

Programming Languages G22.2110 Summer 2007 hw03

Assigned Th 5/31/2007, due We 6/6/2007 at 1pm.

How to Submit Homework Assignments

Email your answers, in either plain text format or as pdf, to Abhijit Guria <guria@cs.nyu.edu>. Assignments are due on Wednesdays at 1pm. This deadline will be strictly enforced.

Reading Assignments

- For lecture on 5/31/2007: Scott 10.0, 10.2, 10.3.0-10.3.4; Candolin (<http://www.cs.hut.fi/Studies/T-93.210/schemetutorial/>)
 - For lecture on 6/7/2007: Scott 3.2, 3.3.0-3.3.3, 3.3.6
-

Homework Assignments

1. Functional rewriting (7 points)

Consider the following Scheme definition for function r:

```
(define (r ls) (if (null? ls) ls (append (r (cdr ls)) (list (car ls)))))
```

Indicate which function or special form was applied for each of the 17 rewrite steps below.

```
(r '(1 2))
1 => (if (null? '(1 2)) '(1 2) (append (r (cdr '(1 2))) (list (car '(1 2)))))
2 => (if #f '(1 2) (append (r (cdr '(1 2))) (list (car '(1 2)))))
3 => (append (r (cdr '(1 2))) (list (car '(1 2))))
4 => (append (r '(2)) (list (car '(1 2))))
5 => (append (r '(2)) (list 1))
6 => (append (r '(2)) '(1))
7 => (append (if (null? '(2)) '(2) (append (r (cdr '(2))) (list (car '(2))))) '(1))
8 => (append (if #f '(2) (append (r (cdr '(2))) (list (car '(2))))) '(1))
9 => (append (append (r (cdr '(2))) (list (car '(2)))) '(1))
```

```

10 => (append (append (r '()) (list (car '(2)))) '()) '()) '()
11 => (append (append (r '()) (list 2)) '()) '()
12 => (append (append (r '()) '()) '()) '()
13 => (append (append (if (null? '()) '()) (append (r (cdr '())) (list (car '())))) '()) '()
14 => (append (append (if #t '()) (append (r (cdr '())) (list (car '())))) '()) '()
15 => (append (append '() '()) '()) '()
16 => (append '() '()) '()
17 => '()

```

2. Functional rewriting (13 points)

Consider the following Scheme definition for function `e2`:

```
(define (e2 n) (if (= 0 n) 1 (* 2 (e2 (- n 1)))))
```

Show how Scheme evaluates the expression `(e2 2)` by rewriting it similar to Question 1 above (you should end up with 13 rewrite steps).

3. Scheme (8 points)

While you write the Scheme code for answering Question 4 below, you will probably get some error messages. Describe two error messages using the following format:

- Code: *a very short piece of code that triggers the error*
- Symptom: *the error message itself*
- Cause: *an explanation for what triggered the error message*
- Solution: *how to fix the code to prevent the error*

4. Scheme (22 = 5 + 5 + 5 + 7 points)

Write Scheme programs exercising fundamental features. Adopt a functional programming style by avoiding assignments, favoring `let`-bindings with nested bodies instead. Use only functions that are listed in Section 6 (Standard procedures) of the Revised(5) Report on Scheme. These should be available in both DrScheme (language level R5RS) and guile.

4a. I/O (5 points)

Write a program that prompts the user for his or her name, reads the name from input, then politely greets the user by name. Here is an example interactive session:

```

Please type your name surrounded by double-quotes.
"Bob"
Hello, Bob, nice to meet you!

```

4b. Libraries (5 points)

Write a program that uses Scheme library functions to compute $\sqrt{2}$, $\sin(3.5)$, and $e^{2.5}$, and then prints the results like this (don't worry if the decimals aren't 100% identical):

```
square root of 2.0:    1.4142135623730951
sine of 3.5:          -0.35078322768961984
e to the power of 2.5: 12.182493960703473
```

4c. Types (5 points)

The following code creates a variable `b` with the boolean value `#t`, and then prints a description and the value of the variable:

```
(let ((b #t))
  (display "name b, type boolean, value ") (display b) (display "\n"))
```

Extend this program by creating and printing more variables of different types. Your program should produce the following output:

```
name b, type boolean, value #t
name c, type char, value Z
name i, type int, value 42
name l, type list, value (1 4 9 16)
name r, type real, value 3.141
name str, type string, value hello
name sym, type symbol, value *
```

4d. Control flow (7 points)

Write a Scheme function `count-occurrences` that takes two parameters, a string and a character, and returns the number of occurrences of the character in the string. For example, `(count-occurrences "hello" #\l)` should return 2. Do not use a loop, use recursion. Hint: you can use `string->list` and write a helper function that counts occurrences of a character in a list of characters.

5. Python (0 points)

Practice your Python skills by writing some more code. For example, you could solve problems from other classes in Python. If you need inspiration, take a look at the Examples section on the class web page, which lists extension ideas for the code you read for earlier homeworks.

<http://www.cs.nyu.edu/courses/summer07/G22.2110-001/hw03.pdf>

Total points: 50.