Task 1.1

(a) When you do this, the program is guaranteed to work as intended.

Verification and testing. Without both, it wouldn't be 'guaranteed' to work as intended due to possible bugs, security properties, or not known cases. Verification and testing makes sure the program works correctly and meets specification and makes sure it doesn't break or have bugs.

(b) The code's been in use for 10 years and nobody's seen a bug.

Testing. Basically been 'tested' for 10 years and works with no known bugs.

(c) I have a proof that the code will meet its specification.

Verification. Checks that the code is 'correct' and that it meets specification.

(d) I expect the code to produce an integer, so I wrote it in a statically typed language, annotated the code with the type "int" and it compiled.

Verification. The specification is it produces an integer and it meets that specification.

(e) Doing this on a piece of code can give you more confidence that it will work correctly.

Verification and testing. Formally checking that a program is correct and that it meets specification or testing allows you to make sure that the code will work correctly and have no devastating bugs.

Task 1.2

(a) $p \wedge q \vee r$ $\qquad$ $r \vee q \wedge p$

Semantically equal. Commutative and Associative.

(b) $p \wedge q \vee r$ $\qquad$ $p \wedge r \vee q$

Neither

(c) $\neg p \vee q \to \neg(p \wedge \neg q)$ $\qquad$ $((\neg p) \vee q) \to \neg(p \wedge (\neg q))$

Syntactically equal therefore also semantically equal. Removing the unnecessary parenthesis results in the LHS.

(d) $p \to q$ $\qquad$ $\neg p \to \neg q$

Neither

(e) $p \wedge \neg p$ $\qquad$ F

Semantically equal. $p \wedge \neg p$ is a contradiction.

Task 2.1

(a) $(P \to (Q \to R)) \leftrightarrow ((P \to Q) \to R)$

Contingency. P = Q = R = False results in False. P = Q = False and R = True results in True.

(b) $(P \to Q) \leftrightarrow (\neg Q \to \neg P)$

Tautology

(c) $\neg P \wedge Q \leftrightarrow \neg Q \wedge P$

Contingency. P = Q = False results in True. P = False and Q = True results in False.

Task 2.2

Included in the assignment submission is the file, **tauto.log**, which has the proof for this task.

Task 2.3

In the previous task, we didn't prove that the statement was a tautology by proving $((P \to Q) \wedge \neg P) \vee P \Rightarrow T$ because this says that we already know that the statement logically implies True. We are trying to prove that it is a tautology, so instead, we start with $T \Rightarrow ((P \to Q) \wedge \neg P) \vee P$.

Task 2.4

Included in the assignment submission is the file, **uncurry.log**, which has the proof for this task.

Task 3.1

$\forall x \in A.\exists min, max \in A.min <= x <= max$
For all x in A, there exists a min and a max such that $min <= x <= max$. Thus, A is finite and has a minimum and maximum element.

Task 3.2

(a) $\forall x \in \mathbb{Z}.\exists y \in \mathbb{Z}.x = y * 2$

False. With x = 3, y has to be 1.5, but y can only be an integer. Thus the statement is false.

(b) $\exists x \in \mathbb{Z}.\neg(\exists a \in \mathbb{Z}.\exists b \in \mathbb{Z}.a > 1 \wedge b > 1 \wedge a * b = x)$
$\exists x \in \mathbb{Z}.\forall a \in \mathbb{Z}.\forall b \in \mathbb{Z}.a < 1 \vee b < 1 \vee a * b \neq x$

True. By simplifying and using DeMorgan's Law, the statement is easier to understand. We can use x = 3, a = 0, and b = -2 as a witness.

(c) $\forall x \in \mathbb{Z}.\exists y \in \mathbb{Z}.y = x * 2$

True. With $x \in \mathbb{Z}$, then $x * 2 \in \mathbb{Z}$. Thus, the statement is true, $\exists y \in \mathbb{Z}.y = x * 2$.

Task 3.3

Included in the assignment submission is the file, **pred.log**, which has the proof for this task.

Task 4.1

I spent about 5 hours of actual working time on this homework. Though that is mostly because I had to tinker around the proof checker, relearn LaTeX, and review certain things. Even then, some of the answers, I am still unsure of.