W6.2 Predicate logic assignments

For this week, please perform the following assignments.

In [3]:

```
from pyprover import *
fill_in_by_student = "Vidhya Narayanasamy(2733527)"
```

Assignment 1

Please use PyProver to formalize the problem below:
John gave something to Peter.

First define the required predicates:

In [4]:

```
props("G S") # G being the predicate for 'giving' something to someone and S represents the
```

Out[4]:

```
map(<class 'pyprover.logic.Prop'>, ['G', 'S'])
```

Then define the required objects and variables

In [5]:

```
terms("x j p") # x indicates the object 'something'; j indicates the object 'John'; p indic
```

Out[5]:

```
map(<class 'pyprover.logic.Const'>, ['x', 'j', 'p'])
```

Then formalize the problem:

In [6]:

```
f1 = TE(x,(S(x) & G(j, x, p)))
```

In [7]:

```
print(f1)
```

```
TE x, (S(x) & G(j, x, p))
```

Assignment 2

Please use PyProver to formalize the following statements:

1. Everything is subject to decay.
2. John gave Peter nothing.
3. John gave Peter a book.
4. A whale is a mammal.
5. Boys who are late are to be punished.

2.1 Everything is subject to decay.

Please use PyProver to formalize the following statement: Everything is subject to decay.

First define the required predicates:

In [8]:

```
props("D") # with D for predicate Decay
```

Out[8]:

```
map(<class 'pyprover.logic.Prop'>, ['D'])
```

Then define the required objects and variables

In [9]:

```
terms("e") # with variable 'e' representing everything
```

Out[9]:

```
map(<class 'pyprover.logic.Const'>, ['e'])
```

Then formalize the problem:

In [10]:

```
# Everything is subject to decay.
f2_1 = FA(e, D(e))
```

In [11]:

```
print(f2_1)
```

```
FA e, D(e)
```

2.2 John gave Peter nothing.

Please use PyProver to formalize the following statement: John gave Peter nothing.

First define the required predicates:

In [12]:

```
props("G S") # G being the predicate for 'giving' something to someone and S represents the
```

Out[12]:

```
map(<class 'pyprover.logic.Prop'>, ['G', 'S'])
```

Then define the required objects and variables

In [13]:

```
terms("x j p") # x indicates the object 'something'; j indicates the object 'John'; p indic
```

Out[13]:

```
map(<class 'pyprover.logic.Const'>, ['x', 'j', 'p'])
```

Then formalize the problem:

In [14]:

```
f2_2 = ~f1
```

In [15]:

```
print(f2_2)
```

```
~(TE x, (S(x) & G(j, x, p)))
```

2.3 John gave Peter a book.

Please use PyProver to formalize the following statement: John gave Peter a book.

In [16]:

```
props("G B") #  G being the predicate for 'giving' something to someone and B represents th
terms("x j p") # x indicates the object 'something'; j indicates the object 'John'; p indic

f2_3 = TE(x,(B(x) & G(j, x, p)))
print(f2_3)
```

```
TE x, (B(x) & G(j, x, p))
```

2.4 A whale is a mammal.

Please use PyProver to formalize the following statement: A whale is a mammal.

In [17]:

```
props("W M") # W being the predicate for 'being' something and M represents the predicate o
terms("x") # object x represents something
f2_4 = FA(x, W(x) >> M(x))
print(f2_4)
```

```
FA x, (W(x) -> M(x))
```

2.5 Boys who are late are to be punished.

Please use PyProver to formalize the following statement: Boys who are late are to be punished.

In [18]:

```
# L is the predicate being 'Late'; P is the predicate for being 'punished'; B represents th
props("L P B")
terms("x") # object x represents'something'
f2_5 = FA(x, (B(x)&L(x)) >> P(x))
print(f2_5)

# For Double checking
#props("E P B") # with E for predicate being early
#terms("x")
#f2 = FA(x, (B(x)&(~E(x))) >> P(x))
#print("f2:", f2)
```

```
FA x, ((B(x) & L(x)) -> P(x))
```

Assignment 3

Please use PyProver to formalize the problem below and prove that the statements lead to a contraction (i.e. ⊥):
Mary is wearing a blue skirt.
Nobody is wearing a blue skirt.

First define the required predicates:

In [19]:

```
props("S W") # BS is the predicate of being a blue skirt and W is the predicate for being w
```

Out[19]:

```
map(<class 'pyprover.logic.Prop'>, ['S', 'W'])
```

Then define the required objects and variables

In [20]:

```
terms("x m") # object x represents skirt and m represents Mary
```

Out[20]:

```
map(<class 'pyprover.logic.Const'>, ['x', 'm'])
```

Then formalize the problem:

In [21]:

```
f3_1 = (S(m) & W(m))   # Formula for 'Mary is wearing a blue skirt'
f3_2 = FA(x, (W(x) >> ~S(x)))        # Formula for 'Nobody is wearing a blue skirt'
```

In [22]:

```
print(f3_1)
print(f3_2)
```

```
S(m) & W(m)
FA x, (W(x) -> ~S(x))
```

And finally check for the contradiction:

In [23]:

```
print(proves((f3_1, f3_2), bot))
```

```
True
```

Assignment 4

Please use PyProver to formalize the problem below and prove that the statements lead to a contraction (i.e. $\perp$):
If the train is late and there is no bus at the station, no person will arrive on time.
The train is late
There is no bus at the station
Simon is a person
Simon is not late

First define the required predicates:

In [32]:

```
# L is the predicate for being late; P is the predicate for being a person; R is the prdica
props("L P R")
```

Out[32]:

```
map(<class 'pyprover.logic.Prop'>, ['L', 'P', 'R'])
```

Then define the required objects and variables

In [34]:

```
terms("x s t b") # object s represents Simon; object t represents the train; object b repre
```

Out[34]:

```
map(<class 'pyprover.logic.Const'>, ['x', 's', 't', 'b'])
```

Then formalize the problem:

In [35]:

```
f4_1 = (L(t)&~R(b))>>(FA(x, L(s)))       # No one will arrive on time if the train is late a
f4_2 = L(t)         # The train is late
f4_3 = ~R(b)         # There is no bus at the station
f4_4 = P(s)         # Simon is a person
f4_5 = ~L(s)      # Simon is not late
```

In [36]:

```
print(f4_1)
print(f4_2)
print(f4_3)
print(f4_4)
print(f4_5)
```

```
(L(t) & ~R(b)) -> (FA x, L(s))
L(t)
~R(b)
P(s)
~L(s)
```

And finally check for the contradiction:

In [37]:

```
proves((f4_1, f4_2, f4_3, f4_4, f4_5), bot)
```

Out[37]:

```
True
```