Due Wednesday, September 26, by 4:00pm, to Crowdmark.

All submitted work must be the student's own.

Question 1 (6 marks).
A **full binary tree** is one of the following:

- A node.
- A node with two children, both of which are also full binary trees that share no nodes.

Define $n(T)$ to be the number of nodes in a full binary tree T, $\ell(T)$ be the number of leaves of T (nodes with no children) and let $e(T)$ be the number of edges of T (connections between full binary trees).

Use structural induction to show that every full binary tree T has $e(T) = 2(n(T) - \ell(T))$.

Question 2 (6 marks).

This exercise uses the following definition.

> A **substitution** S is a function from propositional variables to formulas. We apply a substitution to a formula φ by simultaneously replacing each variable p in φ by the formula $S(p)$. Formally, we define $S(\varphi)$ by induction:
>
> - If φ is a propositional variable p, then $S(\varphi)$ is the formula $S(p)$, or simply p if $S(p)$ is undefined.
> - If φ is $(\neg\eta)$, then $S(\varphi)$ is $\big(\neg S(\eta)\big)$.
> - If φ is $(\eta \star \zeta)$ for a binary connective $\star$, then $S(\varphi)$ is the formula $(S(\eta) \star S(\zeta))$.
>
> For example, if $S(p) = (q \wedge r)$ and $S(q) = (p \vee q)$ and $S(r)$ is undefined, then
>
> $$S\big(\,(p \to (q \to r))\,\big) \text{ is } \big((q \wedge r) \to \big((p \vee q) \to r\big)\big)\ .$$

Fix an arbitrary substitution S, and an arbitrary valuation t. Prove that there is a valuation u such that for every formula φ, the value of φ under u is the same as the value of $S(\varphi)$ under t: in symbols, $\varphi^u = S(\varphi)^t$. Use structural induction on φ.

(Hint: your answer must specify how the valuation u depends on S and t. If you cannot see how to do that at first, try simply starting the proof without specifying u. Then ask yourself: what properties of u do you need in order to make the required arguments?)

Question 3 (4 marks).

Let p, q, r and s be Propositional variables. Let α be the propositional formula

$$\big((p \vee r) \to ((\neg q) \leftrightarrow (s \wedge p))\big).$$

[2] (a) Draw the parse tree for the formula α.

[2] (b) Determine α^t for the truth valuation t given below. Show how you arrived at your answer.

$$\begin{aligned} t(p) &= \texttt{F}, \\ t(q) &= \texttt{F}, \\ t(r) &= \texttt{T}, \text{ and} \\ t(s) &= \texttt{F}. \end{aligned}$$

Question 4 (9 marks).
In each part, use the specified method!

[3] (a) Determine using a valuation tree (consider the propositional variables in alphabetical order) if α is a tautology, satisfiable and/or a contradiction where α is

$$\Big(\big(((\neg p) \vee q) \wedge (r \to q)\big) \leftrightarrow \big(((\neg p) \vee q) \wedge (q \vee (\neg r))\big)\Big)$$

Circle one selection per row below:

Tautology	**Not a Tautology**
Contradiction	**Not a Contradiction**
Satisfiable	**Not Satisfiable**

[3] (b) Determine using logical equivalences (equivalence laws) if β is a tautology, satisfiable and/or a contradiction where β is

$$\Big(\big((\neg(p \to r)) \to (\neg q)\big) \wedge \big((p \wedge q) \wedge (\neg r)\big)\Big).$$

(In other words, simplify the above equation first using logical equivalences as much as you can and then draw your conclusions. Cite the rules you are using on each line, one per line.) Circle one selection per row below:

Tautology	**Not a Tautology**
Contradiction	**Not a Contradiction**
Satisfiable	**Not Satisfiable**

[3] (c) Determine using a truth table if γ is a tautology, satisfiable and/or a contradiction where γ is

$$\Big(\big((\neg((\neg q) \vee (p \wedge r))) \wedge (p \rightarrow (q \wedge r))\big) \wedge \big(r \rightarrow ((\neg q) \vee p)\big)\Big)$$

Circle one selection per row below:

Tautology	**Not a Tautology**
Contradiction	**Not a Contradiction**
Satisfiable	**Not Satisfiable**

Question 5 (4 marks).
In the code fragment given below, there are three blocks of code: B_1, B_2, B_3 and B_4.

```
if ( p || q ) {
    if ( q || r ) {
        B1
    } else {
        if ( !(p && r) ) {
            B2
        } else {
            B3
        }
    }
} else {
    B4
}
```

Determine which of B_1, B_2, B_3 or B_4 are dead blocks of code (i.e. code that does not get executed no matter what the values of p, q and r are). If the block is not dead, give explicit truth values for p, q and r that will reach that code. If the code is dead, justify formally that it cannot be reached.