Homework 1: Lectures 1 & 2

CS 440: Programming Languages and Translators, Spring 2020

Due Fri Jan 24, 11:59 pm

1/21 p.1

How to submit

See http://cs.iit.edu/~cs440 → Homework Policies for information on working with others, how to submit, etc. If you want to submit multiple files, zip them together and submit the zipped file.

Problems [50 points]

1. [15 = 5 * 3 pts] In ghci, what happens on the inputs below? Give results or briefly describe the error(s). ("Brief" = Don't rewrite or rephrase the error message; translate/rephrase it into a short bit of English.)

   a. `sin (cos pi)`

   b. `cos -1`

   c. `sin cos pi`

   d. `(sqrt . head [sqrt]) 16.0` (Recall: infix dot is function composition)

2. [9 = 3 * 3 pts]. What do you get if you delete all the extra and the problematic parentheses from the expressions below? I.e., if an expression doesn't compile in ghci, figure out which parentheses need to be removed to fix the problem. In any case, drop redundant parentheses to get to the minimum set needed.

   a. `(cos(sqrt(2.5))+((sin)(pi)))(*)(2)`

   b. `((:) (('a' : ("b")) ++ "cd")) ( ( [ ( ['c'] ) ++ "(d)"] ) )`

   c. `( [ ( [ ( [ 17 ] ) ] ) ] : ( [ ( [ ] ) ] ) )`

3. [4 pts] Rewrite the expression below so that it uses prefix functions throughout. (Hint: (+).)

   `((a + b) * c) / (d ^e) -- ^ is exponentiation`

4. [4 pts]. Rewrite the following expression so that it uses infix notation where possible [1/21]. (Hint: `` `name` ``.)

   `f (g x (h a b)) (c (d e f))`

5. [5 pts] Complete the following function definition so that on any list that can be tested for == [1/21], `f` returns True.

   `f x = x == [x !! i ???]`

6. [5 pts]. Complete the following function definition: `stutter n x` should return a list of length `n` where each element is `x`. E.g., `stutter 3 5` = `[5,5,5]`. Use a list comprehension to produce the result. Note `stutter 0 x` = `[]`. Don't worry about `n` < 0; the canonical solution makes stutter return `[]`.


```
stutter n x = [ ???1 | ???2 <- ???3 ]
```

(Hints: What do we want for $???_1$? We need a list for the $???_3$; what should its length be? Do the values of $???_3$ matter?)

7. [11 pts]. Let `g` be the list defined below.

```
> rot x = last x : init x
> g = [1,3,5] : [rot x | x <- g]
```

Use referential transparency to calculate/show the pattern of the values for `take n g` for `n` = 0, 1, 2, …. Don't just show the results of the expressions: The point of the problem is to explain how the calculations work/are related. If you want, you can use a couple of properties:

`take (m+1) g = head g : take m (tail g)`

and

`take (m+1) g = take m g ++ [`*e*`]` where *e* is the expression for the last element of `take (m+1) g`.

