Database (Also in code):

```
age(mary, 109).
age(sarah, 95).
age(tony, 100).
age(grant, 83).
age(hank, 81).
age(kelsey, 84).
age(jake, 43).
age(joe, 53).
age(bob, 50).
age(mike, 49).
age(piper, 52).
age(dillon, 44).
age(justin, 23).
age(christian, 22).
age(taylor, 32).
age(daniel, 29).
age(peter, 31).
child(sarah, mary).
child(tony, mary).
child(grant, sarah).
child(hank, tony).
```

```
child(kelsey, tony).

child(jake, grant).

child(joe, hank).

child(bob, hank).

child(mike, hank).

child(piper, hank).

child(dillon, kelsey).

child(justin, joe).

child(christian, joe).

child(taylor, mike).

child(daniel, piper).

child(peter, piper).
```

Script for Queries:

```
consult("geneology.pl").
consult("functions.pl").

nthcousinkremoved(X,Y,N,K).

% Click semi colon numerous times

kthchild(C,P,K).

% Click semi colon numerous times
```

Output:

```
?- nthcousinkremoved(X,Y,N,K).
X = grant,
Y = hank,
N = 1,
K = 0 ;
```

X = grant,
Y = kelsey,
N = 1,
K = 0 ;
X = hank,
Y = grant,
N = 1,
K = 0 ;
X = kelsey,
Y = grant,
N = 1,
K = 0 ;
X = joe,
Y = dillon,
N = 1,
K = 0 ;
X = bob,
Y = dillon,
N = 1,
K = 0 ;
X = mike,
Y = dillon,
N = 1,
K = 0 ;
X = piper,
Y = dillon,
N = 1,
K = 0 ;
X = dillon,
Y = joe,
N = 1,
K = 0 ;
X = dillon,
Y = bob,
N = 1,
K = 0 ;
X = dillon,
Y = mike,
N = 1,
K = 0 ;
X = dillon,
Y = piper,
N = 1,
K = 0 ;
X = justin,
Y = taylor,

N = 1,
K = 0 ;
X = justin,
Y = daniel,
N = 1,
K = 0 ;
X = justin,
Y = peter,
N = 1,
K = 0 ;
X = christian,
Y = taylor,
N = 1,
K = 0 ;
X = christian,
Y = daniel,
N = 1,
K = 0 ;
X = christian,
Y = peter,
N = 1,
K = 0 ;
X = taylor,
Y = justin,
N = 1,
K = 0 ;
X = taylor,
Y = christian,
N = 1,
K = 0 ;
X = taylor,
Y = daniel,
N = 1,
K = 0 ;
X = taylor,
Y = peter,
N = 1,
K = 0 ;
X = daniel,
Y = justin,
N = 1,
K = 0 ;
X = daniel,
Y = christian,
N = 1,
K = 0 ;

X = daniel,
Y = taylor,
N = 1,
K = 0 ;
X = peter,
Y = justin,
N = 1,
K = 0 ;
X = peter,
Y = christian,
N = 1,
K = 0 ;
X = peter,
Y = taylor,
N = 1,
K = 0 ;
X = jake,
Y = joe,
N = 2,
K = 0 ;
X = jake,
Y = bob,
N = 2,
K = 0 ;
X = jake,
Y = mike,
N = 2,
K = 0 ;
X = jake,
Y = piper,
N = 2,
K = 0 ;
X = jake,
Y = dillon,
N = 2,
K = 0 ;
X = joe,
Y = jake,
N = 2,
K = 0 ;
X = bob,
Y = jake,
N = 2,
K = 0 ;
X = mike,
Y = jake,

N = 2,
K = 0 ;
X = piper,
Y = jake,
N = 2,
K = 0 ;
X = dillon,
Y = jake,
N = 2,
K = 0 ;
X = hank,
Y = jake,
N = K, K = 1 ;
X = kelsey,
Y = jake,
N = K, K = 1 ;
X = grant,
Y = joe,
N = K, K = 1 ;
X = grant,
Y = bob,
N = K, K = 1 ;
X = grant,
Y = mike,
N = K, K = 1 ;
X = grant,
Y = piper,
N = K, K = 1 ;
X = grant,
Y = dillon,
N = K, K = 1 ;
X = dillon,
Y = justin,
N = K, K = 1 ;
X = jake,
Y = justin,
N = 2,
K = 1 ;
X = grant,
Y = justin,
N = 1,
K = 2 ;
X = dillon,
Y = christian,
N = K, K = 1 ;
X = jake,

Y = christian,
N = 2,
K = 1 ;
X = grant,
Y = christian,
N = 1,
K = 2 ;
X = dillon,
Y = taylor,
N = K, K = 1 ;
X = jake,
Y = taylor,
N = 2,
K = 1 ;
X = grant,
Y = taylor,
N = 1,
K = 2 ;
X = dillon,
Y = daniel,
N = K, K = 1 ;
X = jake,
Y = daniel,
N = 2,
K = 1 ;
X = grant,
Y = daniel,
N = 1,
K = 2 ;
X = dillon,
Y = peter,
N = K, K = 1 ;
X = jake,
Y = peter,
N = 2,
K = 1 ;
X = grant,
Y = peter,
N = 1,
K = 2 ;
false.

?- kthchild(C,P,K).
C = jake,
P = grant,
K = 1 ;

C = joe,
P = hank,
K = 1 ;
C = piper,
P = hank,
K = 2 ;
C = bob,
P = hank,
K = 3 ;
C = mike,
P = hank,
K = 4 ;
C = justin,
P = joe,
K = 1 ;
C = christian,
P = joe,
K = 2 ;
C = dillon,
P = kelsey,
K = 1 ;
C = tony,
P = mary,
K = 1 ;
C = sarah,
P = mary,
K = 2 ;
C = taylor,
P = mike,
K = 1 ;
C = peter,
P = piper,
K = 1 ;
C = daniel,
P = piper,
K = 2 ;
C = grant,
P = sarah,
K = 1 ;
C = kelsey,
P = tony,
K = 1 ;
C = hank,
P = tony,
K = 2.

What I learned:

Like many of my peers, this assignment marked my initial foray into the world of Prolog. The language, with its unique syntax and structure, was a departure from the programming languages I was accustomed to. This divergence initially sparked a sense of panic within me. However, as I delved deeper into the language, I found myself adapting to its intricacies with haste.

One of the most enlightening aspects of this assignment was the introduction to the concept of defining predicates. Despite their simplicity, predicates possess an inherent cleverness in their design and functionality. They offered a fresh perspective on creating functionality, which was both challenging and rewarding.

Another significant learning curve was the use of recursion. Prior to this assignment, my comfort level with recursion was somewhat limited. However, the necessity of recursion in creating certain predicates, such as ‘nth cousin’ and ‘nth cousin k removed’, compelled me to confront and overcome this challenge.

In retrospect, I am grateful for the opportunity this assignment provided to not only learn a new language but also to enhance my understanding of recursion. It was a rewarding experience that allowed me to expand my programming skills and problem-solving abilities. This assignment served as a reminder that stepping out of one’s comfort zone often leads to the most significant learning experiences.