EXPERIMENT 8

AIM: WAP to implement factorial, Fibonacci of a given number in PROLOG.

THEORY :

Factorial In Prolog : The factorial of a number N is the product of all positive integers from 1 to N, and is denoted as N!. In Prolog, this is implemented using recursion with a base case factorial(0, 1) which returns 1, since 0! is defined as 1. The recursive rule states that the factorial of N is N multiplied by the factorial of N-1, allowing Prolog to compute the result by breaking down the problem into smaller subproblems until the base case is reached.

Fibonacci in Prolog : The Fibonacci sequence is a series of numbers where each term is the sum of the two preceding ones, starting from 0 and 1. In Prolog, this is defined using two base cases: fibonacci(0, 0) and fibonacci(1, 1), which represent the first two terms of the sequence. For values greater than 1, the recursive rule computes the Fibonacci number by summing the results of fibonacci(N-1) and fibonacci(N-2), building up the sequence until the desired position is reached.

CODE :

Factorial in Prolog :

```
% Base case: factorial of 0 is 1

factorial(0, 1).



% Recursive case: factorial of N is N * factorial of (N-1)

factorial(N, F) :-

    N > 0,

    N1 is N - 1,

    factorial(N1, F1),

    F is N * F1.
```

OUTPUT :

```
?- factorial(5, F).
F = 120.
```

Fibonacci in Prolog :

```
% Base case: Fibonacci of 0 is 0, Fibonacci of 1 is 1
fibonacci(0, 0).
fibonacci(1, 1).

% Recursive case: Fibonacci of N is Fibonacci(N-1) + Fibonacci(N-2)
fibonacci(N, F) :-
    N > 1,
    N1 is N - 1,
    N2 is N - 2,
    fibonacci(N1, F1),
    fibonacci(N2, F2),
    F is F1 + F2.
```

```
?- fibonacci(5, F).
F = 5.
```