

CS 536: Quiz 1 (30 minutes) Solution

Instructions

The quiz is closed book, closed notes, no support equipment (calculators, phones, computers, etc). All the questions are short-answer questions. The usual penalty for copying or sharing answers on a quiz or exam is a final grade of E for the course. If you have any questions, please ask during the quiz, not after. **Please write your name & id on the top of each sheet.** (I plan to unstaple the quiz for grading.) **You DO NOT have to fill all available space with answers.**

Questions

1. [10 points] Give two examples of situations where reasoning about program correctness is at least as important as program testing.

(1) Testing is too expensive or dangerous; (2) We can't look at a solution and figure out if it's right; (other examples....)

2. [15 points] Let σ_0 be the memory state shown. Give a written definition for σ_0 . (I.e., $\sigma_0(\dots) = \dots$)

$\sigma_0(b)(0) = 7$, $\sigma_0(b)(1) = 3$, $\sigma_0(b)(2) = 8$, $\sigma_0(x) = 14$,

$\sigma_0(y) = \text{true}$

3. [15 points] Say s_0 is $b[\text{if } x-b[0] > 2 \text{ then } 2 \text{ else } x-b[0] \text{ fi}]$. Calculate the value of s_0 in the state σ_0 of Question 2.

Write out your work in notation. (E.g., " $\sigma_0(s_0)$ ", not "the value of s_0 in σ_0 .")

[10 points] Say $\sigma_2(x) = 2$. Which one of $\sigma_2[z := 4]$ and $\sigma_2[z := x+x]$ uses incorrect notation?

Briefly (in a sentence or two) explain why.

- (3a) Let's first evaluate the test of the conditional expression.

$$\begin{aligned} \sigma_0(x-b[0] > 2) &= \sigma_0(x-b[0]) > \sigma_0(2) \\ &= \sigma_0(x) - \sigma_0(b[0]) > 2 \\ &= 14 - \sigma_0(b)(\sigma_0(0)) > 2 \\ &= 14 - \sigma_0(b)(0) > 2 = 14 - 7 > 2 = \text{true} \end{aligned}$$

So now we can calculate

$$\begin{aligned} \sigma_0(s_0) &= \sigma_0(b[\text{if } x-b[0] > 2 \text{ then } 2 \text{ else } x-b[0] \text{ fi}]) \\ &= \sigma_0(b)(\sigma_0(\text{if } x-b[0] > 2 \text{ then } 2 \text{ else } x-b[0] \text{ fi})) \\ &= \sigma_0(b)(\sigma_0(2)) \quad [\text{the true branch}] \text{ because } \sigma_0(x-b[0] > 2) = \text{true} \\ &= 8 \end{aligned}$$

- (3b) $\sigma_2[z := x+x]$ is incorrect because you can only update a state with a value, not with an expression. (If we wanted "the value of $x+x$ ", we should have written $\sigma_2[z := \sigma_2(x+x)]$.)

	0	1	2
b	7	3	8
x	14		
y	true		

4. [10 points] Translate the following into English: “ $\tau_2 = \tau_1[x := \tau_1(y)]$ where $\tau_1 = \tau_0[y := 8]$.” [10 points] Calculate $\sigma_1[p := 6][q := 7][p := 8](p+q)$, where σ_1 maps p and q to 4 and 5 respectively; show each step of the calculation.

(4a) In English, “ $\tau_2 = \tau_1[x := \tau_1(y)]$ where $\tau_1 = \tau_0[y := 8]$ ” means (some variation of) “ τ_2 is state τ_1 updated at x with the value of y in τ_1 , where τ_1 is state τ_0 updated at y with 8.”

(4b)
$$\begin{aligned} \sigma_1[p := 6][q := 7][p := 8](p+q) &= \sigma_1[p := 6][q := 7][p := 8](p) + \sigma_1[p := 6][q := 7][p := 8](q) \\ &= 8 + \sigma_1[p := 6][q := 7](q) \\ &= 8 + 7 = 15 \end{aligned}$$

(Note: It doesn't matter what σ_1 maps p or q to, nor is the update of p to 6 relevant — they're overridden with the updates of q to 4 and p to 8.)

Propositional Logic Rules

<p>Associativity</p> <ul style="list-style-type: none"> • $(p \vee q) \vee r \leftrightarrow p \vee (q \vee r)$ • $(p \wedge q) \wedge r \leftrightarrow p \wedge (q \wedge r)$ <p>Distributivity</p> <ul style="list-style-type: none"> • $(p \vee q) \wedge r \leftrightarrow (p \wedge r) \vee (q \wedge r)$ • $(p \wedge q) \vee r \leftrightarrow (p \vee r) \wedge (q \vee r)$ <p>Transitivity</p> <ul style="list-style-type: none"> • $(p \rightarrow q) \wedge (q \rightarrow r) \rightarrow (p \rightarrow r)$ • $(p \leftrightarrow q) \wedge (q \leftrightarrow r) \rightarrow (p \leftrightarrow r)$ <p>Double negation (Pierce's Law)</p> <ul style="list-style-type: none"> • $\neg\neg p \leftrightarrow p$ 	<p>Excluded middle</p> <ul style="list-style-type: none"> • $p \vee \neg p \leftrightarrow T$ <p>Identity</p> <ul style="list-style-type: none"> • $p \wedge T \leftrightarrow p$ • $p \vee F \leftrightarrow p$ <p>Domination</p> <ul style="list-style-type: none"> • $p \vee T \leftrightarrow T$ • $p \wedge F \leftrightarrow F$ <p>Idempotency</p> <ul style="list-style-type: none"> • $p \vee p \leftrightarrow p$ • $p \wedge p \leftrightarrow p$ 	<p>Definition of \rightarrow and \leftrightarrow</p> <ul style="list-style-type: none"> • $(p \rightarrow q) \leftrightarrow (\neg p \vee q)$ • $(p \leftrightarrow q) \leftrightarrow (p \rightarrow q) \wedge (q \rightarrow p)$ <p>DeMorgan's Laws</p> <ul style="list-style-type: none"> • $\neg(p \wedge q) \leftrightarrow (\neg p \vee \neg q)$ • $\neg(p \vee q) \leftrightarrow (\neg p \wedge \neg q)$ <p>Commutativity</p> <ul style="list-style-type: none"> • $p \vee q \leftrightarrow q \vee p$ • $p \wedge q \leftrightarrow q \wedge p$ • $(p \leftrightarrow q) \leftrightarrow (q \leftrightarrow p)$ <p>Contradiction: $p \wedge \neg p \leftrightarrow F$</p>
---	---	---

5. [15 points] Give a formal proof that $p \wedge q \rightarrow p$ is a tautology, using the rules above.

$p \wedge q \rightarrow p$
 iff $\neg(p \wedge q) \vee p$ Defn \rightarrow
 iff $(\neg p \vee \neg q) \vee p$ DeMorgan's law
 iff $(\neg p \vee p) \vee \neg q$ Associativity & commutativity [optional step]
 iff $T \vee \neg q$ Excluded middle
 iff T Domination [optional]

6. [15 points] Use the notation of predicate calculus to define a predicate $Split(b, m, n)$ that means “Every value in $b[0..m]$ is $<$ every value in $b[m+1..n-1]$.” Note: the values don't have to be sorted. E.g., if $b[0..3]$ are 5, 3, 7, 6 then $Split(b, 1, 4)$ is true.

A couple of possibilities:

$Split(b, m, n) \equiv \forall j. 0 \leq j \leq m \rightarrow \forall k. m < k < n \rightarrow b[j] < b[k]$
 $Split(b, m, n) \equiv \forall j. \forall k. 0 \leq j \leq m \wedge m < k < n \rightarrow b[j] < b[k]$