WEEK-1

Date:18-06-2025

List of programs:

1. Write a C program to find the factorial of given number using a recursive function.
2. Write a C program to find the GCD of two numbers using a recursive function.
3. Write a C program to generate Fibonacci series using a recursive function.
4. Write a C program to demonstrate passing of an array to a C function.
5. Write a C program to implement the Towers of Hanoi problem using recursion.

1. **Aim:** To write a C program to find the factorial of given number using a recursive function.

Program:

```
#include<stdio.h>
int fact(int n)
{
    if(n==1)
        return 1;
    else
        return n*fact(n-1);
}
void main()
{
    int n,factorial;
    printf("Enter a positive integer: ");
    scanf("%d",&n);
    factorial=fact(n);
    printf("Factorial of %d is %d\n",n,factorial);
```

```
}
```

Output:

```
Enter a positive integer: 4
Factorial of 4 is 24
```

2.Aim: To Write a C program to find the GCD of two numbers using a recursive function.

Program:

```
#include<stdio.h>
int gcd(int a,int b)
{
    if(a==0)
        return b;
    else if(b==0)
        return a;
    else if(a<b)
        return gcd(a,b%a);
    else
        return gcd(b,a%b);
}
void main()
{
    int x,y,GCD;
    printf("Enter two integers: ");
    scanf("%d%d",&x,&y);
    GCD=gcd(x,y);
    printf("GCD of %d and %d is %d\n",x,y, GCD);
}
```

Output:

```
Enter two integers: 55
65
GCD of 55 and 65 is 5
```

3.Aim: Write a C program to generate Fibonacci series using a recursive function.

Program:

```
#include<stdio.h>
int fib(int n)
{
    if(n==1)
        return 0;
    else if(n==2)
        return 1;
    else
        return fib(n-1)+fib(n-2);
}
void main()
{
    int num,fibonacci;
    printf("Enter an integer: ");
    scanf("%d",&num);
    fibonacci=fib(num);
    printf("%d\n ",fibonacci);
}
```

Output:

```
Enter an integer: 6
5
```

4.Aim: Write a C program to demonstrate passing of an array to a C function.

Program:

```
#include <stdio.h>
void printArray(int arr[], int size)
{
   printf("Array elements are: ");
   for(int i = 0; i < size; i++)
  {
      printf("%d ", arr[i]);
   }
   printf("\n");
}
int sumArray(int arr[], int size)
{
   int sum = 0;
   for(int i = 0; i < size; i++)
{
      sum += arr[i];
   }
   return sum;
}
int main()
{
   int arr[5] = {10, 20, 30, 40, 50};
   int size = 5;
   printArray(arr, size);
   int sum = sumArray(arr, size);
```

```
    printf("Sum of array elements = %d\n", sum);


 return 0;

}
```

Output:

```
Array elements are: 10 20 30 40 50
Sum of array elements = 150
```

5.Aim:

Write a C program to implement the Towers of Hanoi problem using recursion.

Program:

```
#include<stdio.h>
void TOH(int n,char source,char aux,char dest)
{
    if(n==1)
    {
        printf("Move disk 1 fron %c to %c\n",source,dest);
    }
    else
    {
        TOH(n-1,source,dest,aux);
        printf("Move disk %d fron %c to %c\n",n,source,dest);
        TOH(n-1,aux,source,dest);
    }
}
void main()
{
    int n;
    printf("Enter number of disks: ");
    scanf("%d",&n);
    TOH(n,'A','B','C');
}
```

Output:

```
Enter number of disks: 4
Move disk 1 fron A to B
Move disk 2 fron A to C
Move disk 1 fron B to C
Move disk 3 fron A to B
Move disk 1 fron C to A
Move disk 2 fron C to B
Move disk 1 fron A to B
Move disk 4 fron A to C
Move disk 1 fron B to C
Move disk 2 fron B to A
Move disk 1 fron C to A
Move disk 3 fron B to C
Move disk 1 fron A to B
Move disk 2 fron A to C
Move disk 1 fron B to C
```

Inferences:

- Recursion is a powerful concept where a function calls itself to solve a problem step by step.
- Problems like factorial, GCD, Fibonacci series, and Towers of Hanoi are classic examples suited for recursive solutions.
- Recursive programs require a base case (stopping condition) and a recursive case (function calling itself).
- Factorial and Fibonacci demonstrate recursion in mathematical sequences.
- GCD using recursion highlights the efficiency of Euclid's algorithm.
- Towers of Hanoi illustrates recursion in problem-solving using divide-and-conquer.
- Passing arrays to functions shows that arrays are passed by reference in C, allowing functions to access/modify array elements.
- These programs improve understanding of functions, recursion, parameter passing, and modular programming in C.