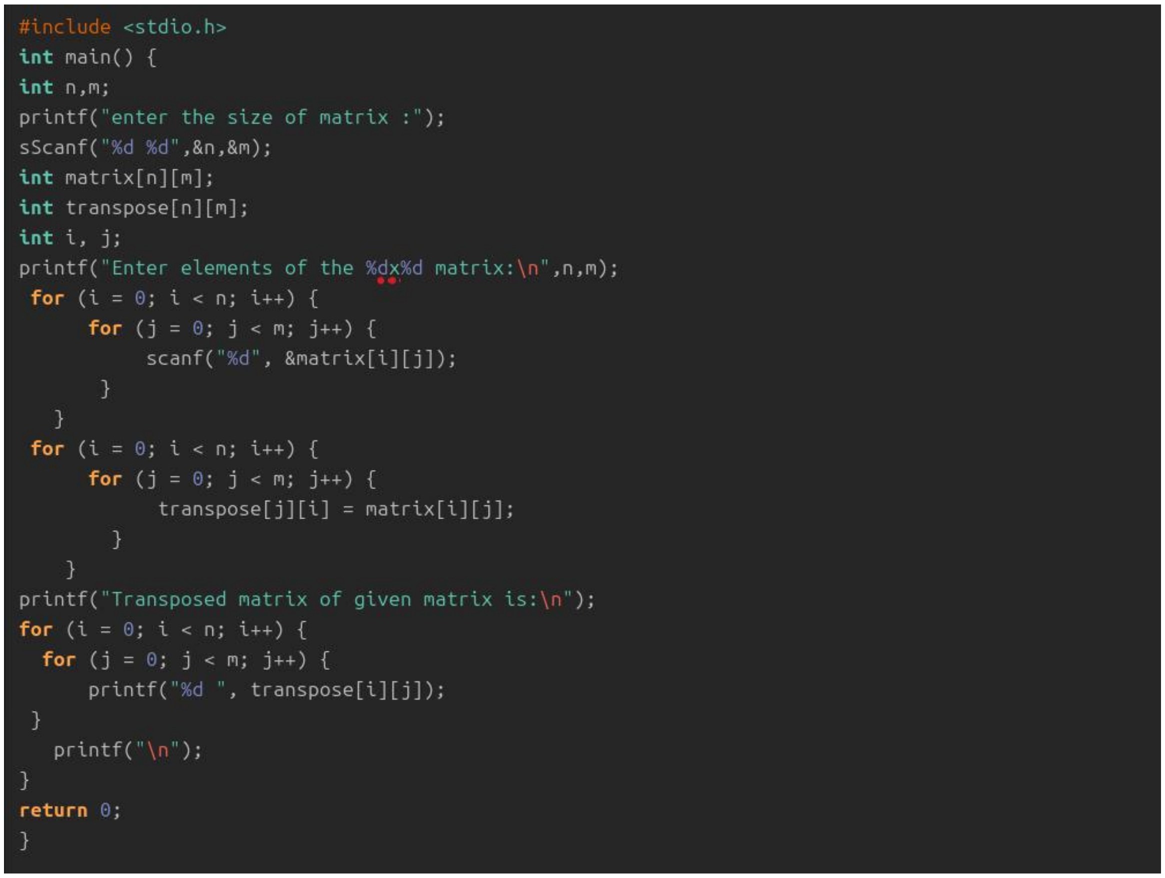
-The function makes multiple recursive calls, and each call adds a new frame to the stack.

-As the number of calls increases with n, the memory used also grows.

-Therefore, the space complexity is O(n).

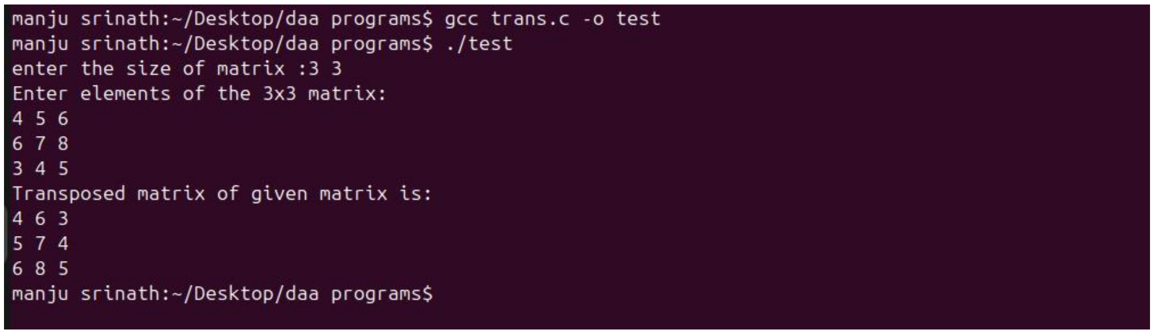
5) WRITE A PROGRAM TO TRANSPOSE A 3X3 MATRIX ?

CODE:



```
#include <stdio.h>
int main() {
int n,m;
printf("enter the size of matrix :");
sScanf("%d %d",&n,&m);
int matrix[n][m];
int transpose[n][m];
int i, j;
printf("Enter elements of the %dx%d matrix:\n",n,m);
 for (i = 0; i < n; i++) {
      for (j = 0; j < m; j++) {
           scanf("%d", &matrix[i][j]);
       }
   }
 for (i = 0; i < n; i++) {
      for (j = 0; j < m; j++) {
            transpose[j][i] = matrix[i][j];
        }
    }
printf("Transposed matrix of given matrix is:\n");
for (i = 0; i < n; i++) {
  for (j = 0; j < m; j++) {
      printf("%d ", transpose[i][j]);
 }
   printf("\n");
}
return 0;
}
```

OUTPUT:



```
manju srinath:~/Desktop/daa programs$ gcc trans.c -o test
manju srinath:~/Desktop/daa programs$ ./test
enter the size of matrix :3 3
Enter elements of the 3x3 matrix:
4 5 6
6 7 8
3 4 5
Transposed matrix of given matrix is:
4 6 3
5 7 4
6 8 5
manju srinath:~/Desktop/daa programs$
```

SPACE COMPLEXITY :

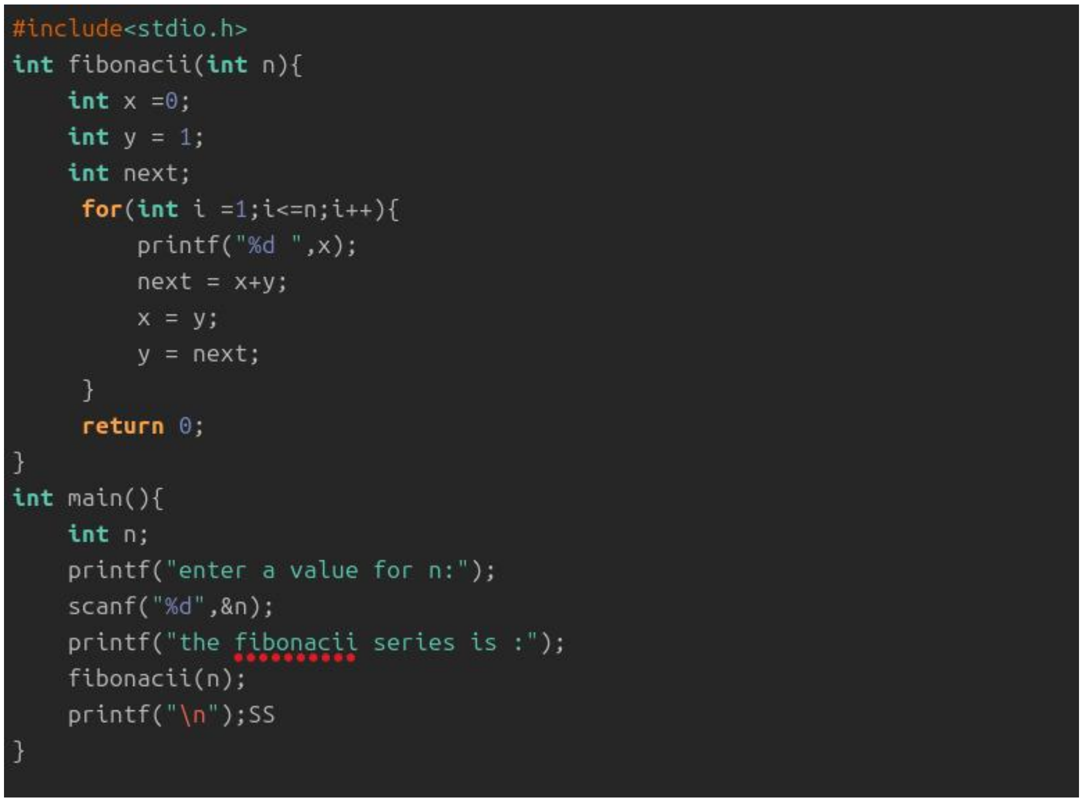
-The program uses a 3×3 array, which has a fixed size that doesn't change with the input.

-Only a few loop variables are used in addition to this.

-Therefore, the space complexity is O(1).

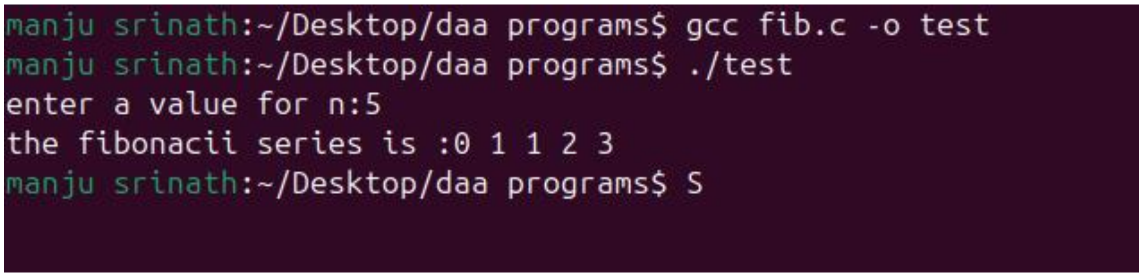
6) WRITE A PROGRAM TO PRINT THE FIBONACCI SERIES UPTO A GIVEN NUMBER USING USER DEFINED FUNCTION (FOR FIRST N NATURAL NUMBERS) ?

CODE :



```
#include<stdio.h>
int fibonacii(int n){
    int x =0;
    int y = 1;
    int next;
     for(int i =1;i<=n;i++){
         printf("%d ",x);
         next = x+y;
         x = y;
         y = next;
     }
     return 0;
}
int main(){
    int n;
    printf("enter a value for n:");
    scanf("%d",&n);
    printf("the fibonacii series is :");
    fibonacii(n);
    printf("\n");SS
}
```

OUTPUT:



```
manju srinath:~/Desktop/daa programs$ gcc fib.c -o test
manju srinath:~/Desktop/daa programs$ ./test
enter a value for n:5
the fibonacii series is :0 1 1 2 3
manju srinath:~/Desktop/daa programs$ S
```

SPACE COMPLEXITY :

-The program uses only a fixed set of variables (a, b, sum, temp), and this number does not change as n increases.

-The loop does not allocate any additional memory.

-Therefore, the space complexity is O(1).