Define the successor predicate on the first 5 natural numbers

\[
\text{succ}(1,2). \\
\text{succ}(2,3). \\
\text{succ}(3,4). \\
\text{succ}(4,5). \\
\text{succ}(5,6).
\]

Define the less than predicate on the first 3 natural numbers

\[
\text{lt}(1,2). \\
\text{lt}(2,3). \\
\text{lt}(1,3).
\]
Exercises

Given the predicates `female/1`, `male/1`, and `parent/2`, define the following predicates.

(1) `father/2` and `mother/2`

- `father(X,Y) :- parent(X,Y), male(X).`
- `mother(X,Y) :- parent(X,Y), female(X).`

(2) `grandfather/2`, and `grandmother/2`

- `grandfather(X,Y) :- father(X,Z), parent(Z,Y).`
- `grandmother(X,Y) :- mother(X,Z), parent(Z,Y).`

(3) `child/2`, `son/2`, and `daughter/2`

- `child(X,Y) :- parent(Y,X).`
- `son(X,Y) :- child(X,Y), male(X).`
- `daughter(X,Y) :- child(X,Y), female(X).`
Exercises

Given the predicates `female/1`, `male/1`, and `parent/2`, define the following predicates.

(4) **grandchild/2, grandson/2, and granddaughter/2**

- `grandchild(X,Y) :- child(X,Z), child(Z,Y).`
- `grandson(X,Y) :- grandchild(X,Y), male(X).`
- `granddaughter(X,Y) :- grandchild(X,Y), female(X).`

(5) **ancestor/2**

- `ancestor(X,Y) :- parent(X,Y).`
- `ancestor(X,Y) :- parent(X,Z), ancestor(Z,Y).`

(6) **sibling/2, brother/2, and sister/2**

- `sibling(X,Y) :- parent(Z,X), parent(Z,Y), X\=Y.`
- `brother(X,Y) :- sibling(X,Y), male(X).`
- `sister(X,Y) :- sibling(X,Y), female(X).`
Exercises

Given the predicates `female/1`, `male/1`, and `parent/2`, define the following predicates.

(7) **aunt/2 and uncle/2**

   ```prolog
   aunt(X,Y) :- parent(Z,Y), sister(X,Z), \(+ parent(X,Y).
   uncle(X,Y) :- parent(Z,Y), brother(X,Z), \(+ parent(X,Y).
   ```

(8) **mate/2**

   ```prolog
   mate(X,Y) :- parent(X,Z), parent(Y,Z), X\=Y.
   ```

(9) **incest/2**

   ```prolog
   incest(X,Y) :- mate(X,Y), sibling(X,Y).
   ```

male(tywin).
male(jaime).
male(tyrion).
male(joffrey).
male(tommen).
female(joanna).
female(cersei).
female(myrcella).

parent(tywin,jaime).
parent(tywin,cersei).
parent(tywin,tyrion).
parent(joanna,jaime).
parent(joanna,cersei).
parent(joanna,tyrion).
parent(jaime,joffrey).
parent(jaime,tommen).
parent(jaime,myrcella).
parent(cersei,joffrey).
parent(cersei,tommen).
parent(cersei,myrcella).
Exercises

Define predicate `del/3`, such that in `del(X, L, M)` `M` is equal to the list `L` with the first occurrence of `X` removed.

```prolog
del(X, [], []).  
del(X, [X|L], L).  
del(X, [Y|L], [Y|M]) :- del(X, L, M).
```

What is the output of `del(a, [a,b], L)`?

```prolog
?- del(a, [a,b], L).
L = [b] ;
L = [a,b] ;
false.
?- del(a, [b,a], L).
L = [b] ;
L = [b,a] ;
false.
?- del(a, [a,b,a], L).
L = [b,a] ;
L = [a,b] ;
L = [a,b,a] ;
false.
```
Exercises

Avoid unwanted results in the definition of \texttt{del(X, L, M)}.

\begin{align*}
\texttt{del(X, [], [])}.
\texttt{del(X, [X|L], L).}
\texttt{del(X, [Y|L], [Y|M]) :- X\neq Y, del(X, L, M).}
\end{align*}

What is the output of \texttt{del(a, [a,b], L)}?

\begin{align*}
\texttt{?- del(a, [a,b], L).} & \quad \texttt{L = [b] ; false.} \\
\texttt{?- del(a, [b,a], L).} & \quad \texttt{L = [b] ; false.} \\
\texttt{?- del(a, [a,b,a], L).} & \quad \texttt{L = [b,a] ; false.}
\end{align*}
Define a predicate `sublist/2`, such that `sublist(S, L)` holds if the list `S` is a sublist of `L`.

**Examples:**
- `sublist([c,d,e], [a,b,c,d,e,f])` yes
- `sublist([c,e], [a,b,c,d,e,f])` no

**Hint:** use `append/3`

**Hint:** `S` is a sublist of `L` if `L` can be decomposed into `L1` and `L2` and `L2` can be decomposed into `S` and `L3`.

```
sublist(S, L) :- append(L1, L2, L), append(S, L3, L2).
```
Exercises

Define a predicate \texttt{length/2}, such that \texttt{N} in \texttt{length(L, N)} is equal to the length of the list \texttt{L}.

\begin{verbatim}
length([], 0).
length([_|L], N) :- length(L, M), N is M+1.
\end{verbatim}

Can we swap the goals in the body of the second rule?

What does the goal \texttt{length([a,b,c],N)} compute if we use the following, alternative rule?

\begin{verbatim}
length([_|L], N) :- length(L, M), N = M+1.
\end{verbatim}

?- length([a,b,c], N).
\texttt{N = 0+1+1+1.}
Exercises

Simplify the following rule for \textit{length}.

\begin{align*}
\text{length}([\_|L], N) & : - \text{length}(L, M), N = M + 1. \\
\text{length}([\_|L], N+1) & : - \text{length}(L, N).
\end{align*}

Using the above version, which goal can compute the length of a list?

\texttt{?- length([2,4,5], T), N is T.}

\begin{align*}
T & = 0 + 1 + 1 + 1, \\
N & = 3.
\end{align*}