CMSC 671 - HW 5
Name : Akshay Peshave

Q1. Checking validity

```
>>> tt_true(Expr('|', P, expr('~P')))
True
>>>
>>> tt_true(Expr('>>', P, P))
True
>>>
>>> tt_true(Expr('>>', P, expr('P | Q')))
True
>>>
>>> tt_true(Expr('>>', expr('P | Q'), P))
False
>>>
>>> tt_true(Expr('<=>', Expr('>>', expr('A & B'), expr('C')), Expr('>>', A, expr('B >> C'))))
True
>>>
>>> tt_true(Expr('>>', Expr('>>', expr('A >> B'), A), A))
True
```

Q2. Satisfiability

```
>>> dpll_satisfiable(Expr('>>', 'Alive', NaryExpr('&', expr('~Dead'), expr('~Alive'), expr('~Dead'))))
{Alive: False}
>>>
>>> dpll_satisfiable(Expr('>>', P, Expr('|', expr('~P'), P)))
{P: True}
>>>
>>> dpll_satisfiable(Expr('~', Expr('|', P, expr('~P'))))
False
```

Q3. Propositional Consequence

2. $p \models p \wedge q$: False
3. $p \models p \vee q$: True
4. $p \models \neg \neg p$: True
5. $p \rightarrow q \models \neg p \rightarrow \neg q$: False

6. ¬ p ⊨ p → q : True
7. ¬ q ⊨ p → q : False
8. p, p → q ⊨ q : True
9. ¬ p , q → p ⊨ ¬ q : True

Q4. English to FOL

2. Everything is either dead or alive.

   A x things(x) **^** (dead(x) **v** alive(x))
3. Dead things are not animate.

   A x things(x) **^** dead(x) **^ ~**animate(x)
4. Zombies are not alive but they are animate.

   A x zombies(x) **^ ~**alive(x) **^** animate(x)
5. Good food is not cheap and cheap food is not good.

   A x food(x) **^** (good(x) **<=> ~**cheap(x))
6. John has exactly two brothers.

   E x, y brother(John, x) **^** brother(John, y) **^ ~(**x=y) **^ ~**(**E** z brother(John, z) **^ ~**(z=x) **^ ~**(z=y))
7. No person can have two mothers.

   A x, y, z person(x) **^** person(y) **^** person(z) **^** mother(x, z) **^** mother(y, z) **^** (y=z)
8. If John has a sister, she is smart.

   E x sister(John, x) **=>** smart(x)
9. Every person is either male or female and no person can be both male and female.

   A x person(x) **=>** (male(x) **v** female(x)) **^ ~**(male(x) **^** female(x))
10. The enemy of your enemy is your friend.

    A x, y (enemy(you, x) **^** enemy(y, x)) **=>** friend(you, y)
11. An ancestor of your ancestor is your ancestor.

A x, y (ancestor(you, x) **^** ancestor(x, y)) **=>** ancestor(you, y)

Q5. CNF and Horn Clauses

2. ∀x ∀y married(x, y) → loves(x, y) v hates(x, y)

   Everyone either loves or hates those they have been married to.

   It cannot be expressed as a set of horn clause.

   CNF : (loves(x, y) | ~married(x, y) | hates(x, y))

3. ∀x ∀y loves(x, y) ↔ loves(y, x)

   Everyone loves all those who love them.

   It can be expressed as a set of horn clauses.

   CNF : ((loves(x, y) | ~loves(y, x)) & (loves(y, x) | ~loves(x, y)))

   Horn Clause : {(loves(x, y) | ~loves(y, x)) , (loves(y, x) | ~loves(x, y)}

4. ∀x ∀y dating(x, y) v engaged(x, y) → knows(x, y) ∧likes(x, y)

   Everyone knows and likes those they are dating or are engaged to.

   It can be expressed as a set of horn clauses.

   CNF : ((~dating(x, y) | knows(x, y)) & (~engaged(x, y) | knows(x, y)) & (~dating(x, y) | likes(x, y)) & (~engaged(x, y) | likes(x, y)))

   Horn Clause : {(~dating(x, y) | knows(x, y)) , (~engaged(x, y) | knows(x, y)) , (~dating(x, y) | likes(x, y)) , (~engaged(x, y) | likes(x, y))}

5. ∀x ∀y loves(x, y) → ¬ hates(x, y)

   No one hates those they love.

   It can be expressed as a set of horn clauses.

   (~hates(x, y) | ~loves(x, y))

Horn Clause : {(~hates(x, y) | ~loves(x, y))}

6. ∀x ∀y ¬ knows(x, y) → ¬ likes(x, y)

   No one likes those they don't know.

   It can be expressed as a set of horn clauses.

   (~likes(x, y) | knows(x, y))

   Horn Clause : {(~likes(x, y) | knows(x, y))}

7. ∀x ∃y knows(x, y) ∧ hates(x, y)

   Everyone hates someone they know.

   It can be expressed as a set of horn clauses.

   CNF : knows(x, F(x)) ∧ hates(x, F(x))

   Horn Clause : {knows(x, F(x)) , hates(x, F(x))}

   skolem functions : {F(x)}

8. ∃y ∀x knows(x, y) ∧ hates(x, y)

   Some are hated by all those who know them.

   It can be expressed as a set of horn clauses.

   CNF : knows(x, A) ∧ hates(x, A)

   Horn Clause : {knows(x, A) , hates(x, A)}

   skolem constants : {A}

9. ¬ (∀x loves(x, x))

There are some who don't love themselves.

It can be expressed as a set of horn clauses.

CNF : ~loves(A,A)

Horn Clause : {~loves(A,A)}

skolem constants : {A}

10. ¬ (∃x ∀y knows(x, y))

None knows everyone.

It can be expressed as a set of horn clauses.

CNF : ~knows(x,F(x))

Horn Clause : {~knows(x,F(x))}

skolem functions : {F(x)}