CZ3005: TS4 - Lab II

Aurelio Jethro - U1921390C

Exercise One

Translate the natural language statements above describing the dealing within the Smart Phone Industry in to First Order Logic (FOL)

Constants: sumsum, appy, galactica-s3, stevey

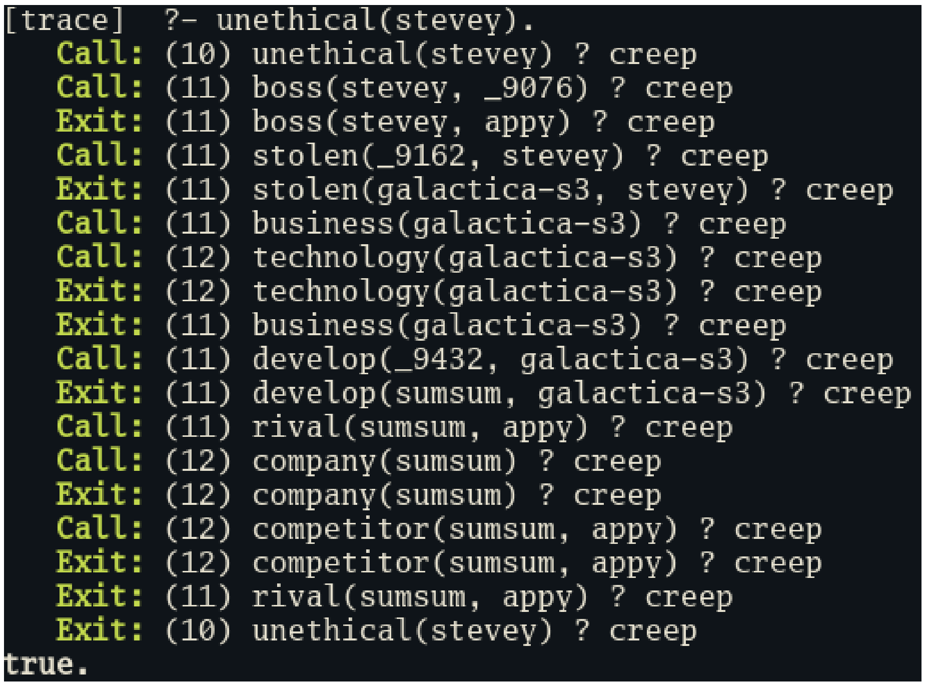
Predicates:

- Company(x) - “x is a company”
- Competitor(x, y) - “x is a competitor of y”
- Technology(x) - “x is a smart phone technology”
- Developed(x, y) - “x develops y”
- Stolen(x, y) - “x is stolen by y”
- Boss(x, y) - “x is a boss of y”
- Rival(x, y) - “x is a rival of y”
- Business(x) - “x is a business”
- Unethical(x) - “x is unethical”

Sentence:

- Company(sumsum)
- Company(appy)
- Competitor(sumsum, appy)
- Technology(galactica-s3)
- Developed(sumsum, galactica-s3)
- Stolen(galactica-s3, stevey)
- Boss(stevey, appy)
- $\forall$x Technology(x) $\Rightarrow$ Business(x)
- $\forall$x, (Company(x), Competitor(x, appy)) $\Rightarrow$ Rival(x, appy)
- $\forall$x, y, z, c (Boss(x, y) $\land$ Stolen(z, x) $\land$ Business(z) $\land$ Developed(c, z) $\land$ Rival(c, y) $\Rightarrow$ Unethical(x))

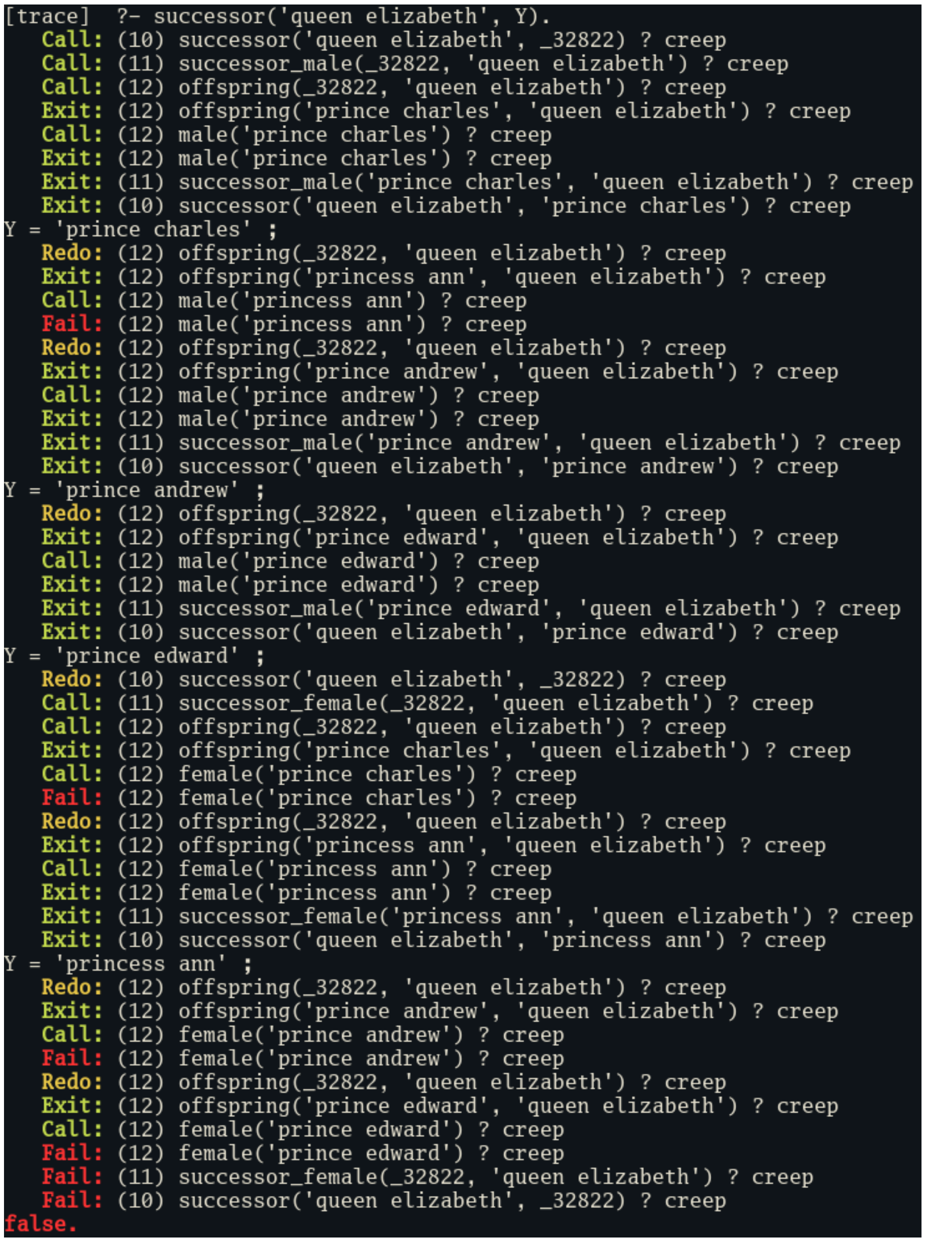
Using Prolog, prove that Stevey is unethical. Show a trace of your proof.



```
[trace]  ?- unethical(stevey).
   Call: (10) unethical(stevey) ? creep
   Call: (11) boss(stevey, _9076) ? creep
   Exit: (11) boss(stevey, appy) ? creep
   Call: (11) stolen(_9162, stevey) ? creep
   Exit: (11) stolen(galactica-s3, stevey) ? creep
   Call: (11) business(galactica-s3) ? creep
   Call: (12) technology(galactica-s3) ? creep
   Exit: (12) technology(galactica-s3) ? creep
   Exit: (11) business(galactica-s3) ? creep
   Call: (11) develop(_9432, galactica-s3) ? creep
   Exit: (11) develop(sumsum, galactica-s3) ? creep
   Call: (11) rival(sumsum, appy) ? creep
   Call: (12) company(sumsum) ? creep
   Exit: (12) company(sumsum) ? creep
   Call: (12) competitor(sumsum, appy) ? creep
   Exit: (12) competitor(sumsum, appy) ? creep
   Exit: (11) rival(sumsum, appy) ? creep
   Exit: (10) unethical(stevey) ? creep
true.
```

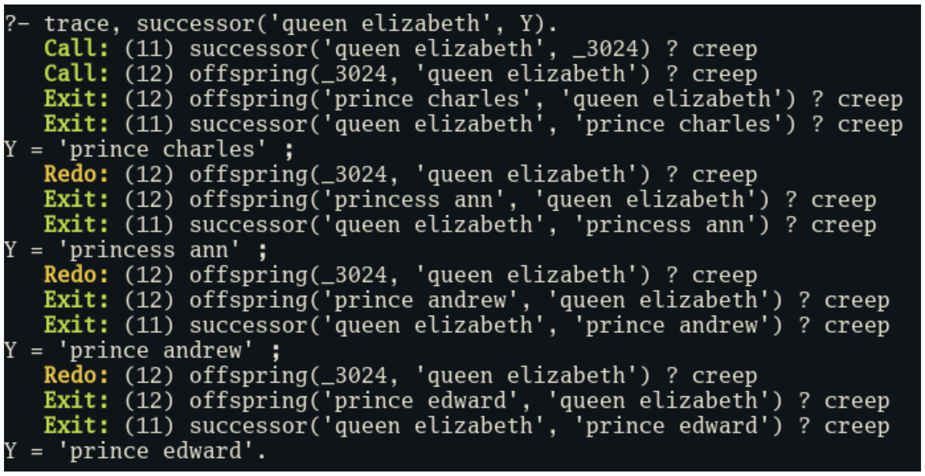
Exercise Two

Define their relations and rules in a Prolog rule base. Hence, define the old Royal succession rule. Using this old succession rule determine the line of succession based on the information given. Do a trace to show your results.



```
[trace]  ?- successor('queen elizabeth', Y).
   Call: (10) successor('queen elizabeth', _32822) ? creep
   Call: (11) successor_male(_32822, 'queen elizabeth') ? creep
   Call: (12) offspring(_32822, 'queen elizabeth') ? creep
   Exit: (12) offspring('prince charles', 'queen elizabeth') ? creep
   Call: (12) male('prince charles') ? creep
   Exit: (12) male('prince charles') ? creep
   Exit: (11) successor_male('prince charles', 'queen elizabeth') ? creep
   Exit: (10) successor('queen elizabeth', 'prince charles') ? creep
Y = 'prince charles' ;
   Redo: (12) offspring(_32822, 'queen elizabeth') ? creep
   Exit: (12) offspring('princess ann', 'queen elizabeth') ? creep
   Call: (12) male('princess ann') ? creep
   Fail: (12) male('princess ann') ? creep
   Redo: (12) offspring(_32822, 'queen elizabeth') ? creep
   Exit: (12) offspring('prince andrew', 'queen elizabeth') ? creep
   Call: (12) male('prince andrew') ? creep
   Exit: (12) male('prince andrew') ? creep
   Exit: (11) successor_male('prince andrew', 'queen elizabeth') ? creep
   Exit: (10) successor('queen elizabeth', 'prince andrew') ? creep
Y = 'prince andrew' ;
   Redo: (12) offspring(_32822, 'queen elizabeth') ? creep
   Exit: (12) offspring('prince edward', 'queen elizabeth') ? creep
   Call: (12) male('prince edward') ? creep
   Exit: (12) male('prince edward') ? creep
   Exit: (11) successor_male('prince edward', 'queen elizabeth') ? creep
   Exit: (10) successor('queen elizabeth', 'prince edward') ? creep
Y = 'prince edward' ;
   Redo: (10) successor('queen elizabeth', _32822) ? creep
   Call: (11) successor_female(_32822, 'queen elizabeth') ? creep
   Call: (12) offspring(_32822, 'queen elizabeth') ? creep
   Exit: (12) offspring('prince charles', 'queen elizabeth') ? creep
   Call: (12) female('prince charles') ? creep
   Fail: (12) female('prince charles') ? creep
   Redo: (12) offspring(_32822, 'queen elizabeth') ? creep
   Exit: (12) offspring('princess ann', 'queen elizabeth') ? creep
   Call: (12) female('princess ann') ? creep
   Exit: (12) female('princess ann') ? creep
   Exit: (11) successor_female('princess ann', 'queen elizabeth') ? creep
   Exit: (10) successor('queen elizabeth', 'princess ann') ? creep
Y = 'princess ann' ;
   Redo: (12) offspring(_32822, 'queen elizabeth') ? creep
   Exit: (12) offspring('prince andrew', 'queen elizabeth') ? creep
   Call: (12) female('prince andrew') ? creep
   Fail: (12) female('prince andrew') ? creep
   Redo: (12) offspring(_32822, 'queen elizabeth') ? creep
   Exit: (12) offspring('prince edward', 'queen elizabeth') ? creep
   Call: (12) female('prince edward') ? creep
   Fail: (12) female('prince edward') ? creep
   Fail: (11) successor_female(_32822, 'queen elizabeth') ? creep
   Fail: (10) successor('queen elizabeth', _32822) ? creep
false.
```

Recently, the Royal succession rule has been modified. The throne is now passed down according to the order of birth irrespective of gender. Modify your rules and Prolog knowledge base to handle the new succession rule. Explain the necessary changes to the knowledge needed to represent the new information. Use this new succession rule to determine the new line of succession based on the same knowledge given. Show your results using a trace.



```
?- trace, successor('queen elizabeth', Y).
   Call: (11) successor('queen elizabeth', _3024) ? creep
   Call: (12) offspring(_3024, 'queen elizabeth') ? creep
   Exit: (12) offspring('prince charles', 'queen elizabeth') ? creep
   Exit: (11) successor('queen elizabeth', 'prince charles') ? creep
Y = 'prince charles' ;
   Redo: (12) offspring(_3024, 'queen elizabeth') ? creep
   Exit: (12) offspring('princess ann', 'queen elizabeth') ? creep
   Exit: (11) successor('queen elizabeth', 'princess ann') ? creep
Y = 'princess ann' ;
   Redo: (12) offspring(_3024, 'queen elizabeth') ? creep
   Exit: (12) offspring('prince andrew', 'queen elizabeth') ? creep
   Exit: (11) successor('queen elizabeth', 'prince andrew') ? creep
Y = 'prince andrew' ;
   Redo: (12) offspring(_3024, 'queen elizabeth') ? creep
   Exit: (12) offspring('prince edward', 'queen elizabeth') ? creep
   Exit: (11) successor('queen elizabeth', 'prince edward') ? creep
Y = 'prince edward'.
```