Programming Languages (COP 4020/6021) [Spring 2018]

Assignment II

Objectives

1. To understand and use higher-order functions, including map and fold.
2. To continue practicing writing functional programs, using ML features like Currying.
3. To gain experience with deductive systems, inference rules, and proving properties of judgments by induction on their derivations.

Due Date: Monday, February 5, 2018 (at 5pm).
Late submission: You may submit any part (or both parts) of this assignment late (i.e., between 5pm on 2/5 and 5pm on 2/7) with a 15% penalty on the whole assignment.

Machine Details: Complete this assignment by yourself, the programming portion on the following CSEE network computers: c4lab01, c4lab02, ..., c4lab20. These machines are physically located in ENB 220. You can connect to the C4 machines from home using SSH. (Example: Host name: *c4lab01.csee.usf.edu* Login ID and Password: <your NetID username and password>) You are responsible for ensuring that your programs compile and execute properly on these machines.

Assignment Description

(0) Read Sections 5.1, 5.3-5.6, and 6.1-6.3 of the *Elements of ML Programming* textbook.

(1) **Programming portion:** Create a file called *as2.sml*. In that file, re-implement the functions from Assignment I, without defining any extra top-level functions, other values, or types. Note: The `ipowerset` and `product` functions are each worth 5% extra credit for undergraduates.

For this second assignment, your functions must:

a) Never call built-in/library functions, *except* `map`, `foldr`, and `foldl`.
b) Never be recursive. All recursion must occur within `map` and `fold`s.
c) Never contain let- or local-expressions. The keyword "let" (and similarly, "local") should never appear in *as2.sml*.
d) Never cause side effects (such as I/O or pointer/array operations) to occur.
e) Match the types shown below (*note that arguments must be Curried*).
f) Be commented and formatted properly; as part of this restriction, use spaces instead of tabs and limit line widths to 100 characters.
g) Use ML constructs like pattern matching and anonymous variables when appropriate.
h) Compile and execute on the C4 machines with no errors or warnings.
i) Not be significantly more complicated than necessary.
j) Make the same data-structure-invariant assumptions as on Assignment I (e.g., assume that incoming set arguments are valid and guarantee that returned sets are valid).

Function Types

```
val size = fn : int * int list -> int
val elementof = fn : int -> int * int list -> bool
val subset = fn : int * int list -> int * int list -> bool
val same = fn : int * int list -> int * int list -> bool
val union = fn : int * int list -> int * int list -> int * int list
val lunion = fn : (int * int list) list -> int * int list
val intersection = fn : int * int list -> int * int list -> int * int list
val lintersection = fn : (int * int list) list -> int * int list
val powerset = fn : int * int list -> (int * int list) list
val ipowerset = fn : (int * int list) list -> (int * int list) option
val product = fn : (int * int list) list -> int list list
```

Hints

You can test your implementations on the examples shown in the Assignment I handout, though for this assignment arguments need to be curried. As always, we'll test your submissions on examples not shown in the handouts.

Programming-portion Submission Reminders

- Type the following pledge as an initial comment in your *as2.sml* file: "I pledge my Honor that I have not cheated, and will not cheat, on this assignment." Type your name after the pledge. Not including this pledge will lower your grade 50%.
- Upload and submit your *as2.sml* file in Canvas.
- You may submit your assignment in Canvas as many times as you like; we'll grade your latest submission.

(2) **Theory portion:**

(a) Define a deductive system having three judgment forms:

i. Judgments of the form *L list* are valid iff *L* is a list of natural numbers. For example, *nil list* and *Z::S(S(Z))::nil list* are valid judgments.

ii. Judgments of the form $N \in L$ are valid iff natural-number *N* appears in list *L*.

iii. Judgments of the form $L_1 \subseteq L_2$ are valid iff all the elements in list L_1 are also in list L_2.

Your rules may assume that only natural numbers will ever be used as list elements, so your rules never need to contain judgments of the form *N nat*. The inference rules for deriving *N nat* are irrelevant for this assignment.

(b) Using your definitions from (a) above, formally prove that the $\subseteq$ relation is *transitive*:

Theorem. $\forall\, L_1, L_2,$ and L_3: ($(L_1 \subseteq L_2 \wedge L_2 \subseteq L_3) \Rightarrow (L_1 \subseteq L_3)$)

Note: This theorem may be challenging to prove at times. Please do your best with it; partial credit is possible. If you get stuck at some point, just explain informally whatever ideas you're having trouble stating formally.

Theory-portion Submission Reminders

- Write the following pledge at the end of your submission: "I pledge my Honor that I have not cheated, and will not cheat, on this assignment." Sign your name after the pledge. Not including this pledge will lower your grade 50%.
- For full credit, turn in a hardcopy (handwritten or printed) version of your solutions.
- Late submissions may be emailed or submitted in hardcopy.
- All emailed submissions, even if sent before the deadline, will be graded as if they were submitted late, i.e., with a 15% penalty.
- If you think there's a chance you'll be absent or late for class on the date this assignment is due, you're welcome to submit solutions early by giving them to me or the TA before or after class, or during any of our office hours.