CSC324 Course Notes Errata

David Liu
March 29, 2015

If you find any typos or errors in the course notes, please let me know so that I can add them to the list!

1. (pg. 11, Taher) Course overview first paragraph: “… macros, which give the programmer the ability to add new syntax…”

2. (pg. 24, Ian) Bottom: “Because recursion is extremely common in functional programming…”

3. (pg. 24, Anon.) Tail-recursive sum: arguments to sum-helper should be switched.

4. (pg. 26, Jasmin) Second paragraph: “delay” instead of “delays”

5. (pg. 29, Jasmin) Variable shadowing: “the ability to bind a name”

6. (pg. 31, Jasmin) Top: “we see how the (+ x y) becomes (+ 10 y)”

7. (pg. 33, Jasmin) Bottom: “But what happens when…”

8. (pg. 34, Jasmin) Bottom: “And it is ideas like that which motivate research…”

9. (pg. 35, Jasmin) Top: “A common beginner mistake when creating these functions…”

10. (pg. 40, Jasmin) Margin note: “The macro now treats for and in as pattern variables”

11. (pg. 42, Ian) Middle: “Let us write a macro…”

12. (pg. 44, Jason) Class macro: x should be attr in both places.

13. (pg. 49, Jasmin) Second comment: “Note that the last line shows…”

14. (pg. 50, Jasmin) Bottom: “Not only does this work with…”

15. (pg. 53, Jasmin) Top: “we need to look at Haskell’s unusual evaluation order.”

16. (pg. 54, Jason) Code at top: should be foldl’, not foldl.

17. (pg. 54, Anon.) Middle: take should take the integer and then the list; occurs multiple times

18. (pg. 56, Jasmin) 4th note: “try running this program yourself!”

19. (pg. 57, Jasmin) Middle: “if we have compile-time guarantees about the types of all expressions for the duration of the program’s run…”

20. (pg. 60, Jasmin) Bottom: “This is quite suggestive: somehow Haskell interprets (&&) as a…”

21. (pg. 61, Jasmin) ((&&) True) is then applied to True

22. (pg. 62, Jasmin) Bottom: “Haskell lists must contain elements…”

23. (pg. 65, Jasmin) Top: “…then we expect that functions which operate on values of type A…”

24. (pg. 65, Dylan) Middle: the method f of class B is overridden, not overloaded as the comment says.

25. (pg. 69, Jasmin) Middle: “Most built-in types in Haskell are members…” Side note: “One notable exception is…”
26. (pg. 70, Jasmin) Bottom: “By creating different **types** which implement...”

27. (pg. 73, Jasmin) Top: “Now let us return to the original goal”

28. (pg. 81, Felix) **sumOfStack** definition, first line should be **sumOfStack** [] = (0, []) (tuple missing).

29. (pg. 87, Jasmin) Middle: “A Prolog “program” will consist of two parts: facts and rules, which **encode**...”

30. (pg. 89, Jasmin) Top: “A predicate in a query will be called a goal, which can either succeed **or** fail.”

31. (pg. 92, Jasmin) Exercise break: “may require” is duplicated

32. (pg. 94, Jasmin) Top: “We are **not** defining a function that returns a boolean value...”

33. (pg. 99, Jasmin) Middle: “I want the left side to succeed: the rule tells me it does when...”

34. (pg. 100, Anon.) Last code block: change line 5 to `f(a) :- student(david).` A rule with a false right-side is required for the example to make sense.

35. (pg. 102, Anon.) First code block: change the query on line 6 to `happy(lily).`

36. (pg. 102, Jasmin) Middle: “with the idea that **once** a query is completely finished...including choice points of **completed** subqueries.”

37. (pg. 103/104, Jasmin) “1. This is unified with the rule on line 4... 3. Backtracking! Line 5 is now chosen...”

38. (pg. 111, Jasmin) Top: “The following queries **work** just fine...”