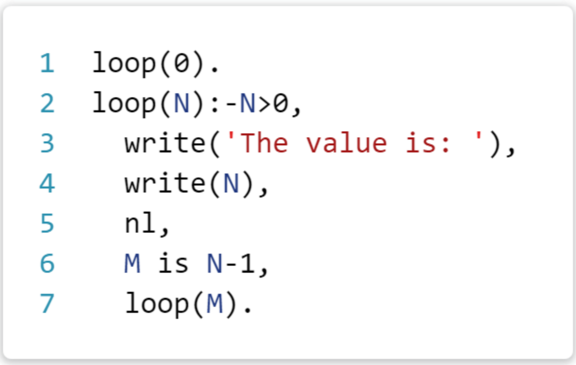
Looping using recursion.

Code:



```prolog
loop(0).
loop(N):-N>0,
  write('The value is: '),
  write(N),
  nl,
  M is N-1,
  loop(M).
```

Output:

```
1 ?- [recursion].
true.

2 ?- loop(2).
The value is: 2
The value is: 1
true ;
false.

3 ?- loop(12).
The value is: 12
The value is: 11
The value is: 10
The value is: 9
The value is: 8
The value is: 7
The value is: 6
The value is: 5
The value is: 4
The value is: 3
The value is: 2
The value is: 1
true ;
false.
```

1. WAP in Prolog to to calculate factorial of a number using recursion

Code:

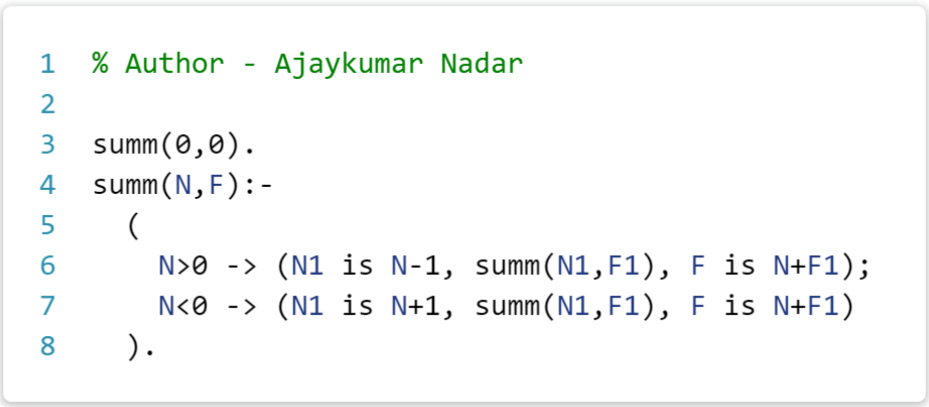
```prolog
% Author - Ajaykumar Nadar

fact(0,1).
fact(N,F):-
  (
    N>0 -> (N1 is N-1, fact(N1,F1), F is N*F1);
    N<0 -> (N1 is N+1, fact(N1,F1), F is N*F1)
  ).
```

Output:

```
Welcome to SWI-Prolog (threaded, 64 bits, version 9.0.4)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

For online help and background, visit https://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

1 ?- [factorial].
true.

2 ?- fact(4,X).
X = 24 .

3 ?- fact(-5,X).
X = -120 .

4 ?- fact(0,X).
X = 1 .
```

2. WAP in Prolog to find the sum of first n natural numbers using recursion

Code:



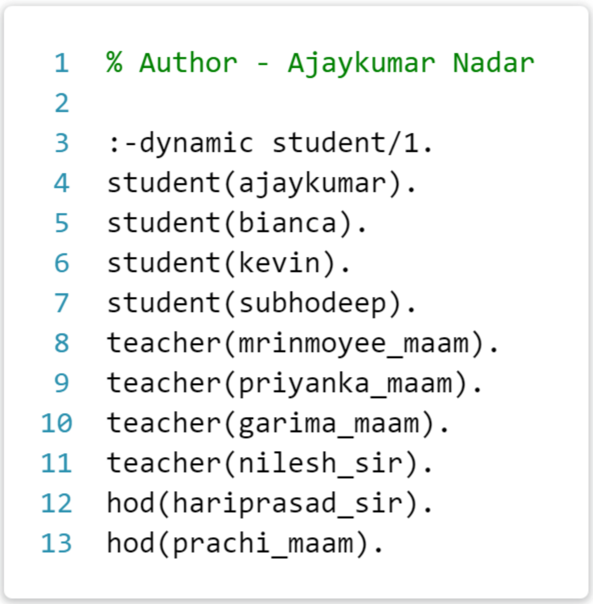
```prolog
% Author - Ajaykumar Nadar

summ(0,0).
summ(N,F):-
  (
    N>0 -> (N1 is N-1, summ(N1,F1), F is N+F1);
    N<0 -> (N1 is N+1, summ(N1,F1), F is N+F1)
  ).
```

Output:

```
Welcome to SWI-Prolog (threaded, 64 bits, version 9.0.4)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

For online help and background, visit https://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

1 ?- [sum].
true.

2 ?- summ(5,X).
X = 15 .

3 ?- summ(-5,X).
X = -15 .
```

3. Create a knowledge base1 to include few facts and rules. On this knowledge base, using database manipulation command- (a) append facts (b) append rules (c) delete facts (d) delete rule (e) append facts at the beginning and end of knowledge base

Knowledge Base:



```
% Author - Ajaykumar Nadar

:-dynamic student/1.
student(ajaykumar).
student(bianca).
student(kevin).
student(subhodeep).
teacher(mrinmoyee_maam).
teacher(priyanka_maam).
teacher(garima_maam).
teacher(nilesh_sir).
hod(hariprasad_sir).
hod(prachi_maam).
```

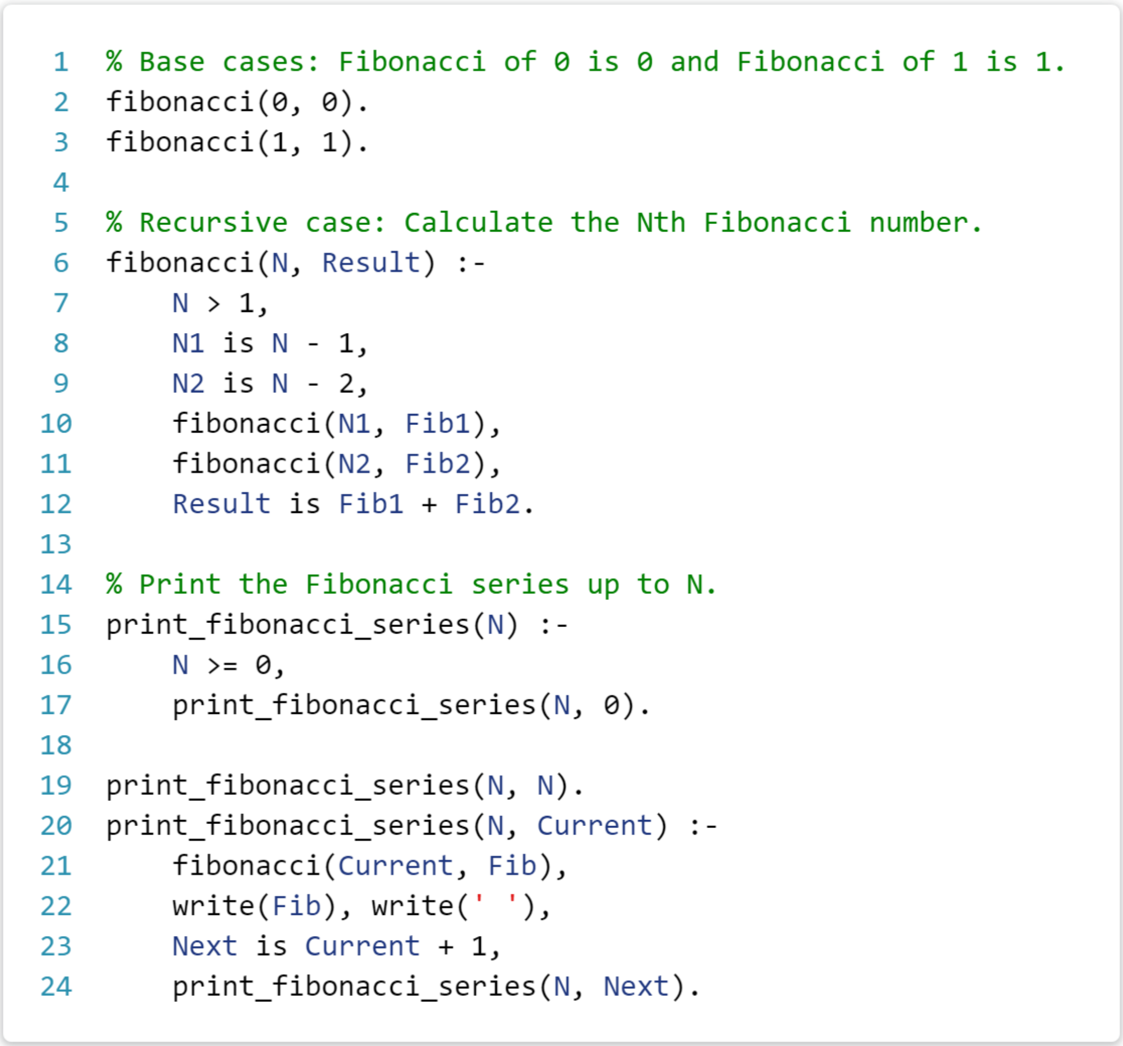
Query & Output:

```
1 ?- [example1].
true.

2 ?- listing.

:- dynamic library_directory/1.
:- multifile library_directory/1.


:- dynamic prolog_load_file/2.
:- multifile prolog_load_file/2.


:- dynamic student/1.

student(ajaykumar).
student(bianca).
student(kevin).
student(subhodeep).

teacher(mrinmoyee_maam).
teacher(priyanka_maam).
teacher(garima_maam).
teacher(nilesh_sir).
```

```
3 ?- assert(student(mokshada)).
true.

4 ?- retract(student(mokshada)).
true.

5 ?- retract(student(lincia)).
false.

6 ?- assertz(student(valerie)).
true.

7 ?- asserta(student(saurabh)).
true.

8 ?- listing.

:- dynamic library_directory/1.
:- multifile library_directory/1.


:- dynamic prolog_load_file/2.
:- multifile prolog_load_file/2.


:- dynamic student/1.

student(saurabh).
student(ajaykumar).
student(bianca).
student(kevin).
student(subhodeep).
student(valerie).

teacher(mrinmoyee_maam).
teacher(priyanka_maam).
teacher(garima_maam).
teacher(nilesh_sir).

:- multifile message_property/2.


:- dynamic expand_answer/2.
:- multifile expand_answer/2.


hod(hariprasad_sir).
hod(prachi_maam).
```

1. Write a program in Prolog to print fibonacci series of number N using recursion

Code:



```prolog
% Base cases: Fibonacci of 0 is 0 and Fibonacci of 1 is 1.
fibonacci(0, 0).
fibonacci(1, 1).

% Recursive case: Calculate the Nth Fibonacci number.
fibonacci(N, Result) :-
    N > 1,
    N1 is N - 1,
    N2 is N - 2,
    fibonacci(N1, Fib1),
    fibonacci(N2, Fib2),
    Result is Fib1 + Fib2.

% Print the Fibonacci series up to N.
print_fibonacci_series(N) :-
    N >= 0,
    print_fibonacci_series(N, 0).

print_fibonacci_series(N, N).
print_fibonacci_series(N, Current) :-
    fibonacci(Current, Fib),
    write(Fib), write(' '),
    Next is Current + 1,
    print_fibonacci_series(N, Next).
```

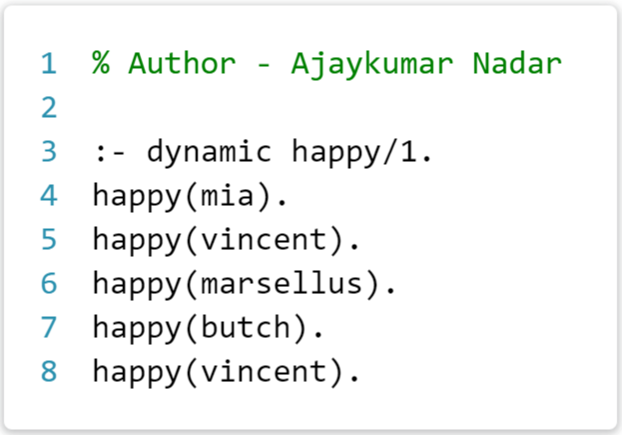
Query & Output:

```
Welcome to SWI-Prolog (threaded, 64 bits, version 9.0.4)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

For online help and background, visit https://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

1 ?- [fibonacci].
true.

2 ?- print_fibonacci_series(10).
0 1 1 2 3 5 8 13 21 34
true .
```

2. Create a knowledge base consisting of the following facts happy(mia). happy(vincent). happy(marsellus). happy(butch). happy(vincent).

To this knowledge base append the following facts happy(jia) at the beginning and happy(john) at the end. Also delete the fact happy(marsellus).

Knowledge Base:



```
% Author - Ajaykumar Nadar

:- dynamic happy/1.
happy(mia).
happy(vincent).
happy(marsellus).
happy(butch).
happy(vincent).
```

Query & Output:

```
Welcome to SWI-Prolog (threaded, 64 bits, version 9.0.4) SWI-Prolog
comes with ABSOLUTELY NO WARRANTY. This is free software. Please run
?- license. for legal details.

For online help and background, visit https://www.swi-prolog.org For
built-in help, use ?- help(Topic). or ?- apropos(Word).

1 ?- [post_exp_2].
true.

2 ?- listing.
:- dynamic happy/1.

happy(mia).
happy(vincent).
happy(marsellus).
happy(butch).
happy(vincent).
true.

3 ?- asserta(happy(jia)).
true.

4 ?- listing.
:- dynamic happy/1.

happy(jia).
happy(mia).
happy(vincent).
happy(marsellus).
happy(butch).
happy(vincent).
true.

5 ?- assertz(happy(john)).
true.

6 ?- listing.
:- dynamic happy/1.

happy(jia).
happy(mia).
happy(vincent).
happy(marsellus).
happy(butch).
happy(vincent).
happy(john).
true.

7 ?- retract(happy(marsellus)).
true.

8 ?- listing.
:- dynamic happy/1.

happy(jia).
happy(mia).
happy(vincent).
happy(butch).
happy(vincent).
happy(john).
true.
```