LAB4.

GCD

```
#include <stdio.h>
int gcd(int n1, int n2);
int main()
{
    int n1, n2;
    printf("Enter two positive integers: ");
    scanf("%d %d", &n1, &n2);
    printf("G.C.D of %d and %d is %d.\n", n1, n2, ~~gcd~~
                                                    gcd(n1, n2));
    return 0;
}
int gcd(int n1, int n2)
{
    if(n2 != 0)
      return gcd(n2, n1 % n2);
    else
      return n1;
}
```

expected output:

```
Enter two positive integers: 3  12
G.C.D of 3 and 12 is 3.
```

Factorial of given number using Recursion:

```
#include <stdio.h>
int fact(int n);
int main()
{
    int n;
    printf("Enter any number: ");
    scanf("%d", &n);
    printf("%d! = %d\n", n, fact(n));
    return 0;
}
int fact(int n)
{
    if (n==0)
    return 1;
    return n * fact(n-1);
}
```

Expected output:

```
Enter any number : 5
5! =120
```

Scanned by TapScanner

WAP to implement binary search using recursion.

```
#include <stdio.h>
#include <stdbool.h>
void sort(int n, int a[]);
bool search(int k, int i, int j, int a[]);
int main()
{
    printf("How many numbers do you want to enter\n");
    int n;
    scanf("%d", &n);
    int a[n];
    printf("Enter the elements of the array: \n");
    for(int i=0; i<n; i++)
    scanf("%d", &a[i]);
    sort(n,a);
    printf("Enter the element that you want to search\n");
    int k;
    scanf("%d", &k);
    if(search(k,0, n,a))
    {
    printf("Element found\n");
    return 0;
    }
    printf("Element No found \n");
    return 1;
```

Scanned by TapScanner

```
void sort(int n, int a[])
{
    int swap = 1, k, j = n;

    while (swap != 0)
    {
        swap = 0;
        for(int i = 0 ; i < n-1; i++)
        {
            if (a[i] > a[i+1])
            {
                k = a[i];

                a[i] = a[i+1];

                a[i+1] = k;

                swap++;
            }.

        }

    n--;
    }
}
bool search(int k, int i, int j, int a[])
{
    int mid = (i+j)/2;

    if(i <= j)
    {
```

```
    if (k==a[mid])
    return true;
    else if (a[mid]< k)
    return search(k, mid+1, j, a);
    else
    return search(k, i, mid -1,a);

  }
  return false;

}
```

expected output

```
How many numbers do you want to enter

5

Enter the elements of the array:

5 4 3 2 0

Enter the element that you want to search

2

Element Found
```

WAP to implement tower of hanoi algorithm :

```
#include <stdio.h>
void towers(char a, char c, char b, int n);
int main()
{
    int n;
    char source = 'A';
    char destination = 'C';
    char auxillary = 'B';
    printf("How many disks do you want to enter: ");
    scanf("%d",&n);
   tower(source, destination, auxillary, n);
   return 0;
}
void tower(char a, char c, char b, int n)
{
   if (n==1)
   { printf(" Move disk %d from %c to %c\n", n,a,c);
      return;
   }
   tower(a, b, c, n-1);
   printf("Move disk %d from %c to %c \n", n,a,c)
   tower(b, c, a, n-1);
   return;
}
```

Scanned by TapScanner

expected output :

```
How many disks do you want to enter: 3
Move disk 1 from A to C
Move disk. 2 from A to B
Move disk 1 from C to B
Move disk 3 from A to C
Move disk 1 from B to A
Move disk 2 from B to C
Move disk 1 from A to C
```

7. WAP to implement fibonacci sequence using recursion.

```
#include <stdio.h>
int fib(int n);
int main()
{
  int n;
  printf("Enter the number of terms ~~in~~ to be printed from the fibonacci
                 sequence: ");

  scanf("%d", &n);
  printf("Following is the Fibonacci sequence up to %d
            terms: \n", n);

  for(int i =0; i<=n; i++)
  printf("%d  ", fib(i));
     printf("\n");
  return 0;
}
int fib(int n)
{  if (n<=1)
   return n;
   return fib(n-1) + fib(n-2);
}
```

Expected output

```
Enter the number of terms to be ~~the~~ printed from fibonacci
sequence: 7
Following is the Fibonacci sequence up to 7 terms:
 0  1  1  2  3  5  8  13
```