# Problem 1

State, with the help of truth tables, whether the following statements in propositional logic are valid or not: The truth tables for the statements are as follows:

1. $((p \to q) \land p) \to q$.

| $p$ | $q$ | $p \to q$ | $(p \to q) \land p$ | $((p \to q) \land p) \to q$ |
|---|---|---|---|---|
| $T$ | $T$ | $T$ | $T$ | $T$ |
| $T$ | $F$ | $F$ | $F$ | $T$ |
| $F$ | $T$ | $T$ | $F$ | $T$ |
| $F$ | $F$ | $T$ | $F$ | $T$ |

2. $((p \to q) \land \neg q) \to \neg p$.

| $p$ | $q$ | $p \to q$ | $\neg q$ | $(p \to q) \land \neg q$ | $\neg p$ | $((p \to q) \land \neg q) \to \neg p$ |
|---|---|---|---|---|---|---|
| $T$ | $T$ | $T$ | $F$ | $F$ | $F$ | $T$ |
| $T$ | $F$ | $F$ | $T$ | $F$ | $F$ | $T$ |
| $F$ | $T$ | $T$ | $F$ | $F$ | $T$ | $T$ |
| $F$ | $F$ | $T$ | $T$ | $T$ | $T$ | $T$ |

3. $((p \to q) \land (p \to \neg q)) \to p$

| $p$ | $q$ | $p \to q$ | $\neg q$ | $p \to \neg q$ | $(p \to q) \land (p \to \neg q)$ | $((p \to q) \land (p \to \neg q)) \to p$ |
|---|---|---|---|---|---|---|
| $T$ | $T$ | $T$ | $F$ | $F$ | $F$ | $T$ |
| $T$ | $F$ | $F$ | $T$ | $T$ | $F$ | $T$ |
| $F$ | $T$ | $T$ | $F$ | $T$ | $T$ | $F$ |
| $F$ | $F$ | $T$ | $T$ | $T$ | $T$ | $F$ |

4. $((p \lor q) \land \neg p) \to q$.

| $p$ | $q$ | $p \lor q$ | $\neg q$ | $(p \lor q) \land \neg q$ | $((p \lor q) \land \neg q) \to q$ |
|---|---|---|---|---|---|
| $T$ | $T$ | $T$ | $F$ | $F$ | $T$ |
| $T$ | $F$ | $T$ | $T$ | $T$ | $T$ |
| $F$ | $T$ | $T$ | $F$ | $F$ | $T$ |
| $F$ | $F$ | $F$ | $T$ | $F$ | $T$ |

Thus, we can see that except the third statement, $((p \to q) \land (p \to \neg q)) \to p$, all the other statements are valid.

# Problem 2

Find an equivalent statements for the following statements, obtained by converting all implications into their equivalences containing solely $\neg$ and $\land$. Ensure that the final result should not have any negations except for direct negations of predicates:

1. $(p \to q) \to r$
   $= \neg(p \to q) \lor r$
   $= \neg(\neg p \lor q) \lor r$
   $= (p \land \neg q) \lor r$

2. $(p \land q) \to p$
   $= \neg(p \land q) \lor p$

$= (\neg p \lor \neg q) \lor p$

3. $\neg p \leftrightarrow q$
$= (\neg p \to q) \land (q \to \neg p)$
$= ((p \lor q) \land ((\neg q \lor \neg p))$

4. $(p \leftrightarrow q) \land p$
$= ((p \to q) \land (q \to p)) \land p$

# Problem 3

Find the negations of the following first order logic statements. The final form should not have any negations except for direct negations of predicates.

1. $\forall p.\forall q.(isOdd(p) \land isOdd(q) \to isOdd(p + q))$
$\neg(\forall p.\forall q.(isOdd(p) \land isOdd(q) \to isOdd(p + q)))$
$= \exists p.\neg\forall q.(isOdd(p) \land isOdd(q) \to isOdd(p + q))$
$= \exists p.\exists q.\neg(isOdd(p) \land isOdd(q) \to isOdd(p + q))$
$= \exists p.\exists q.\neg(\neg(isOdd(p) \land isOdd(q)) \lor isOdd(p + q))$
$= \exists p.\exists q.((isOdd(p) \land isOdd(q)) \land \neg isOdd(p + q))$

2. $\exists S.(Set(S) \land \forall x.x \notin S)$
$\neg(\exists S.(Set(S) \land \forall x.x \notin S))$
$= \forall S.\neg(Set(S) \land \forall x.x \notin S))$
$= \forall S.(\neg Set(S) \lor \neg(\forall x.x \notin S)))$
$= \forall S.(Set(S) \to \neg(\forall x.x \notin S)))$
$= \forall S.(Set(S) \to (\exists x.\neg(x \notin S)))$
$= \forall S.(Set(S) \to (\exists x.(x \in S)))$

# Problem 4

Formalize the english statement using first order logic using the list of first order predicates and functions provided. You can use any first order construct (equality, connectives, quantifiers etc.) but you must only use the predicates, functions and constants provided:

1. Given the predicate

$Natural(x)$, which states that x is a natural number,

the function

$Product(x, y)$, which yeilds the product of x and y,

and the constants 1 and 7, write a statement in first order logic which says "7 is prime".

One possible solution is:

$$\forall p.\forall q.(Natural(p) \land Natural(q) \land Product(p, q) = 7 \to ((p = 1 \land q = 7) \lor (p = 7 \land q = 1))$$

This statement says that if you can find a pair of natural numbers p and q whose product is 7, then either they are 1 and 7, or 7 and 1.

2. Given the predicates

$$Morality(x) \text{ which states that x is a morality,}$$
$$Practiced(x) \text{ which states that x is practiced, and}$$
$$Preached(x) \text{ which states that x is preached,}$$

write a statement in first order logic which states "there are exactly two moralities; one of which is practiced but not preached, and one of which is preached but not practiced" (paraphrased from a quote by Bertrand Russel).

One possible solution is:

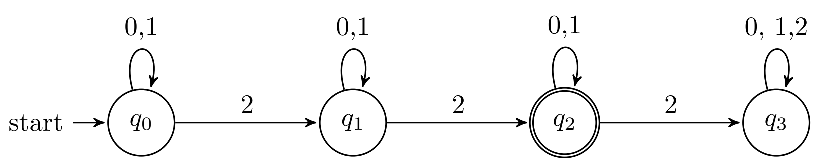
$$\exists p.\exists q.(Morality(p) \land Morality(q) \land Practiced(p) \land \neg Preached(p) \land \neg Practiced(q) \land Preached(q) \land \forall m.(Morality(m) \to m = p \lor m = q))$$

This statement says that there are moralities $p$ and $q$, such that $p$ is practiced and not preached, and $q$ is preached and not practiced, and all moralities are either $p$ or $q$.

# Problem 5

For each of the languages over the indicated alphabets, construct a DFA which accepts precisely those strings that are in the language. Specify the DFA as a state transition diagram:
**We have an online tool that can be used to design, test and submit DFAs for this question**. To use it, visit `https://www.stanford.edu/class/cs103/cgi-bin/nfa/edit.php`. We strongly recommend this tool, as it makes it easy to design, test and submit your solutions. If you submit it via this system, please make a note of it in your homework submission so that we know to look online for your answers.

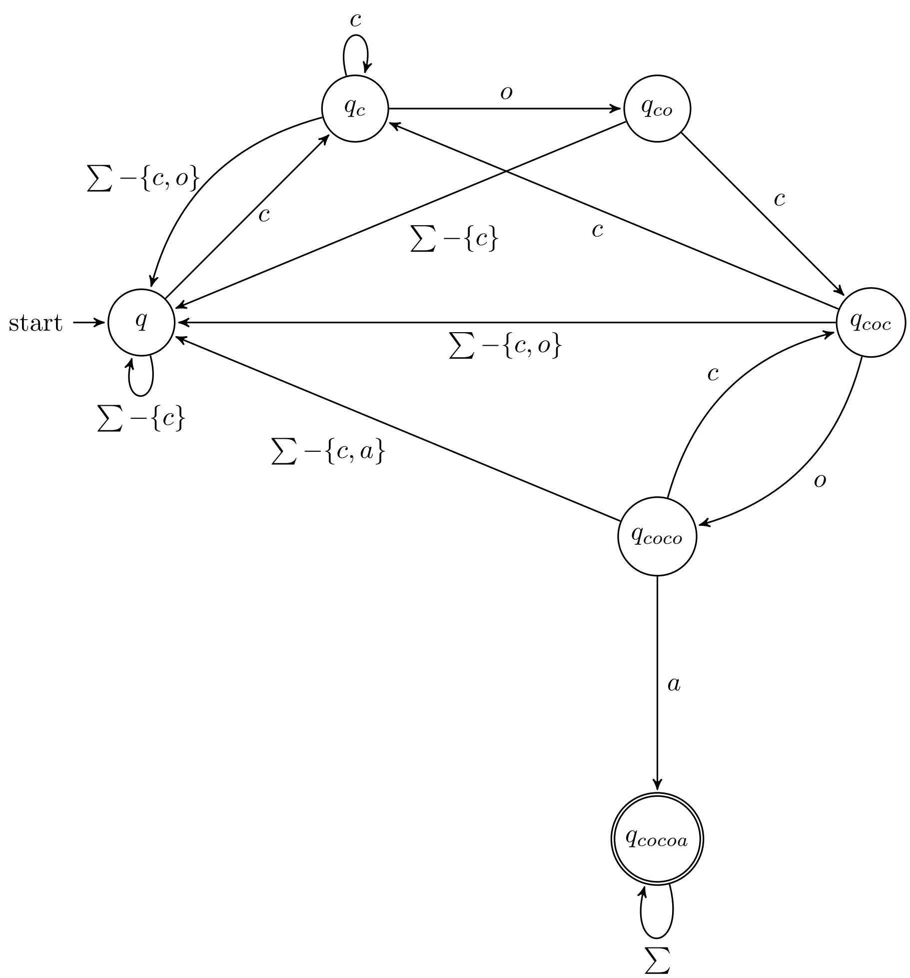
1. For the alphabet $\sum = \{0,\ 1,\ 2\}$, construct a DFA for the language L= $\{w \in \sum^* \mid$ w contains exactly two $2$ s$\}$




Here, each state corresponds to having seen 2 some number of times. Specifically, $q_0$, $q_1$, $q_2$ correspond to having seen 2 zero, one and two times. $q_3$ corresponds to having seen 2 more than once- and hence, is a non accepting state from which it is impossible to recover.

2. For the alphabet $\sum = \{a, b, c...z\}$, construct a DFA for the language L= $\{w \in \sum^* \mid$ w contains the word "*cocoa*" as a substring$\}$. As a shorthand, you can specify multiple letters in a transition by using set operations on $\sum$ (for example $\sum$ - $\{a, b\}$).

*(Hint: Here, notice that the word "cocoa" contains a consecutive and repeating "co", which creates a tricky situation. Specifically, the DFA needs to be able to handle strings which contain more than two consecutive "co"s)*




One thing to notice in this DFA is that if you read a "$c$" in state $q_{coco}$, you do not transition back to state $q_c$. Instead, you transition back to state $q_{coc}$, since the "*coc*" that you have already read might be the real start of the string.