DESIGN AND ANALYSIS OF ALGORITHMS

LAB WORKBOOK

WEEK - 1

NAME : M.RAAHITHYA

ROLL NUMBER : CH.SC.U4CSE24124

CLASS : CSE-B

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

CODE:

```c
#include <stdio.h>
int sumNum(int n){
    int i,sum=0;
    for(i=1;i<=n;i++){
      sum+=i;
      }
      return sum;
      }
int main(){
  int n,result;
 printf("enter a number n:\n");
 scanf("%d",&n);
 result=sumNum(n);
 printf("the sum of %d natural numbers is %d\n",n,result);
 return 0;
 }
```

OUTPUT:

```
enter a number n:
10
the sum of 10 natural numbers is 55
```

Space Complexity: O(1)

Justification:

Space Usage:

main():

n ,sum uses 4+4=8 bytes

sumNum():

i ,sum uses 4 bytes each

Total = 16 bytes = constant

Space Complexity = O(1)

Question 2: Write a program to find sum of squares of the first n natural numbers.

CODE:

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

OUTPUT:

```
enter a number n:
5
the sum squares of 5 natural numbers is 55
```

Space Complexity: O(1)

Justification:

Space Usage:

main():

n, i, sum use 4 + 4 + 4 = 12 bytes

And no other functions

Total = 12 bytes = constant
so, Space Complexity = O(1)

Question 3: Write a program to find sum of cubes of the first n natural numbers.

CODE:

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

OUTPUT:

```
enter a number n:
5
the sum of cubes 5 natural numbers is 225
```

Space Complexity: O(1)

Justification:

Space Usage:

main():

n, i, sum use 4 + 4 + 4 = 12 bytes

And,no other functions

Total = 12 bytes = constant
So,Space Complexity = O(1)

Question 4: Write a program to find the factorial of a given integer 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, result;
    printf("Enter a number: ");
    scanf("%d", &num);
    if (num < 0)
        printf("Factorial does not exist for negative numbers.\n");
    else
    {
        result = factorial(num);
        printf("Factorial of %d is %d\n", num, result);
    }
    return 0;
}
```

OUTPUT:

```
Enter a number: 5
Factorial of 5 is 120
```

Space Complexity: O(n)

Justification:

Space Usage:

main():

`num`, `result` = 8 bytes

factorial():

Each recursive call stores n = 4 bytes

This function is recursive, so for input n, the function calls n times.
Space required for the factorial function is 4n bytes.

Overall Memory = 8 + 4n bytes
Space Complexity = O(n)

Question 5: Write a program for transposing a 3 x 3 matrix.

CODE:

```c
#include <stdio.h>
int main(){
int M[3][3];
printf("enter matrix elements\n");
for(int i=0;i<3;i++){
    for(int j=0;j<3;j++){
     scanf("%d",&M[i][j]);
     }
     }
     printf("the transpose of matrix is\n");
for(int i=0;i<3;i++){
    for(int j=0;j<3;j++){
     printf("%d ",M[j][i]);
    }
    printf("\n");

     }
     return 0;
     }
```

OUTPUT:

```
enter matrix elements
1
2
3
4
5
6
7
8
9
the transpose of matrix is
1 4 7
2 5 8
3 6 9
```

Space Complexity: O(1)

Justification:

Space Usage:

main():

`M[3][3]` → 9 integers → 9 × 4 = 36 bytes

`i`, `j` → 4 + 4 = 8 bytes

Total = 36 + 8 = 44 bytes = constant
Space Complexity = O(1)

Question 6: Write a program to calculate Fibonacci series

CODE:

```c
#include <stdio.h>
int main(){
  int n,i;
  int a=0, b=1, c;
  printf("Enter the number of Fibonacci Numbers: ");
  scanf("%d" , &n);
  if(n<=0){
    printf("Please enter a positive integer");
    return 0;
  }
  printf("Fibonacci Series: ");
  for (i=1; i<=n;i++){
    if (i==1){
      printf("%d ", a);
      continue;
    }
    if (i==2){
      printf("%d ",b);
      continue;
    }
    c = a+b;
    printf("%d ", c);
    a=b;
    b=c;
  }
  printf("\n");
  return 0;
}
```

OUTPUT:

```
ppr276@ubuntu:~/Documents/DAA$ gcc SUM.c -o SUM
ppr276@ubuntu:~/Documents/DAA$ ./SUM
Enter the number of Fibonacci Numbers: 9
Fibonacci Series: 0 1 1 2 3 5 8 13 21
```

Space Complexity: O(1)

Justification:

Space Usage:

main():

n, i, a, b, c → 5 × 4 = 20 bytes

And there are no additional functions

So,Total = 20 bytes = constant
Space Complexity = O(1)