# *EECS 755 - Final Exam*

*Spring Semester 2024*

*May 6 2024*

**Exercise 1** *Which of the following statements are true? (1pt each)*

1. *The* desruct *tactic implements proof-by-cases.*
2. *The* Inductive *construct defines a set of all elements of a type.*
3. *The* Definition *construct can define a recursive function.*
4. *A nonterminating tactical introduces an inconsistency in Coq.*
5. *A* tactical *takes* tactics *as arguments.*
6. *t1 ; t2 is a tactical that applies t1 and subsequently applies t2 to* the first *goal resulting from t1.*
7. *try t tries tactic t and if it fails returns the proof goal to its previous state.*
8. *The inversion implements backwards reasoning while apply implements forwardss reasoning*
9. *True is a proposition with exactly one proof.*
10. *not A unfolds to (not A)->False*
11. *split takes a goal of the form A<->B and produces two separate goals A->B and B->A.*
12. *In the Hoare triple {{Q}}c{{P}} specifies that Q must be true before c executes for P to hold afterwards.*
13. *state is a mapping from variable names to their values.*
14. *In* The Adventures of Buckaroo Bonzai *John Lithgow plays a mad scientist trying to return to the $8^{th}$ dimension using a device called an overthruster.*

**Exercise 2** *In this exercise we will be thinking command definition. You will define a new do-until loop that executes its body and then checks its termination condition. Unlike while which checks before. You will also be thinking about specifying read-only memory.*

1. *Extend the inductive proposition definition of IMP to define do c until b*
2. *Can you extend the eval function for IMP to include do c until b? If so, do it. If not explain why not.*

3. *What is the difference between IMP* com *expressions and the* aexp *and* bexp *expressions? Specifically, what is the difference between how they are evaluated?*

4. *Most languages split their states to isolate read-only values from values that may change. Using Coq, specify a new* state *that does this.*

5. *How would we prove that an IMP program does not attempt to modify its read-only values?*

**Exercise 3** *In this exercise we're going to think a bit more about the* until *loop and equivalence with* while*. Also looking at proof commands for reasoning about language constructions.*

1. *Using* while *define a construct that is equivalent to* until *from the previous problem.*

2. *What theorem should you prove to show this new* until *and your* while *construction are equivalent.*

3. *When reasoning about some program of the form* `c1 ; c2` *in the assumptions of a proof, why might* inversion *or be good options? Are we reasoning forwards or backwards?*

4. *When reasoning about some program of the form* `c1 ; c2` *as he goal of a proof, why would* eapply E_Seq *be a good option? Are we reasoning forwards or backwards?*

5. *Assume that we've defined a new, swanky optimizer for IMP called* swanky *that inputs an IMP program and returns an optimized IMP program. What theorem would you prove to show the optimizer is correct?*

**Exercise 4** *Finally we're going to think a bit about Hoare Logic. We're going to push a bit here and think about concurrency and what that might mean.*

1. *Briefly explain what the notation* `{{X=m}}X:=X+1{{X=m+1}}` *means?*

2. *If we have some command defined by* `{{True}}c{{Y=n AND X=m}}`*, can we evaluate* `c;X:=X+1`*? Why or why not? (Your answer need not be formal.)*

3. *Let's define a new command* `c1 || c2` *that behaves like sequence except that* `c1` *and* `c2` *execute in parallel.* `c1` *and* `c2` *start in the same state and should end in a state that reflects execution of both commands. Define an inference rule for this command and a Hoare Logic rule.*

4. *Let's say that* `c1 || c2` *is serializable if its execution result is the same no matter who goes first. Can you capture serializability using Hoard Logic? Specifically, how would you prove that* `c1 || c2` *is serializable?*

5. *Let's say that* `c1` *and* `c2` *interfere with each other when the state resulting from* `c1 || c2` *is inconsistent. Can you define interference using Hoare Logic? Specifically, how would yo prove that* `c1 || c2` *exhibits interference?*