CZ3500	**Name:** Singh Jasraj
Assignment 2	**Matric:** U1940558D
April 11, 2023	**Collaborators:** None

Exercise 1.1.

Constants:

- samsum
- appy
- stevey
- galacticas3

Predicates:

- Technology(galacticas3)
  "galacticas3 is a technology"
- Rival(samsum, appy)
  "samsum is a rival of appy"
- Own(samsum, galacticas3)
  "samsum owns galacticas3"
- Boss(appy, stevey)
  "stevey is the boss of appy"
- Steal(stevey, galacticas3)
  "stevey steals galacticas3"

Sentences:

- Technology(A) $\Rightarrow$ Business(A)
  "if A is a technology, then A is a business"
- Rival(A, B) $\Rightarrow$ Rival(B, A)
  "if A is a rival of B, then B is a rival of A"
- Steal(A, X) $\wedge$ Business(X) $\wedge$ own(Y, X) $\wedge$ boss(Z, A) $\wedge$ Rival(Y, Z) $\Rightarrow$ Unethical(A)
  "if A steals X and X is a business and Y owns X and A is the boss of Z and Y is a rival of Z, then A is unethical"

Exercise 1.3.

```
?- trace, unethical(stevey).
   Call: (11) unethical(stevey) ? creep
   Call: (12) steal(stevey, _16714) ? creep
   Exit: (12) steal(stevey, galacticas3) ? creep
```

```
   Call: (12) business(galacticas3) ? creep
   Call: (13) technology(galacticas3) ? creep
   Exit: (13) technology(galacticas3) ? creep
   Exit: (12) business(galacticas3) ? creep
   Call: (12) own(_21560, galacticas3) ? creep
   Exit: (12) own(samsum, galacticas3) ? creep
   Call: (12) boss(_23182, stevey) ? creep
   Exit: (12) boss(appy, stevey) ? creep
   Call: (12) rival(samsum, appy) ? creep
   Exit: (12) rival(samsum, appy) ? creep
   Exit: (11) unethical(stevey) ? creep
true .
```

Exercise 2.1.

```
?- trace, successors(elizabeth, S).
   Call: (11) successors(elizabeth, _15342) ? creep
   Call: (12) findall(_16756, child(_16756, elizabeth), _16764) ? creep
   Call: (17) child(_16756, elizabeth) ? creep
   Call: (18) son(_16756, elizabeth) ? creep
   Exit: (18) son(charles, elizabeth) ? creep
   Exit: (17) child(charles, elizabeth) ? creep
   Redo: (18) son(_16756, elizabeth) ? creep
   Exit: (18) son(andrew, elizabeth) ? creep
   Exit: (17) child(andrew, elizabeth) ? creep
   Redo: (18) son(_16756, elizabeth) ? creep
   Exit: (18) son(edward, elizabeth) ? creep
   Exit: (17) child(edward, elizabeth) ? creep
   Redo: (17) child(_16756, elizabeth) ? creep
   Call: (18) daughter(_16756, elizabeth) ? creep
   Exit: (18) daughter(ann, elizabeth) ? creep
   Exit: (17) child(ann, elizabeth) ? creep
   Exit: (12) findall(_16756, user:child(_16756, elizabeth), [charles, andrew,
      edward, ann]) ? creep
   Call: (12) sort_successors([charles, andrew, edward, ann], _15342) ? creep
   Call: (13) sort_successors([andrew, edward, ann], _30626) ? creep
   Call: (14) sort_successors([edward, ann], _31438) ? creep
   Call: (15) sort_successors([ann], _32250) ? creep
   Call: (16) sort_successors([], _33062) ? creep
   Exit: (16) sort_successors([], []) ? creep
   Call: (16) insert_successor(ann, [], _32250) ? creep
   Exit: (16) insert_successor(ann, [], [ann]) ? creep
   Exit: (15) sort_successors([ann], [ann]) ? creep
   Call: (15) insert_successor(edward, [ann], _31438) ? creep
   Call: (16) not(precedes(edward, ann)) ? creep
   Call: (17) precedes(edward, ann) ? creep
   Call: (18) male(edward) ? creep
   Call: (19) son(edward, _40468) ? creep
   Exit: (19) son(edward, elizabeth) ? creep
   Exit: (18) male(edward) ? creep
   Call: (18) female(ann) ? creep
   Call: (19) daughter(ann, _43700) ? creep
   Exit: (19) daughter(ann, elizabeth) ? creep
   Exit: (18) female(ann) ? creep
```

```
Exit: (17) precedes(edward, ann) ? creep
Fail: (16) not(user:precedes(edward, ann)) ? creep
Redo: (15) insert_successor(edward, [ann], _31438) ? creep
Exit: (15) insert_successor(edward, [ann], [edward, ann]) ? creep
Exit: (14) sort_successors([edward, ann], [edward, ann]) ? creep
Call: (14) insert_successor(andrew, [edward, ann], _30626) ? creep
Call: (15) not(precedes(andrew, edward)) ? creep
Call: (16) precedes(andrew, edward) ? creep
Call: (17) male(andrew) ? creep
Call: (18) son(andrew, _53464) ? creep
Exit: (18) son(andrew, elizabeth) ? creep
Exit: (17) male(andrew) ? creep
Call: (17) female(edward) ? creep
Call: (18) daughter(edward, _56696) ? creep
Fail: (18) daughter(edward, _57506) ? creep
Fail: (17) female(edward) ? creep
Redo: (16) precedes(andrew, edward) ? creep
Call: (17) male(andrew) ? creep
Call: (18) son(andrew, _60738) ? creep
Exit: (18) son(andrew, elizabeth) ? creep
Exit: (17) male(andrew) ? creep
Call: (17) male(edward) ? creep
Call: (18) son(edward, _162) ? creep
Exit: (18) son(edward, elizabeth) ? creep
Exit: (17) male(edward) ? creep
Call: (17) older(andrew, edward) ? creep
Exit: (17) older(andrew, edward) ? creep
Exit: (16) precedes(andrew, edward) ? creep
Fail: (15) not(user:precedes(andrew, edward)) ? creep
Redo: (14) insert_successor(andrew, [edward, ann], _70) ? creep
Exit: (14) insert_successor(andrew, [edward, ann], [andrew, edward, ann]) ?
   creep
Exit: (13) sort_successors([andrew, edward, ann], [andrew, edward, ann]) ?
   creep
Call: (13) insert_successor(charles, [andrew, edward, ann], _18) ? creep
Call: (14) not(precedes(charles, andrew)) ? creep
Call: (15) precedes(charles, andrew) ? creep
Call: (16) male(charles) ? creep
Call: (17) son(charles, _11292) ? creep
Exit: (17) son(charles, elizabeth) ? creep
Exit: (16) male(charles) ? creep
Call: (16) female(andrew) ? creep
Call: (17) daughter(andrew, _14524) ? creep
Fail: (17) daughter(andrew, _15334) ? creep
Fail: (16) female(andrew) ? creep
Redo: (15) precedes(charles, andrew) ? creep
Call: (16) male(charles) ? creep
Call: (17) son(charles, _18566) ? creep
Exit: (17) son(charles, elizabeth) ? creep
Exit: (16) male(charles) ? creep
Call: (16) male(andrew) ? creep
Call: (17) son(andrew, _21798) ? creep
Exit: (17) son(andrew, elizabeth) ? creep
Exit: (16) male(andrew) ? creep
Call: (16) older(charles, andrew) ? creep
Exit: (16) older(charles, andrew) ? creep
```

```
   Exit: (15) precedes(charles, andrew) ? creep
   Fail: (14) not(user:precedes(charles, andrew)) ? creep
   Redo: (13) insert_successor(charles, [andrew, edward, ann], _18) ? creep
   Exit: (13) insert_successor(charles, [andrew, edward, ann], [charles, andrew
      , edward, ann]) ? creep
   Exit: (12) sort_successors([charles, andrew, edward, ann], [charles, andrew,
       edward, ann]) ? creep
   Exit: (11) successors(elizabeth, [charles, andrew, edward, ann]) ? creep
S = [charles, andrew, edward, ann].
```

Exercise 2.2.

```
?- trace, successors(elizabeth, S).
   Call: (11) successors(elizabeth, _15342) ? creep
   Call: (12) findall(_16756, child(_16756, elizabeth), _16764) ? creep
   Call: (17) child(_16756, elizabeth) ? creep
   Call: (18) son(_16756, elizabeth) ? creep
   Exit: (18) son(charles, elizabeth) ? creep
   Exit: (17) child(charles, elizabeth) ? creep
   Redo: (18) son(_16756, elizabeth) ? creep
   Exit: (18) son(andrew, elizabeth) ? creep
   Exit: (17) child(andrew, elizabeth) ? creep
   Redo: (18) son(_16756, elizabeth) ? creep
   Exit: (18) son(edward, elizabeth) ? creep
   Exit: (17) child(edward, elizabeth) ? creep
   Redo: (17) child(_16756, elizabeth) ? creep
   Call: (18) daughter(_16756, elizabeth) ? creep
   Exit: (18) daughter(ann, elizabeth) ? creep
   Exit: (17) child(ann, elizabeth) ? creep
   Exit: (12) findall(_16756, user:child(_16756, elizabeth), [charles, andrew,
      edward, ann]) ? creep
   Call: (12) sort_successors([charles, andrew, edward, ann], _15342) ? creep
   Call: (13) sort_successors([andrew, edward, ann], _30626) ? creep
   Call: (14) sort_successors([edward, ann], _31438) ? creep
   Call: (15) sort_successors([ann], _32250) ? creep
   Call: (16) sort_successors([], _33062) ? creep
   Exit: (16) sort_successors([], []) ? creep
   Call: (16) insert_successor(ann, [], _32250) ? creep
   Exit: (16) insert_successor(ann, [], [ann]) ? creep
   Exit: (15) sort_successors([ann], [ann]) ? creep
   Call: (15) insert_successor(edward, [ann], _31438) ? creep
   Call: (16) not(precedes(edward, ann)) ? creep
   Call: (17) precedes(edward, ann) ? creep
   Call: (18) older(edward, ann) ? creep
   Fail: (18) older(edward, ann) ? creep
   Fail: (17) precedes(edward, ann) ? creep
   Exit: (16) not(user:precedes(edward, ann)) ? creep
   Call: (16) insert_successor(edward, [], _37946) ? creep
   Exit: (16) insert_successor(edward, [], [edward]) ? creep
   Exit: (15) insert_successor(edward, [ann], [ann, edward]) ? creep
   Exit: (14) sort_successors([edward, ann], [ann, edward]) ? creep
   Call: (14) insert_successor(andrew, [ann, edward], _30626) ? creep
   Call: (15) not(precedes(andrew, ann)) ? creep
   Call: (16) precedes(andrew, ann) ? creep
```

```
   Call: (17) older(andrew, ann) ? creep
   Fail: (17) older(andrew, ann) ? creep
   Fail: (16) precedes(andrew, ann) ? creep
   Exit: (15) not(user:precedes(andrew, ann)) ? creep
   Call: (15) insert_successor(andrew, [edward], _46912) ? creep
   Call: (16) not(precedes(andrew, edward)) ? creep
   Call: (17) precedes(andrew, edward) ? creep
   Call: (18) older(andrew, edward) ? creep
   Exit: (18) older(andrew, edward) ? creep
   Exit: (17) precedes(andrew, edward) ? creep
   Fail: (16) not(user:precedes(andrew, edward)) ? creep
   Redo: (15) insert_successor(andrew, [edward], _46912) ? creep
   Exit: (15) insert_successor(andrew, [edward], [andrew, edward]) ? creep
   Exit: (14) insert_successor(andrew, [ann, edward], [ann, andrew, edward]) ?
      creep
   Exit: (13) sort_successors([andrew, edward, ann], [ann, andrew, edward]) ?
      creep
   Call: (13) insert_successor(charles, [ann, andrew, edward], _15342) ? creep
   Call: (14) not(precedes(charles, ann)) ? creep
   Call: (15) precedes(charles, ann) ? creep
   Call: (16) older(charles, ann) ? creep
   Exit: (16) older(charles, ann) ? creep
   Exit: (15) precedes(charles, ann) ? creep
   Fail: (14) not(user:precedes(charles, ann)) ? creep
   Redo: (13) insert_successor(charles, [ann, andrew, edward], _18) ? creep
   Exit: (13) insert_successor(charles, [ann, andrew, edward], [charles, ann,
      andrew, edward]) ? creep
   Exit: (12) sort_successors([charles, andrew, edward, ann], [charles, ann,
      andrew, edward]) ? creep
   Exit: (11) successors(elizabeth, [charles, ann, andrew, edward]) ? creep
S = [charles, ann, andrew, edward].
```