CS113 LAB 4 - Truth Table

SML

```
fun ttgen_ex((a:'a,b:'a),[ ]) = [ ] |
    ttgen_ex((a:'a,b:'a), q::qs : 'a list list) =
        [q @ [a]] @ [q @ [b]] @ ttgen_ex((a,b), qs) : 'a list list;
fun ttgen(0, (_, _)) = [[ ]] : 'a list list |
    ttgen(1, (a:'a,b:'a)) = [[a],[b]] |
    ttgen(d, (a:'a,b:'a)) = ttgen_ex((a,b), ttgen(d-1, (a,b)));
```

SML

```
fun conditional [false, false] = true | conditional [false, true] = true
  | conditional [true, false] = false | conditional [true, true] = true;
fun disjunction [false, false] = false | disjunction [false, true] = true
  | disjunction [true, false] = true | disjunction [true, true] = true;
fun isValid([ ], c, p, q) = c[p, q]
  | isValid(pr::ps, c, p, q) = if pr[p, q]
     then isValid(ps, c, p, q) else true;
fun isTautology([ ]) = true
  | isTautology(x::xs) = x andalso isTautology(xs);
```

```
val conditional = fn: bool list -> bool
val disjunction = fn: bool list -> bool
val isValid = fn: ('a list -> bool) list * ('a list -> bool) * 'a * 'a -> bool
val isTautology = fn: bool list -> bool
```

Algorithm to determine if an argument is valid

- Identify the premises and the conclusion of the argument
- Contruct a truth table including premises and conclusion
- Find all rows in which premises are true
- If in each of the true rows, the conclusion is true then argument is valid
  - i.e. a tautology

ProofPower rules of inference

SML

$val\ premise = asm\_rule;$
$val\ modus\_ponens = \Rightarrow\_elim;$
$val\ modus\_tollens = modus\_tollens\_rule;$
$val\ disjunctive\_addition = \vee\_right\_intro;$
$val\ conjunctive\_addition = \wedge\_intro;$
$val\ conjunctive\_simplification = \wedge\_right\_elim;$
$val\ disjunctive\_syllogism = \vee\_cancel\_rule;$
$val\ hypothetical\_syllogism = \Rightarrow\_trans\_rule;$
$val\ double\_negation = \neg\_\neg\_elim;$

```
val premise = fn: TERM -> THM
:) val modus_ponens = fn: THM -> THM -> THM
:) val modus_tollens = fn: THM -> THM -> THM
:) val disjunctive_addtion = fn: TERM -> THM -> THM
:) val conjunctive_addtion = fn: THM -> THM -> THM
:) val conjunctive_simplification = fn: THM -> THM
:) val disjunctive_syllogism = fn: THM -> THM -> THM
:) val hypothetical_syllogism = fn: THM -> THM -> THM
:) val double_negation = fn: THM -> THM
```

CS113 LAB 4 - Example 4.3

p	q	$(p \to q)$	$(q \to p)$	$((p \to q) \wedge (q \to p))$	$(p \vee q)$	$(((p \to q) \wedge (q \to p)) \to (p \vee q))$
T	T	T	T	T	T	T
T	F	F	T	F	T	T
F	T	T	F	F	T	T
F	F	T	T	T	F	F

$$\begin{array}{l} p \to q \\ q \to p \\ \hline \therefore p \vee q \end{array}$$

SML

$fun\ premise1[p,\ q] = conditional[p,\ q];$
$fun\ premise2[p,\ q] = conditional[q,\ p];$
$fun\ conclusion[p,\ q] = disjunction[p,\ q];$
$fun\ v[p,\ q] = isValid([premise1, premise2],\ conclusion,\ p,\ q);$
$isTautology(map\ v\ (ttgen(2,\ (true,\ false))));$

```
val premise1 = fn: bool list -> bool
```

```
val premise2 = fn: 'a list -> 'a
```

```
val conclusion = fn: 'a list -> 'a
```

```
val v = fn: bool list -> bool
```

```
val it = false: bool
```

CS113 LAB 4 - Example 4.4

p	q	$(p \to q)$	$((p \to q) \wedge p)$	$(((p \to q) \wedge p) \to q)$
T	T	T	T	T
T	F	F	F	T
F	T	T	F	T
F	F	T	F	T

$$\begin{array}{l} p \to q \\ q \\ \hline \therefore q \end{array}$$

```
val premise1 = fn: bool list -> bool
val premise2 = fn: 'a list -> 'a
val conclusion = fn: 'a list -> 'a
val v = fn: bool list -> bool
:) val it = true: bool
```

SML

fun premise1[p, q] = *conditional*[p, q];
fun premise2[p, q] = p;
fun conclusion[p, q] = q;
fun v[p, q] = *isValid*([*premise1*,*premise2*], *conclusion*, p, q);
isTautology(*map v* (*ttgen*(2, (*true*, *false*))));
"*Proof Power*"
val premise = *asm_rule*;
val modus_ponens = $\Rightarrow$*_elim*;
val p1 = *premise* ⌜ $p \Rightarrow q$ ⌝;
val p2 = *premise* ⌜ p:*BOOL* ⌝;
modus_ponens p1 p2;

This argument is **valid**
(This is Modus Ponens)

Proof Power

```
val premise = fn: TERM -> THM
:) val modus_ponens = fn: THM -> THM -> THM
:) val p1 = p ⇒ q ⊢ p ⇒ q: THM
:) val p2 = p ⊢ p: THM
:) val it = p ⇒ q, p ⊢ q: THM
```

p	q	$(p \to q)$	$\sim q$	$((p \to q) \wedge \sim q)$	$\sim p$	$(((p \to q) \wedge \sim q) \to \sim p)$
T	T	T	F	F	F	T
T	F	F	T	F	F	T
F	T	T	F	F	T	T
F	F	T	T	T	T	T

$$\begin{array}{l} p \to q \\ \sim q \\ \hline \therefore \sim p \end{array}$$

SML

$fun\ premise1[p,\ q] = conditional[p,\ q];$

$fun\ premise2[p,\ q] = not\ \ q;$

$fun\ conclusion[p,\ q] = not\ \ p;$

$fun\ v[p,\ q] = isValid([premise1,\ premise2],\ conclusion,\ p,\ q);$

$isTautology(map\ \ v\ \ (ttgen(2,\ (true,\ false))));$

```
val premise1 = fn: bool list -> bool
val premise2 = fn: bool list -> bool
val conclusion = fn: bool list -> bool
val v = fn: bool list -> bool
val it = true: bool
```

This argument is **valid** (This is Modus Tollens)

CS113 LAB 4 - Problem 4.3

Problem 4.3

p	q	r	$\sim p$	$(\sim p \vee q)$	$((\sim p \vee q) \to r)$	$(((\sim p \vee q) \to r) \wedge (\sim p \vee q))$	$((((\sim p \vee q) \to r) \wedge (\sim p \vee q)) \to r)$
T	T	T	F	T	T	T	T
T	T	F	F	T	F	F	T
T	F	T	F	F	T	F	T
T	F	F	F	F	T	F	T
F	T	T	T	T	T	T	T
F	T	F	T	T	F	F	T
F	F	T	T	T	T	T	T
F	F	F	T	T	F	F	T

SML

$val\ premise = asm\_rule;$

$val\ modus\_ponens = \Rightarrow\_elim;$

$val\ p1 = premise\ \ulcorner \neg p \vee q \Rightarrow r \urcorner;$

$val\ p2 = premise\ \ulcorner \neg p \vee q \urcorner;$

$modus\_ponens\ p1\ p2;$

$$\begin{array}{l} \sim p \vee q \to r \\ \sim p \vee q \\ \hline \therefore r \end{array}$$

```
val premise = fn: TERM -> THM
:) val modus_ponens = fn: THM -> THM -> THM
:) val p1 = ¬ p ∨ q ⇒ r ⊢ ¬ p ∨ q ⇒ r: THM
:) val p2 = ¬ p ∨ q ⊢ ¬ p ∨ q: THM
:) val it = ¬ p ∨ q ⇒ r, ¬ p ∨ q ⊢ r: THM
```

This argument is **valid**
(This is Modus Ponens)

CS113 LAB 4 - Problem 4.4

p	q	$(p \to q)$	$((p \to q) \wedge q)$	$(((p \to q) \wedge q) \to p)$
T	T	T	T	T
T	F	F	F	T
F	T	T	T	F
F	F	T	F	T

$$\begin{array}{l} p \to q \\ q \\ \hline \therefore p \end{array}$$

SML

$fun\ premise1[p,\ q] = conditional[p,\ q];$
$fun\ premise2[p,\ q] = q;$
$fun\ conclusion[p,\ q] = p;$
$fun\ v[p,\ q] = isValid([premise1, premise2],\ conclusion,\ p,\ q);$
$isTautology(map\ v\ (ttgen(2,\ (true,\ false))));$

```
val premise1 = fn: bool list -> bool
val premise2 = fn: 'a list -> 'a
val conclusion = fn: 'a list -> 'a
val v = fn: bool list -> bool
val it = false: bool
```

This argument is **invalid**

CS113 LAB 4 - Problem 4.8

p	q	$(p \vee q)$	$\sim p$	$((p \vee q) \wedge \sim p)$	$(((p \vee q) \wedge \sim p) \rightarrow q)$
T	T	T	F	F	T
T	F	T	F	F	T
F	T	T	T	T	T
F	F	F	T	F	T

$$\begin{array}{l} p \vee q \\ \sim p \\ \hline \therefore q \end{array}$$

SML

```
fun premise1[p, q] = disjunction[p, q];
fun premise2[p, q] = not p;
fun conclusion[p, q] = q;
fun v[p, q] = isValid([premise1, premise2], conclusion, p, q);
isTautology(map v (ttgen(2, (true, false))));
```

```
val premise1 = fn: bool list -> bool
val premise2 = fn: bool list -> bool
val conclusion = fn: 'a list -> 'a
val v = fn: bool list -> bool
val it = true: bool
```

This argument is **valid**

p	q	$\sim p$	$(\sim p \to q)$	$\sim q$	$(\sim q \to p)$	$((\sim p \to q) \wedge (\sim q \to p))$	$(\sim p \vee \sim q)$	$(((\sim p \to q) \wedge (\sim q \to p)) \to (\sim p \vee \sim q))$
T	T	F	T	F	T	T	F	F
T	F	F	T	T	T	T	T	T
F	T	T	T	F	T	T	T	T
F	F	T	F	T	F	F	T	T

Premise 1: If Tom is not on team A, then Hua is on team B.
Premise 2: If Hua is not on team B, then Tom is on team A.
Therefore: Tom is not on team A or Hua is not on team B.

$$\begin{array}{c} p \vee q \\ \sim p \\ \hline \therefore q \end{array}$$

SML

$val\ p = "Tom\ is\ on\ team\ A";$
$val\ q = "Hua\ is\ on\ team\ B";$
$fun\ premise1[p,\ q] = conditional[not\ p,\ q];$
$fun\ premise2[p,\ q] = conditional[not\ q,\ p];$
$fun\ conclusion[p,\ q] = disjunction[not\ p,\ not\ q];$
$fun\ v[p,\ q] = isValid([premise1,\ premise2],\ conclusion,\ p,\ q);$
$isTautology(map\ v\ (ttgen(2,\ (true,\ false))));$

```
val premise1 = fn: bool list -> bool
val premise2 = fn: bool list -> bool
val conclusion = fn: bool list -> bool
val v = fn: bool list -> bool
val it = false: bool
```

This argument is **invalid**

CS113 LAB 4 - Problem 4.16

Use Proofpower rules of inference to derive the conclusion.

SML

$val\ L1 = premise \ulcorner \neg p \lor q \Rightarrow r \urcorner;$
$val\ L2 = premise \ulcorner s \lor \neg q \urcorner;$
$val\ L3 = premise \ulcorner \neg t \urcorner;$
$val\ L4 = premise \ulcorner p \Rightarrow t \urcorner;$
$val\ L5 = premise \ulcorner \neg p \land r \Rightarrow \neg s \urcorner;$
$val\ L6 = modus\_tollens\ L4\ L3;$
$val\ L7 = disjunctive\_addition \ulcorner q{:}BOOL \urcorner L6;$
$val\ L8 = modus\_ponens\ L1\ L7;$
$val\ L9 = conjunctive\_addition\ L6\ L8;$
$val\ L10 = modus\_ponens\ L5\ L9;$
$val\ L11 = disjunctive\_syllogism\ L2\ L10;$

1.	$\sim p \lor q \to r$	premise
2.	$s \lor \sim q$	premise
3.	$\sim t$	premise
4.	$p \to t$	premise
5.	$\sim p \land r \to \sim s$	premise
6.	$\sim p$	modus tollens
7.	$\sim p \lor q$	disjunctive addtion 6
8.	r	modus ponens 1 7
9.	$\sim p \land r$	conjunctive addition 6 8
10.	$\sim s$	modus ponens 5 9
11.	$\sim q$	disjunctive syllogism 2 10
	$\therefore \sim q$	conclusion

```
val L1 = ¬ p ∨ q ⇒ r ⊢ ¬ p ∨ q ⇒ r: THM
:) val L2 = s ∨ ¬ q ⊢ s ∨ ¬ q: THM
:) val L3 = ¬ t ⊢ ¬ t: THM
:) val L4 = p ⇒ t ⊢ p ⇒ t: THM
:) val L5 = ¬ p ∧ r ⇒ ¬ s ⊢ ¬ p ∧ r ⇒ ¬ s: THM
:) val L6 = p ⇒ t, ¬ t ⊢ ¬ p: THM
:) val L7 = p ⇒ t, ¬ t ⊢ ¬ p ∨ q: THM
:) val L8 = ¬ p ∨ q ⇒ r, p ⇒ t, ¬ t ⊢ r: THM
:) val L9 = ¬ p ∨ q ⇒ r, p ⇒ t, ¬ t ⊢ ¬ p ∧ r: THM
:) val L10 = ¬ p ∧ r ⇒ ¬ s, ¬ p ∨ q ⇒ r, p ⇒ t, ¬ t ⊢ ¬ s: THM
:) val L11 = s ∨ ¬ q, ¬ p ∧ r ⇒ ¬ s, ¬ p ∨ q ⇒ r, p ⇒ t, ¬ t ⊢ ¬ q: THM
```

Use the valid argument forms of this section to deduce the conclusion from the premises.

SML

$val\ L1 = premise \ulcorner \neg p \Rightarrow r \wedge \neg s \urcorner;$
$val\ L2 = premise \ulcorner t \Rightarrow s \urcorner;$
$val\ L3 = premise \ulcorner u \Rightarrow \neg p \urcorner;$
$val\ L4 = premise \ulcorner \neg w \urcorner;$
$val\ L5 = premise \ulcorner u \vee w \urcorner;$
$val\ L6 = disjunctive\_syllogism\ L5\ L4;$
$val\ L7 = modus\_ponens\ L3\ L6;$
$val\ L8 = modus\_ponens\ L1\ L7;$
$val\ L9 = conjunctive\_simplification\ L8;$
$val\ L10 = modus\_tollens\ L2\ L9;$
$val\ L11 = disjunctive\_addition \ulcorner w{:}BOOL \urcorner L10;$

1.	$\sim p \to r \wedge \sim s$	premise
2.	$t \to s$	premise
3.	$u \to \sim p$	premise
4.	$\sim w$	premise
5.	$u \vee w$	premise
6.	u	disjunctive syllogism
7.	$\sim p$	modus ponens 3 6
8.	$r \wedge \sim s$	modus ponens 1 7
9.	$\sim s$	conjunctive simplification L8
10.	$\sim t$	modus tollens L2 L9
11.	$\sim t \vee w$	disjunctive addition L10
	$\therefore \quad \sim t \vee w$	conclusion

```
:) val L1 = ¬ p ⇒ r ∧ ¬ s ⊢ ¬ p ⇒ r ∧ ¬ s: THM
:) val L2 = t ⇒ s ⊢ t ⇒ s: THM
:) val L3 = u ⇒ ¬ p ⊢ u ⇒ ¬ p: THM
:) val L4 = ¬ w ⊢ ¬ w: THM
:) val L5 = u ∨ w ⊢ u ∨ w: THM
:) val L6 = u ∨ w, ¬ w ⊢ u: THM
:) val L7 = u ⇒ ¬ p, u ∨ w, ¬ w ⊢ ¬ p: THM
:) val L8 = ¬ p ⇒ r ∧ ¬ s, u ⇒ ¬ p, u ∨ w, ¬ w ⊢ r ∧ ¬ s: THM
:) val L9 = ¬ p ⇒ r ∧ ¬ s, u ⇒ ¬ p, u ∨ w, ¬ w ⊢ ¬ s: THM
:) val L10 = t ⇒ s, ¬ p ⇒ r ∧ ¬ s, u ⇒ ¬ p, u ∨ w, ¬ w ⊢ ¬ t: THM
:) val L11 = t ⇒ s, ¬ p ⇒ r ∧ ¬ s, u ⇒ ¬ p, u ∨ w, ¬ w ⊢ ¬ t ∨ w: THM
```

Use the valid argument forms of this section to deduce the conclusion from the premises.

SML

$$\mathit{val}\ \mathit{L1}\ = \mathit{premise}\ \ulcorner\ \neg(p\ \lor\ q)\ \Rightarrow\ r\ \urcorner;$$
$$\mathit{val}\ \mathit{L2}\ = \mathit{premise}\ \ulcorner\ \neg p\ \urcorner;$$
$$\mathit{val}\ \mathit{L3}\ = \mathit{premise}\ \ulcorner\ \neg r\ \urcorner;$$
$$\mathit{val}\ \mathit{L4}\ = \mathit{modus\_tollens}\ \mathit{L1}\ \mathit{L3};$$
$$\mathit{val}\ \mathit{L5}\ = \mathit{double\_negation}\ \mathit{L4};$$
$$\mathit{val}\ \mathit{L6}\ = \mathit{disjunctive\_syllogism}\ \mathit{L5}\ \mathit{L2};$$

1.	$\sim p \lor q \to r$	premise
2.	$\sim p$	premise
3.	$\sim r$	premise
4.	$\sim\sim (p \lor q)$	modus tollens 1 3
5.	$p \lor q$	double-negation 4
6.	q	disjunctive syllogism 2 5
	$\therefore\ \sim s$	conclusion

Notice that ProofPower doesn't apply double-negation automatically, so inserted double-negation manually. (L5)

```
:) val L1 = ¬ (p ∨ q) ⇒ r ⊢ ¬ (p ∨ q) ⇒ r: THM
:) val L2 = ¬ p ⊢ ¬ p: THM
:) val L3 = ¬ r ⊢ ¬ r: THM
:) val L4 = ¬ (p ∨ q) ⇒ r, ¬ r ⊢ ¬ ¬ (p ∨ q): THM
:) val L5 = ¬ (p ∨ q) ⇒ r, ¬ r ⊢ p ∨ q: THM
:) val L6 = ¬ (p ∨ q) ⇒ r, ¬ r, ¬ p ⊢ q: THM
```