Student Name: _____

CS 2500/Accelerated Exam 0—Fall 2017

Matthias Felleisen

September 20, 2017

- The exam is a **one-hour** exam.
- We will not answer any questions during the exam. If you believe a problem statement is ambiguous, choose *any* non-trivial interpretation.
- Write down the answers in the space provided, including the back of the given spaces.
- You may use the paper copy of the book or your notes.
- You may *not* use any electronic gadgets (for example, watches, google glasses, phones, tablets, laptops). Any use of an electronic gadget will lead to immediate expulsion from the exam and class.
- You may use all the definitions, expressions, and functions found BSL, especially those suggested in hints. Define everything else.
- Some basic test taking advice: Before you start answering any problems, read *every* problem, so your brain can be thinking about the harder problems in the background while you knock off the easy ones.

Problem	Max. Points
1	5
2	10
3	15
Total	/ 30

Problem 1 Design the function `adder`, which consumes a `Posn` and computes the sum of its fields. Show **all** steps of the design recipe.

Problem 2

```
(define-struct counting (time-left))
(define-struct going (x y))
; A CD is one of:
; -- "hold"
; -- (make-counting PositiveNumber)
; -- (make-going Number Number)
; interpretation It represents the state of a
; rocket-launch count down.
```

Design the function `cd-as-string`. It consumes a CD and forms a sentence that describes the current state of a countdown. The sentence starts with `"The countdown is "` and continues with an appropriate word, depending on which state is given. For example, when given a `going` structure, the function may add `"over."` When a `counting` structure is given, the sentence must involve the number in the `time-left` field.

intentionally left blank

Problem 3 Design the function `inflate`. It consumes a menu item and an inflation rate. Its result is a menu item whose price has been inflated at the given rate. The restaurant that you are working for represents menu items as follows:

```
(define MP "market price")

(define-struct item (title price))
; A MenuItem is (make-menu String Price)

; A Price is one of:
; -- Number
; -- MP
```

Domain knowledge *Inflation* is the rate at which the price for certain items increases over time. Typical items include phones, cars, food, fuel, and work. If an economy experiences a 10% inflation and a restaurant wishes to keep up with it, it must raise its prices by 10%.

(If prices trend downward, economists speak of a *deflation*.)

intentionally left blank