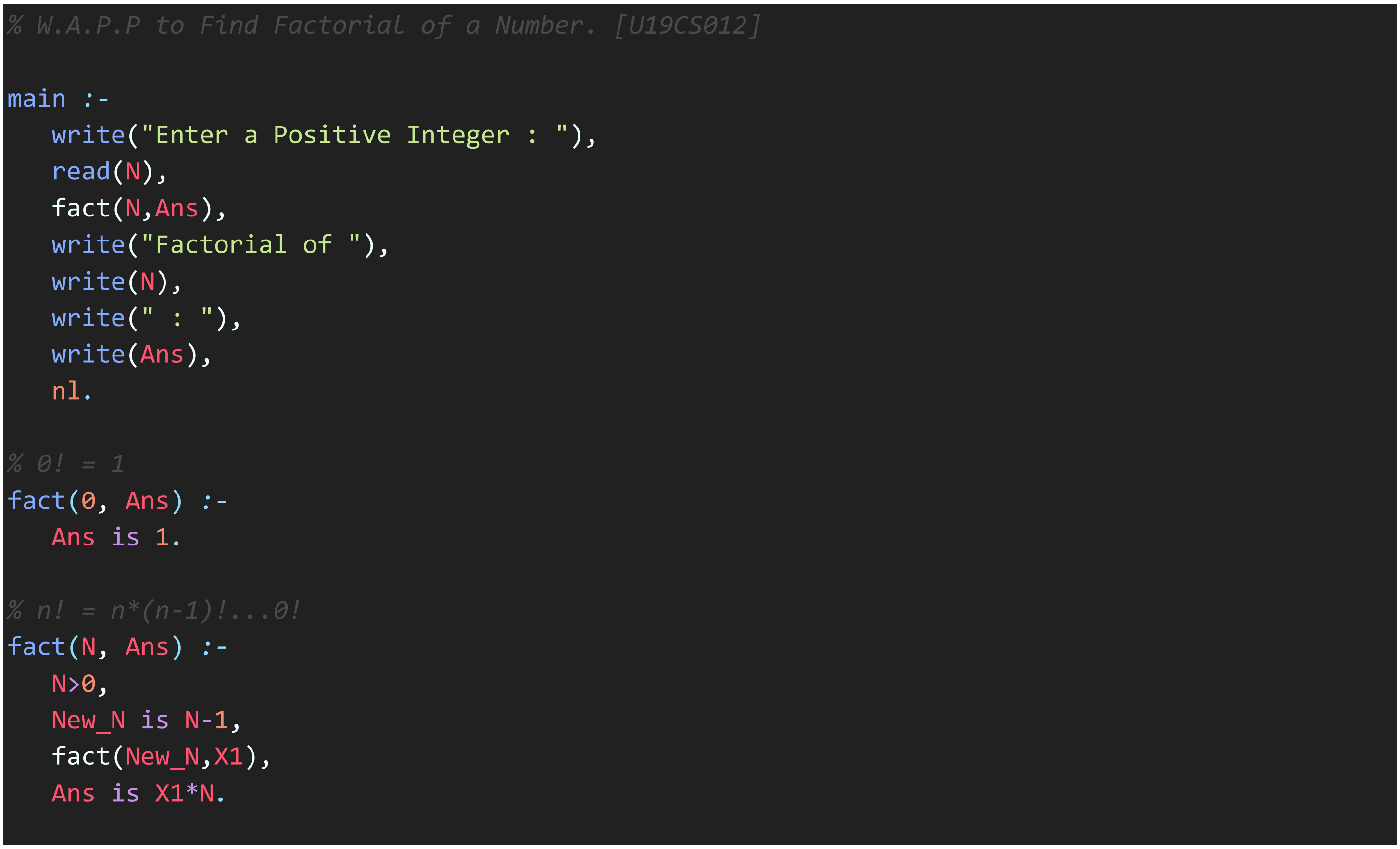
Artificial Intelligence (CS308)

Assignment - 4

U19CS012

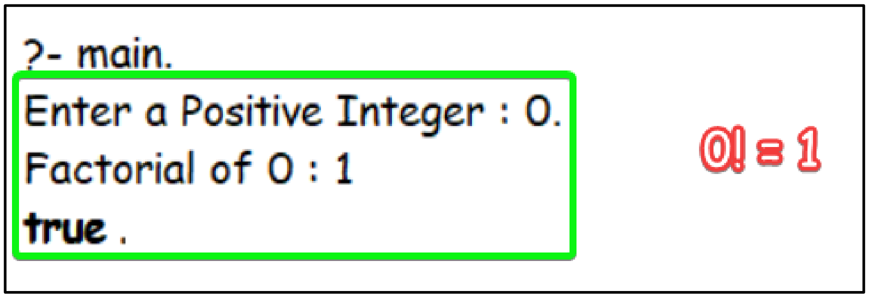
1.) W.A.P.P to Find **Factorial** of a Number. {**W.A.P.P** – Write a Prolog Program}

Prolog Code



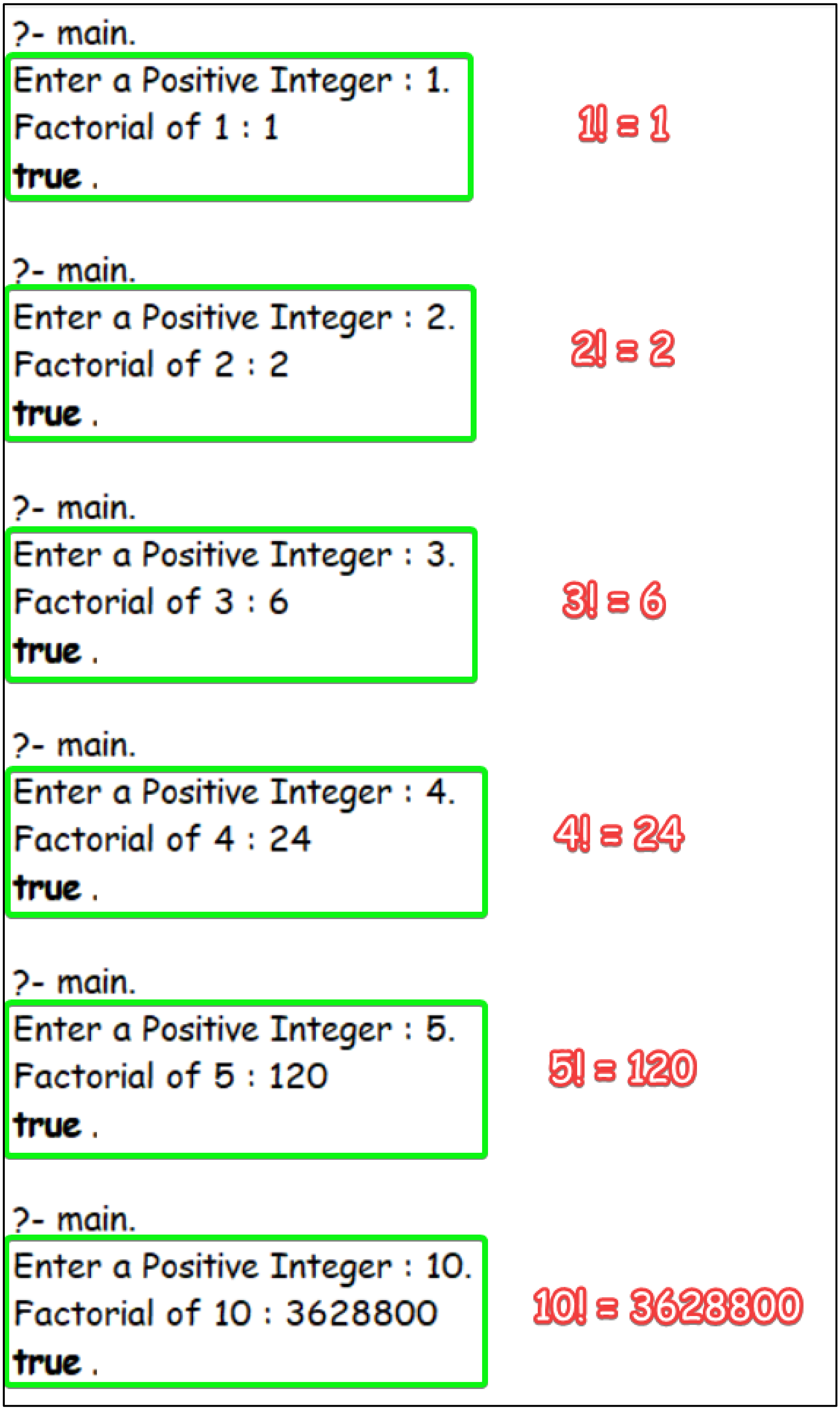
```
% W.A.P.P to Find Factorial of a Number. [U19CS012]

main :-
   write("Enter a Positive Integer : "),
   read(N),
   fact(N,Ans),
   write("Factorial of "),
   write(N),
   write(" : "),
   write(Ans),
   nl.

% 0! = 1
fact(0, Ans) :-
   Ans is 1.

% n! = n*(n-1)!...0!
fact(N, Ans) :-
   N>0,
   New_N is N-1,
   fact(New_N,X1),
   Ans is X1*N.
```

Output



```
?- main.
Enter a Positive Integer : 0.
Factorial of 0 : 1
true .
```

0! = 1



```
?- main.
Enter a Positive Integer : 1.
Factorial of 1 : 1
true .

?- main.
Enter a Positive Integer : 2.
Factorial of 2 : 2
true .

?- main.
Enter a Positive Integer : 3.
Factorial of 3 : 6
true .

?- main.
Enter a Positive Integer : 4.
Factorial of 4 : 24
true .

?- main.
Enter a Positive Integer : 5.
Factorial of 5 : 120
true .

?- main.
Enter a Positive Integer : 10.
Factorial of 10 : 3628800
true .
```

1! = 1

2! = 2

3! = 6

4! = 24

5! = 120

10! = 3628800

2.) W.A.P.P to Print **Fibonacci Series**.

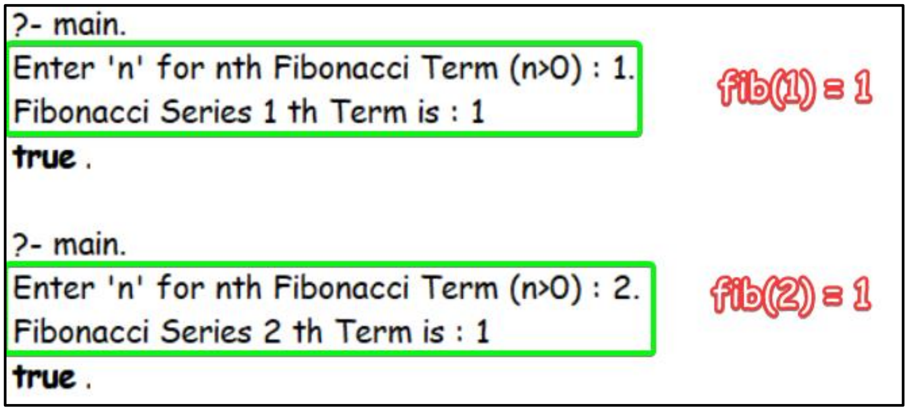
The Fibonacci sequence f (1), f (2), f (3)…is: 1, 1, 2, 3, 5, 8, 13, 21, 34, 55….

```
?- fib (6, R).

R = 8
```

Prolog Code

```
% W.A.P.P to Print Fibonacci Series. [U19CS012]

main:-
        write("Enter 'n' for nth Fibonacci Term (n>0) : "),
        read(N),
        fib(N,X),
        write("Fibonacci Series "),
        write(N),
        write(" th Term"),
        write(" is : "),
        write(X),
        nl.

% fib(1) = 1
fib(1,Ans) :-
        Ans is 1.

% fib(2) = 1
fib(2,Ans) :-
        Ans is 1.

% fib(n) = fib(n-1) + fib(n-2) ... if(n>2)
fib(N,Ans) :-
        N>2,
        Y is N-1,
        Z is N-2,
        fib(Y,X1),
        fib(Z,X2),
        Ans is X1+X2.
```

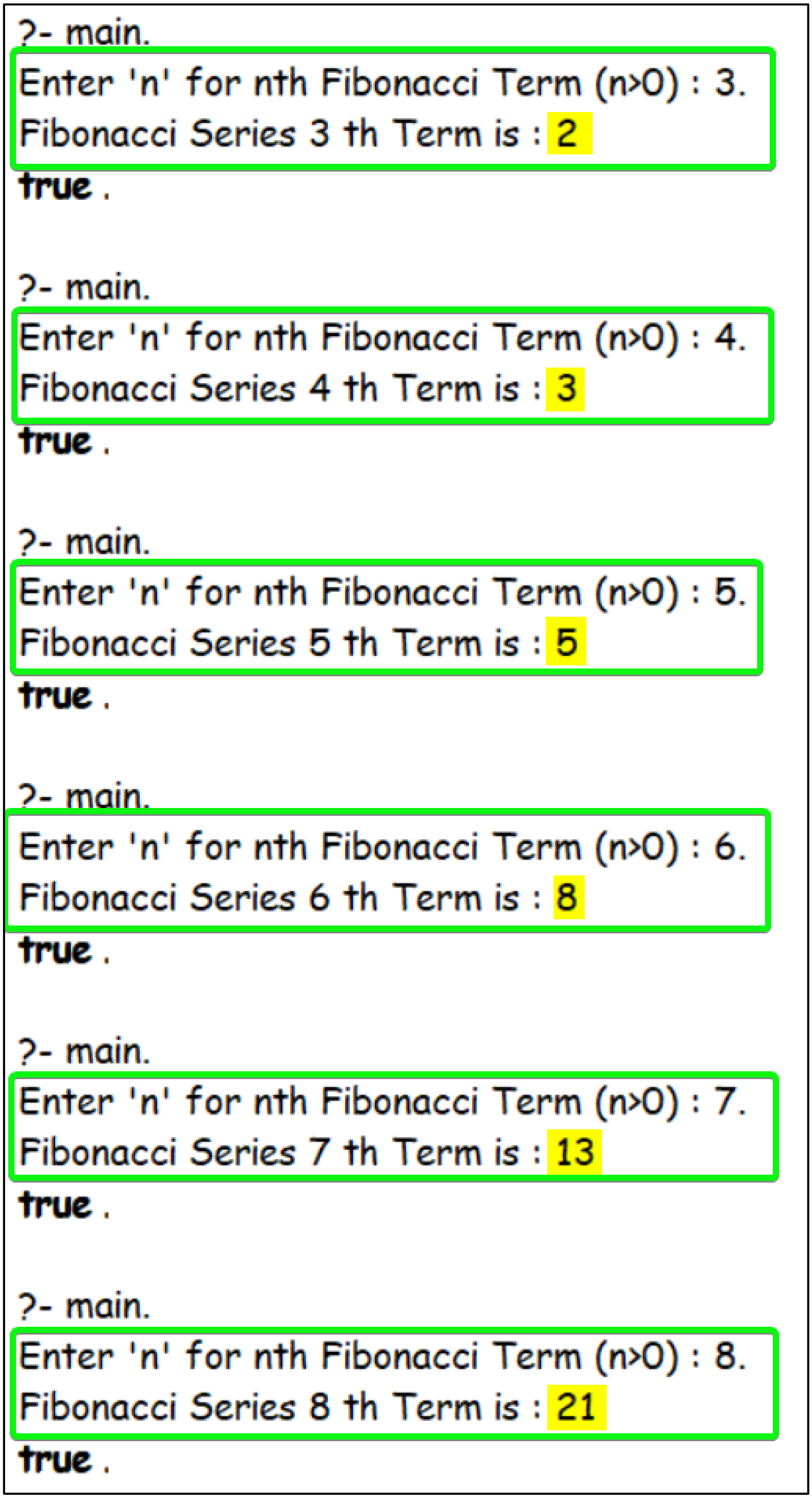
Output



```
?- main.
Enter 'n' for nth Fibonacci Term (n>0) : 1.
Fibonacci Series 1 th Term is : 1
true .

?- main.
Enter 'n' for nth Fibonacci Term (n>0) : 2.
Fibonacci Series 2 th Term is : 1
true .
```

fib(1) = 1

fib(2) = 1



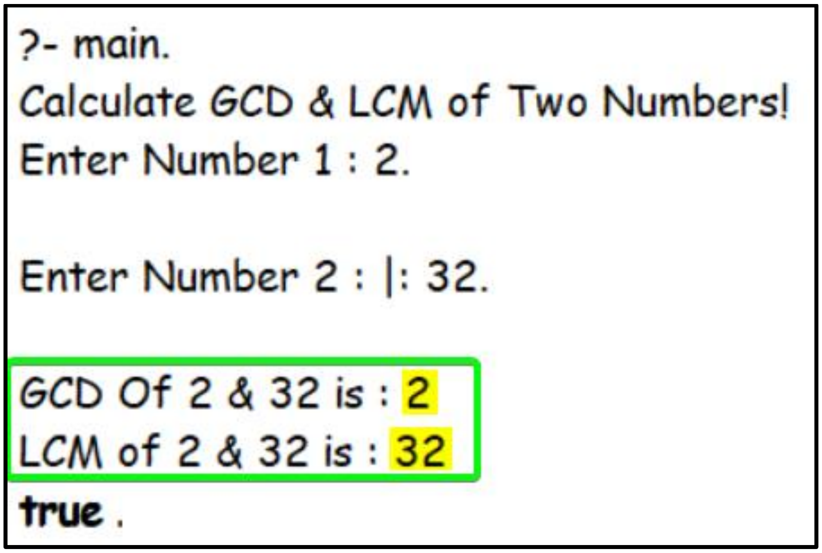
```
?- main.
Enter 'n' for nth Fibonacci Term (n>0) : 3.
Fibonacci Series 3 th Term is : 2
true .

?- main.
Enter 'n' for nth Fibonacci Term (n>0) : 4.
Fibonacci Series 4 th Term is : 3
true .

?- main.
Enter 'n' for nth Fibonacci Term (n>0) : 5.
Fibonacci Series 5 th Term is : 5
true .

?- main.
Enter 'n' for nth Fibonacci Term (n>0) : 6.
Fibonacci Series 6 th Term is : 8
true .

?- main.
Enter 'n' for nth Fibonacci Term (n>0) : 7.
Fibonacci Series 7 th Term is : 13
true .

?- main.
Enter 'n' for nth Fibonacci Term (n>0) : 8.
Fibonacci Series 8 th Term is : 21
true .
```

3.) W.A.P.P to finding the **Greatest Common Divider** (GCD) and **Least Common Multiple** (LCM) of two integers.

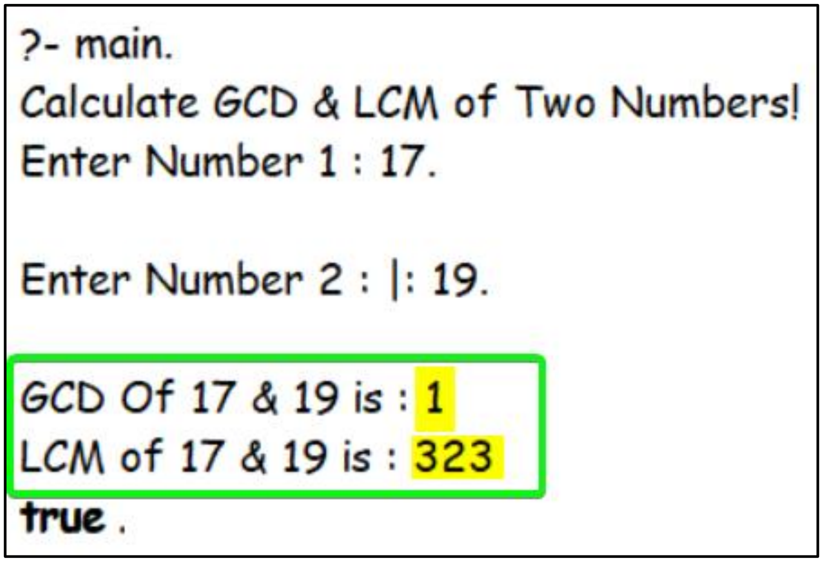
Prolog Code

```
% W.A.P.P to finding the GCD and LCM of two integers. [U19CS012]

% gcd(N,M) * lcm(N,M) = N*M  ... (1)

main :-
    write("Calculate GCD & LCM of Two Numbers!"),
    nl,
    write("Enter Number 1 : "),
    read(N),
    nl,
    write("Enter Number 2 : "),
    read(M),
    nl,
    gcd(N,M,X),
    write("GCD Of "),
    write(N),
    write(" & "),
    write(M),
    write(" is : "),
    write(X),nl,
    Z is N*M,
    Y is Z/X,
    write("LCM of "),
    write(N),
    write(" & "),
    write(M),
    write(" is : "),
    write(Y),
    nl.

% int gcd(int n, int m)
%    return m == 0 ? n : gcd(m, n % m);   ...(2)

% Base Case when M = 0
gcd(N, 0, Ans) :-
    Ans is N.

% gcd(n,m) = gcd(m, n%m)
gcd(N, M, Ans) :-
    M>0,
    Y is mod(N, M),
    gcd(M, Y, Ans).
```

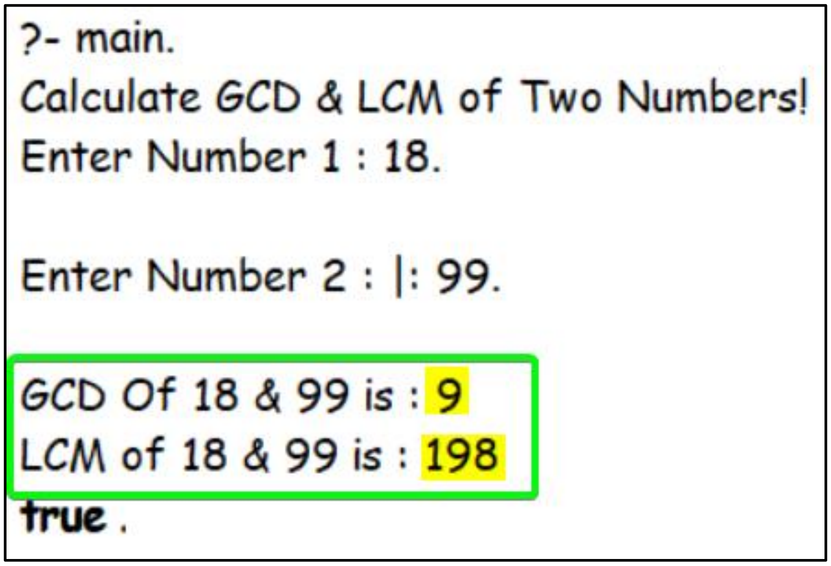
Output



```
?- main.
Calculate GCD & LCM of Two Numbers!
Enter Number 1 : 2.

Enter Number 2 : |: 32.

GCD Of 2 & 32 is : 2
LCM of 2 & 32 is : 32
true .
```



```
?- main.
Calculate GCD & LCM of Two Numbers!
Enter Number 1 : 17.

Enter Number 2 : |: 19.

GCD Of 17 & 19 is : 1
LCM of 17 & 19 is : 323
true .
```

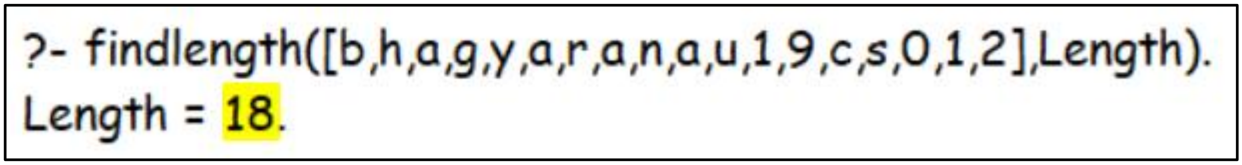


```
?- main.
Calculate GCD & LCM of Two Numbers!
Enter Number 1 : 18.

Enter Number 2 : |: 99.

GCD Of 18 & 99 is : 9
LCM of 18 & 99 is : 198
true .
```

4. W.A.P.P.

A. To find length of the list.

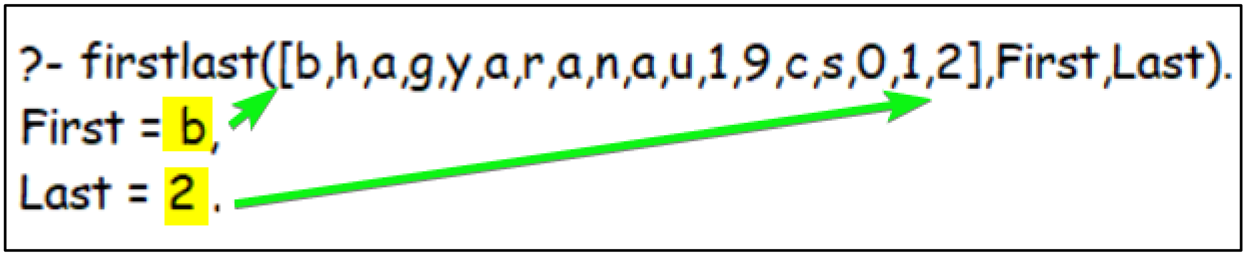
```
findlength([],0).
findlength([_|T], N) :-
    findlength(T,X),
    N is X+1.
```



```
?- findlength([b,h,a,g,y,a,r,a,n,a,u,1,9,c,s,0,1,2],Length).
Length = 18.
```

B. To find first and last element of the list.

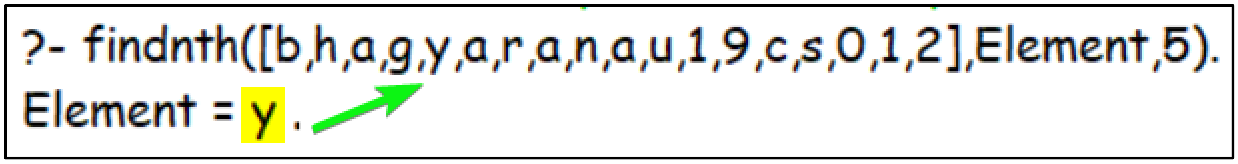
```
firstlast([],[],[]).
firstlast([H],H,H).
firstlast([H|T],H,L) :- firstlast(T,_,L).
```



```
?- firstlast([b,h,a,g,y,a,r,a,n,a,u,1,9,c,s,0,1,2],First,Last).
First = b,
Last = 2 .
```

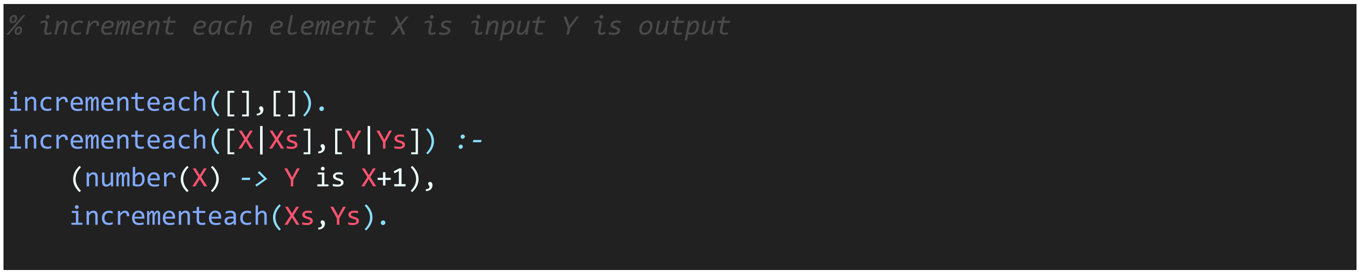
C. To find the nth element of the list.

```
findnth([H|_],H,1).
findnth([_|T],X,N) :-
    N1 is N-1,
    N1 > 0,
    findnth(T,X,N1).
```

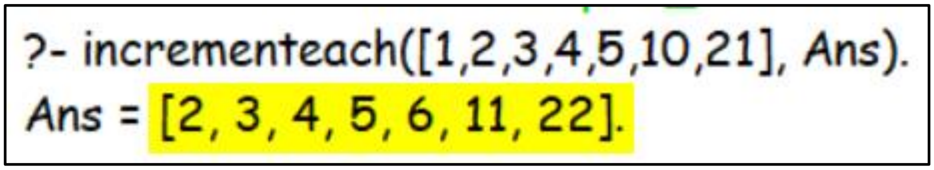


```
?- findnth([b,h,a,g,y,a,r,a,n,a,u,1,9,c,s,0,1,2],Element,5).
Element = y .
```

D. To increment each number in the list.

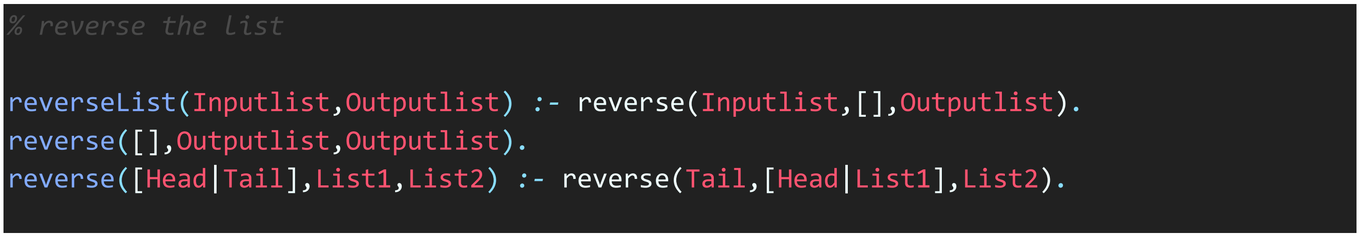


```
% increment each element X is input Y is output

incrementeach([],[]).
incrementeach([X|Xs],[Y|Ys]) :-
    (number(X) -> Y is X+1),
    incrementeach(Xs,Ys).
```

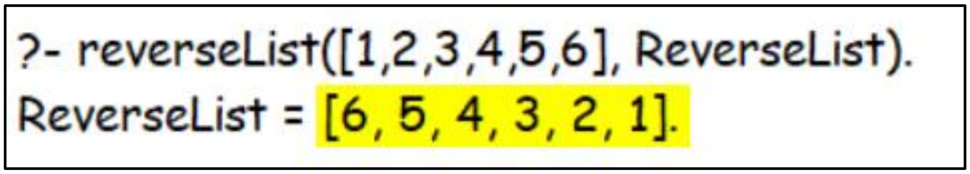


```
?- incrementeach([1,2,3,4,5,10,21], Ans).
Ans = [2, 3, 4, 5, 6, 11, 22].
```

E. To reverse the list.

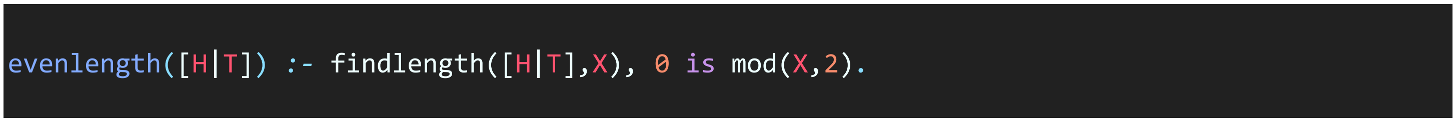


```
% reverse the list

reverseList(Inputlist,Outputlist) :- reverse(Inputlist,[],Outputlist).
reverse([],Outputlist,Outputlist).
reverse([Head|Tail],List1,List2) :- reverse(Tail,[Head|List1],List2).
```

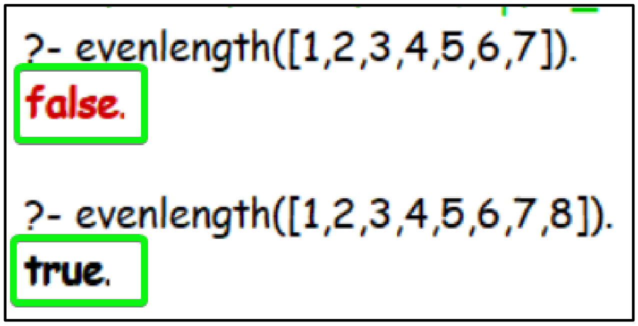


```
?- reverseList([1,2,3,4,5,6], ReverseList).
ReverseList = [6, 5, 4, 3, 2, 1].
```

F. To verify if a list has an even number of elements.

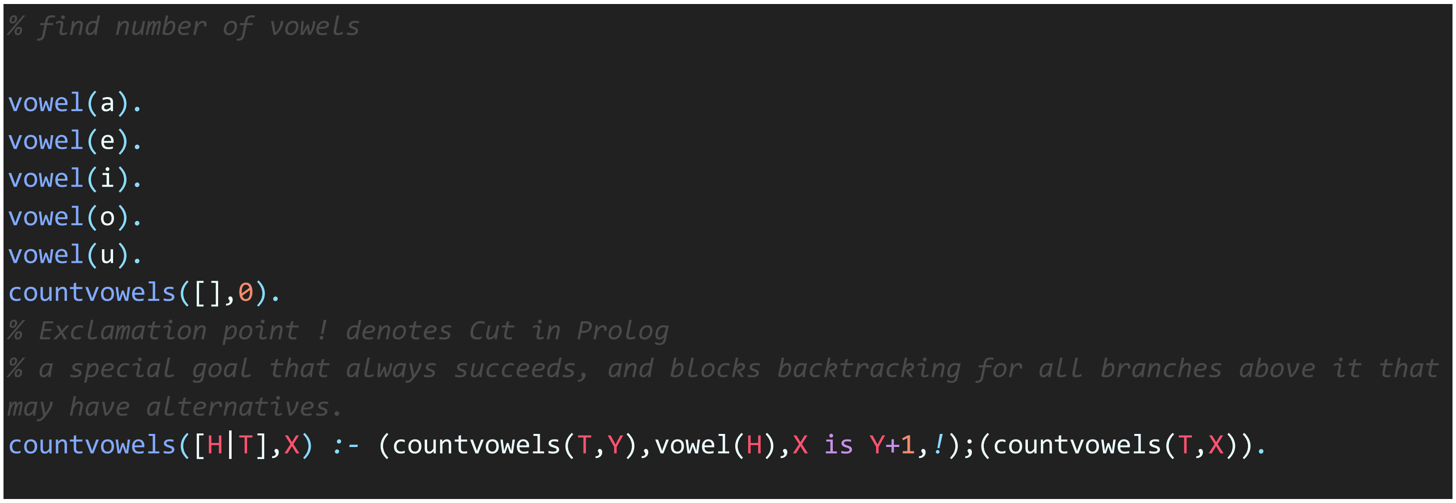


```
evenlength([H|T]) :- findlength([H|T],X), 0 is mod(X,2).
```

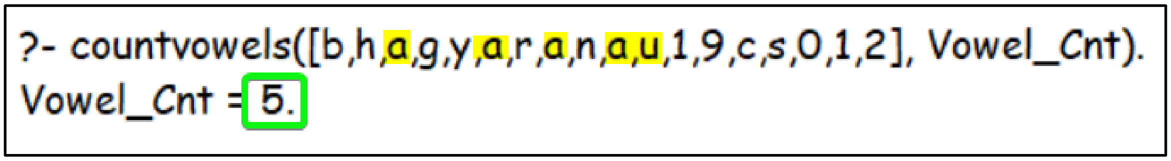


```
?- evenlength([1,2,3,4,5,6,7]).
false.

?- evenlength([1,2,3,4,5,6,7,8]).
true.
```

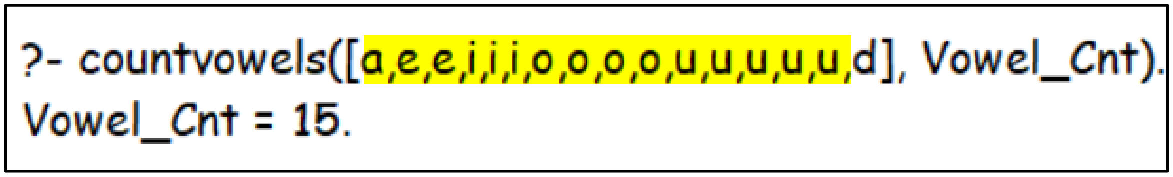
G. To count vowels in the list.



```
% find number of vowels

vowel(a).
vowel(e).
vowel(i).
vowel(o).
vowel(u).
countvowels([],0).
% Exclamation point ! denotes Cut in Prolog
% a special goal that always succeeds, and blocks backtracking for all branches above it that
may have alternatives.
countvowels([H|T],X) :- (countvowels(T,Y),vowel(H),X is Y+1,!);(countvowels(T,X)).
```

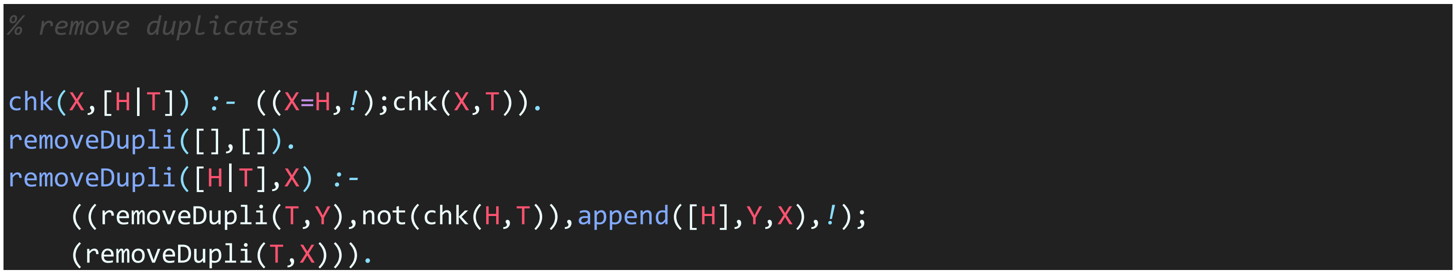


```
?- countvowels([b,h,a,g,y,a,r,a,n,a,u,1,9,c,s,0,1,2], Vowel_Cnt).
Vowel_Cnt = 5.
```

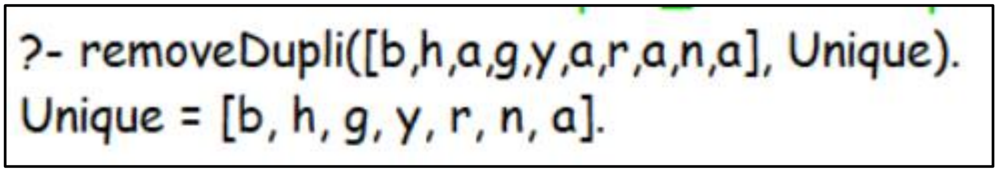


```
?- countvowels([a,e,e,i,i,i,o,o,o,o,u,u,u,u,u,d], Vowel_Cnt).
Vowel_Cnt = 15.
```

H. To remove duplicates from the list.



```
% remove duplicates

chk(X,[H|T]) :- ((X=H,!);chk(X,T)).
removeDupli([],[]).
removeDupli([H|T],X) :-
    ((removeDupli(T,Y),not(chk(H,T)),append([H],Y,X),!);
    (removeDupli(T,X))).
```



```
?- removeDupli([b,h,a,g,y,a,r,a,n,a], Unique).
Unique = [b, h, g, y, r, n, a].
```

SUBMITTED BY: U19CS012

BHAGYA VINOD RANA