1. Recall the (informal) definition of logical entailment:

A collection of sentences $S_1, S_2, ..., S_n$ logically entails another sentence $S$ if the truth of $S$ is implicit in the truth of the $S_i$ sentences. (Therefore, the meaning of the terms in the $S_i$ sentences do not matter in determining whether $S$ is logically entailed by $S_1, S_2, ..., S_n$.)

Consider the following knowledge base (KB), written as a Prolog program.

```prolog
dog(X) :- poodle(X).
dog(X) :- collie(X).
poodle(X) :- poodle(X).
collie(fido).
```

Recall that Prolog uses back-chaining in answering queries.

(a) The KB above logically entails $\text{dog(fido)}$. True or false? Answer: True.

(b) The query $\text{dog(fido)}$ will not succeed. True or false? Answer: True.

(c) Back-chaining is complete on Prolog KBs. True or false? Answer: False.

2. Consider the following incorrect Prolog program.

```prolog
% factorial(N,M) holds when M=Nx(N-1)x...x2x1 (and when N=0 and M-1).
factorial(0,1).
factorial(N,F) :- N>0, factorial(N,F1), F is N*F1.
```

(a) Does this program terminate? Answer: No. (It actually runs out of memory.)

(b) Correct the second clause of the program so that it correctly computes the factorial. Answer:

```prolog
factorial(0,1).
factorial(N,F) :- N>0, N1 is N-1, factorial(N1,F1), F is N*F1.
```