



Exam Review

Corky Cartwright
Department of Computer Science
Rice University



What Is Emphasized?

Design recipe (with minimal testing)

Programming with functions. Using **map**. Using **foldr**. Practice writing functions using **map** and **foldr**.

Example: **cons-all**

Functional abstraction. Inclusion of **map**, **foldr** in Scheme library. What is motivation for including **map**. How do we get **foldr** from the structural recursion template for lists?



An Example from HW3

Write

```
cross: (list-of symbol) (list-of number) ->
      (list-of (pair-of symbol number))
```

Template instantiation:

```
(define (cross los lon)
  (cond [(empty? los) ...]
        [else ... (first los) ...
                  (cross (rest los) lon) ...]))
```



What Is Mentioned

- Hand evaluation
- What expressions are values?
 - Atomic constants
 - Symbols
 - Numbers
 - Strings
 - **true false empty**
 - Constructors applied to values
 - Lambda-expressions



Glitch in HTDP

- In absence of lambda-expressions, defined function names must be classified as values.
- Example:

```
(define (double x) (+ x x))  
(define freezing 32)
```
- What is `double` in

```
(map double ' (1 2 3))
```
- What is `freezing` in

```
(map double (list freezing))
```

But no gotcha questions on this issue on exam.



Glitch in HTDP cont.

- With lambda-expressions, this glitch can be eliminated by interpreting
`(define (f x1 ... xn) M)`
as an abbreviation for
`(define f (lambda (x1 ... xn) M))`
- Over a decade ago, we taught `lambda` notation in `define` statements from the beginning of Comp 210 for this reason!
- On the exam, we will accept either convention regarding treating function names as values on the exam, but be consistent!



Glitch in HTDP cont.

- In the absence of data structures containing functions, where does this issue arise? When evaluating function applications. Given the definition of `double`, what is the next step in evaluating (reducing)

`(double 4)`

- According to the book (prior to Ch. 38),

`(double 4) => (+ 4 4)`

- But what if `double` is not a value?

`(double 4)`

`=> ((lambda (x) (+ x x)) 4)`

`=> (+ 4 4)`



What Is Peripheral But Useful

- Demand-driven evaluation by wrapping an expression in `(lambda () ...)` called a “thunk” or “suspension”.
This material only appears in the Extra Credit problem.
This language construction is not supported in DrScheme until the **Advanced Student** level, which is a superset of the **Intermediate Student with lambda** level.

Example: simulating Scheme `or`



Prior to Taking Exam

- Prepare for exam
- Work on Optional HW 6?
- Reading:
 - Review all assigned reading with an emphasis on functions as data.