Student Number: |_|_|_|_|_|_|_|_|_|_|_|

Last (Family) Name(s): _____

First (Given) Name(s): _____

Please read the instructions below *carefully*.

- Due: Mon Feb 20th 5PM.
- This homework consists of two exercises. Exercise 1s on 5 pages (including this one) and a second exercise in a Racket file.
- Answer the first question in the Racket file provided and the second question directly on a paper.

Marking Guide

Nº 1: _____/ 7

Nº 2: _____/13

TOTAL: _____/20

Question 1. [7 marks]

Part (a) [7 marks]

Define `X` to produce the cartesian product of two lists.
Write your answer in the Racket file attached. Half of the marks are for using higher-order functions to minimize the use of recursion.
You may write helper functions.

```
(check-equal? (X '(a b c) '(d e f))
              '((a d) (a e) (a f)
                (b d) (b e) (b f)
                (c d) (c e) (c f)))
```

Question 2. [13 marks]

Consider a simple imperitive programming language. The syntax of this programming langauge is similar to C. The expression and statement syntax is shown as follows:

```
e := e_1 + e_2  | e_1 - e_2  | e_1 * e_2  | e_1 / e_2
   | e_1 == e_2 | e_1 && e_2 | e_1 || e_2 | ! e_1
   | v          | c

s := v = e | s_1; s_2 | if (e) s_1 else s_2

e, e_1, e_2 ∈ expr
s, s_1, s_2 ∈ stmt
v ∈ var
c ∈ ℤ
```

The language has simple arithmetic operations, equal comparison, and logical operations including "&&" (and), "||" (or), and "!" (not). Like other imperitive programming languages, it has assignment statements and if statements. Multiple statements will be separated by ";" and be executed sequentially. This langauge has only integer values. Similar to C, it uses zero to represent false and non-zero to represent true for boolean conditions and operations.

To formally define the semantics of the programming language, we use the notation we introduced in the class. Specifically, a program in the langauge operates with an execution environment $E = v \to n, \ldots$ defined as a list of entries saperated by comas. Each entry binds a variable to an integer. We use notation $E \vdash e \downarrow n$ to denote that under the enviroment E, an expression e in the language will evaluate to the integer value n. We use notation $E \vdash s \Downarrow E'$ to denote under the environment E, the execution of the statement s will produce a new environment E'.

We can write down the operational semantics for variable reference, integer constant, addition, substraction, multiplication, variable assignment, and sequential execution as the following inference rules:

$$\frac{n \in \mathbb{Z}}{E \vdash n \downarrow n} \qquad \frac{E(v) = n}{E \vdash v \downarrow n} \qquad \frac{E \vdash e_1 \downarrow n_1 \qquad E \vdash e_2 \downarrow n_2}{E \vdash e_1 + e_2 \downarrow n_1 + n_2} \qquad \frac{E \vdash e_1 \downarrow n_1 \qquad E \vdash e_2 \downarrow n_2}{E \vdash e_1 - e_2 \downarrow n_1 - n_2}$$

$$\frac{E \vdash e_1 \downarrow n_1 \qquad E \vdash e_2 \downarrow n_2}{E \vdash e_1 * e_2 \downarrow n_1 * n_2}$$

$$\frac{E \vdash e \downarrow n_1 \qquad E' = v \to n_1, E}{E \vdash v = e \Downarrow E'} \qquad \frac{E \vdash s_1 \Downarrow E' \qquad E' \vdash s_2 \Downarrow E''}{E \vdash s_1; s_2 \Downarrow E''}$$

Part (a) [2 marks]

The operational semantics for the integer division is shown as the following rule:

$$\frac{E \vdash e_1 \downarrow n_1 \qquad E \vdash e_2 \downarrow n_2 \qquad n_2 \neq 0}{E \vdash e_1 / e_2 \downarrow \lfloor n_1 / n_2 \rfloor}$$

Explain why we have the condition $n_2 \neq 0$. How can we handle the divide-by-zero runtime error case with our semantics if we have only this rule for the division?

Because we define e1 divided by e2 to evaluate as n1 divided by n2, to avoid dividing by 0, we need the condition to ensure that n2, the divisor, is not equal to the value 0.

We can use the exception handling mechanism to handle the divide-by-zero error case. In the defintion of integer division, we can catch the case when n2=0 and terminate the program when n2=0.

Part (b) [4 marks]

The following rules define the operational semantics for the equal comparison, the boolean not operator, and the boolean and operator:

$$\frac{E \vdash e_1 \downarrow n_1 \qquad E \vdash e_2 \downarrow n_2 \qquad n_1 = n_2}{E \vdash e_1 == e_2 \downarrow 1} \qquad \frac{E \vdash e_1 \downarrow n_1 \qquad E \vdash e_2 \downarrow n_2 \qquad n_1 \neq n_2}{E \vdash e_1 == e_2 \downarrow 0} \qquad \frac{E \vdash e \downarrow n_1 \qquad n_1 = 0}{E \vdash !e \downarrow 1}$$

$$\frac{E \vdash e \downarrow n_1 \qquad n_1 \neq 0}{E \vdash !e \downarrow 0} \qquad \frac{E \vdash e_1 \downarrow n_1 \qquad n_1 = 0}{E \vdash e_1 \,\&\&\, e_2 \downarrow 0} \qquad \frac{E \vdash e_1 \downarrow n_1 \qquad E \vdash e_2 \downarrow n_2 \qquad n_1 \neq 0 \qquad n2 = 0}{E \vdash e_1 \,\&\&\, e_2 \downarrow 0}$$

$$\frac{E \vdash e_1 \downarrow n_1 \qquad E \vdash e_2 \downarrow n_2 \qquad n_1 \neq 0 \qquad n2 \neq 0}{E \vdash e_1 \,\&\&\, e_2 \downarrow 1}$$

Similar to C, our programming language will treat input operands with zero as false and non-zero as true. But the result of boolean operations will be either zero or one, e.g., in our language 2&&3 would produce the result of 1.

Please write down inference rules to define the semantics of the boolean or operator. Note that both the boolean or and the boolean and operators should be short-circuited.

$$\frac{E \vdash e1 \downarrow n1 \qquad n1 = 1}{E \vdash e1 \,||\, e2 \downarrow 1}$$

$$\frac{E \vdash e1 \downarrow n1 \qquad E \vdash e2 \downarrow n2 \qquad n1 = 0 \qquad n2 = 1}{E \vdash e1 \,||\, e2 \downarrow 1}$$

$$\frac{E \vdash e1 \downarrow n1 \qquad E \vdash e2 \downarrow n2 \qquad n1 = 0 \qquad n2 = 0}{E \vdash e1 \,||\, e2 \downarrow 0}$$

Part (c) [3 marks]

The following rule defines the operational semantics of if statement in the case of the condition being false (i.e., zero):

$$\frac{E \vdash e \downarrow n \qquad E \vdash s_2 \Downarrow E' \qquad n = 0}{E \vdash if\ (e)\, s_1\ else\, s_2 \Downarrow E'}$$

Write down the inference rule to define the semantics of if statement in the case of the condition being true:

$$\frac{E \vdash e \downarrow n \qquad E \vdash s1 \Downarrow E' \qquad n \neq 0}{E \vdash \text{if (e)s1 else s2} \Downarrow E'}$$

Part (d) [4 marks]

Consider the following program in our language:

```
x = 1;
y = x * 2 + 10;
if (y && x)
  z = x
else
  z = 1
```

Suppose the program environment before executing the above code snippet is E_0; The environment after executing the first assignment statement is E_1; The environmnet after executing the second assignment statement is E_2; The environment after executing the last if statement is E_3. Suppose $E_0 = \varnothing$ is an empty list. Write down the value of E_1, E_2, and E_3.

E1 = x —> 1, E0
E1 = x —> 1

E2 = y —> 2 * x + 10, E1
E2 = y —> 2 * x + 10, x —> 1
E2 = y —> 13, x —> 1

E3 = if (y && x) z—>x else z—>1 , E2
E3 = z—>1, y —> 13, x —> 1

Total Marks = 20