1. (8 pts) Given the following clauses, list all answers returned by the queries. Note, these facts may not be factually accurate.

   a. (3 pts) ?- married(X,Y).
      X = kanye,  
      Y = kim ;
      X = kanye, 
      Y = kourtney ;
      X = scott, 
      Y = kim.

   b. (1 pt) ?- \+child(kim).
      true.

   c. (2 pts) ?- family(X,Y).
      X = kim, 
      Y = kourtney ;
      X = kourtney, 
      Y = kim .

   d. (2 pts) ?- sibling(X,Y).
      X = Y, Y = penelope ;
      X = penelope, 
      Y = north ;
      X = north, 
      Y = penelope ;
      X = Y, Y = north.
**Unification:**
(2 pts) Show the variables bindings (values assigned) if the following queries succeed. Use false otherwise.

1. \( f(X,a,h,g(Y)) = f(h(Z), Z, h(W)) \).
   - false.
2. \( p(X,g(b,Y),L) = p(a,g(Z,f(X)),[X|Zs]) \).
   - \( X = a, Y = f(a), L = [a|Zs], Z = b \).

**Prolog Programming:**
(10 pts) Implement a prolog predicate \( \text{takeNth}(N,L,M) \) such that \( \text{takeNth}(N,L,M) \) is true if and only if \( M \) is the list obtained by taking every \( N \)th element from the list \( L \). For example the query:

\[\]
?- \text{takeNth}(2, [1,2,3,4,5,6,7,8], M).
\]
\( M = [1, 3, 5, 7] \).

?- \text{takeNth}(3, [1,2,3,4,5,6,7,8], M).
\]
\( M = [1, 4, 7] \).

*Hint: Consider writing a helper predicate that drops the first \( N \) elements of the list.*

\[\]
\text{takeNth}(N,List,Taken) :-
\]
\( \text{findall}(A, (\text{nth0}(\text{Index},List,A), 0 \equiv \text{Index} \mod N), Taken) \).

**Rubric:**
1. (5 points) any way of finding the Nth element.
2. (5 points) any way of collecting the Nth elements.