CSE 1729 - Introduction to Principles of Programming	October 3, 2017
Preliminary Examination 1 - Review	

Note: most of these involve writing or evaluating code, so you can do them in Scheme – however, you will not have a computer during the exam. The structure of today's lab will be

1. For the first hour, do the code-writing questions on paper. Think first, then try to make them as correct as possible.
2. For the rest of the lab, enter your solutions into DrRacket and see whether they are correct. Debug them if not.

Since the exam is tomorrow, you are not required to turn in your solutions to these tasks. My suggestion: work through as many as possible in lab, use the others for drill later.

1. Define a simple function to compute the following:
   (a) Define a function `USD-to-Bitcoin` which converts U.S. dollars to Bitcoins. Use $4407.25 per Bitcoin as the conversion rate.
   (b) Define a function `energy-from-mass` which uses Einstein's equation $E = mc^2$ to compute the energy equivalent of a mass m. Your function should take a single argument (m, the mass in kg) and return mc^2, where $c = 299,792,458\frac{m}{s^2}$. Use the `let` form to define the constant c. (Incidentally, with this choice of units, you will be computing the energy in Joules and your mass would be measured in kilograms.)
   (c) Define a function which takes the base and perpendicular height of a triangle and computes the area of that triangle (recall that the area is equal to $\frac{1}{2}b \times h$).
   (d) The SuperFresh supermarket chain needs a program to determine the value of all of the change in a cash register drawer. Define a function that, given the number of pennies, nickels, dimes and quarters, will calculate the total value of the change in the drawer. (Note that your function should take 4 arguments).
2. Define a recursive function to compute the following:
   (a) Observe that a repeating decimal can be calculated to an arbitrary number of decimal places using a recursive function. Write a function that takes one parameter, n, and calculates

   $$0.\bar{1} = 0.1111\ldots$$

   to n decimal places. Thus, your function, when called with 3, should return 0.111.

   Note: $\sum_{i=1}^{\infty} \frac{1}{10^i} = 0.\bar{1}$. This is the decimal expansion of the fraction 1/9.
   (b) To amplify the previous problem, define a function that computes

   $$1/22 = .0\overline{45} = .0454545\ldots$$

   to n decimal digits of accuracy. How can you figure out if you should produce a 4 or a 5 in a given recursive step?

   (Hint: There are a few ways to proceed. One is to use the fact that `(even? n)` returns true exactly when n is even. Another is to define a helper function with an argument that tells it(self) whether the next digit should be a 4 or a 5.) For practice, you might try to develop both.
   (c) Let f and g be two functions, mapping integers to integers. Define a function `dominate` so that

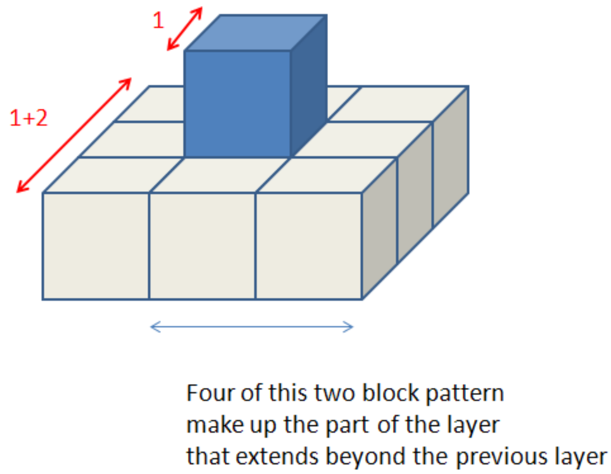

Figure 1: A pyramid of height 2

```
(dominate f g)
```

evaluates to the smallest positive integer k for which $f(k) > g(k)$.

(Thus, if $f(1) > g(1)$, `(dominate f g)` should evaluate to 1. If $f(1) \leq g(1)$ but $f(2) > g(2)$, then `(dominate f g)` should evaluate to 2, etc.)

(d) Consider the construction of a 3-dimensional pyramid using square blocks. A pyramid of height one is constructed with a single block; a pyramid of height two is constructed as shown in Figure 1, by building a foundation of 9 blocks and placing a pyramid of height one on the center.

In general, we build a pyramid of height k in two steps:

i. Build a pyramid of height $k - 1$.

ii. Build a square foundation that's large enough so that when the pyramid of height $k - 1$ is placed on top, the foundation protrudes by one box on each side.

The base of a pyramid of height 1 is 1×1; the base of a pyramid height 2 is 3×3. What should the base of a pyramid of height k be?

One you have figured this out, define a Scheme function that computes the *total* number of blocks in a pyramid of height k.

3. Higher-order functions:

(a) In lecture and in lab, you have seen a higher order function that will compute the summation of n terms of a function, f. Here is one for the product $f(0) \times f(1) \times \cdots f(n)$:

```
(define (product f n)
    (if (= n 0)
        (f 0)
        (* (f n) (product f (- n 1)))))
```

i. Use this `product` function to define a function that computes $n!$

ii. A function related to the factorial function is the product of all of the odd positive integers up to some odd integer n. This function is often referred to as *odd double factorial*. For an odd positive integer $n = 2k - 1, k \geq 1$, odd double factorial is defined as $(2k-1)!! = \prod_{i=1}^{k}(2i-1)$. Define a Scheme function `odd-double-factorial` that uses `product`. Note: `odd-double-factorial` of an even number should be 0.

(b) In mathematics, function composition involves applying a function to the results of another function, i.e. $f(g(x))$, and is denoted $f \circ g$.

   i. Define a function that takes two parameters, two functions of one parameter (f and g) and computes a function that is equal to the composition of f and g.

   ii. Test your function with $f(x) = 5x - 4$ and $g(x) = x^2 + 3$.

(c) Let f be a function that maps integers to integers. (The integers are the numbers $\{\ldots, -2, -1, 0, 1, 2, \ldots\}$.) For such a function, define Δf to be the function

$$\Delta f(x) = f(x+1) - f(x)\,.$$

(Note that, given f, Δf is a *function*, not a number.) Define a Scheme program which, given a function f, returns the function Δf. (The process of transforming f to Δf is sometimes called "taking the discrete derivative.")

This process has some interesting properties. Define the (degree one) polynomial $\ell(x) = 16x + 4$. Compute the discrete derivative using your function above and evaluate the resulting function at a few places (or plot it). What do you notice?

Now define the function $q(x) = 11x^2 - 4x + 11$, a quadratic function. Take the discrete derivative *twice* to yield the function

$$\Delta(\Delta(q))\,.$$

As above, evaluate this new function at a few places (or plot it). What do you notice?

4. Write a Scheme expression to evaluate:

   (a) $(11 + 22) * (44 - 33)$

   (b) $(4 + 8 + 15 + 16 + 23 + 42)$. Hint: How many operands can the + operator accept?

   (c) $((3 - 2)/4 + 5 * (6 - 7))$

   (d) $3 * (4 - 2^{15}) + 6$

   (e) 2^{256}, without using `expt`. (Hint: Of course, a possible solution is

```
(* 2 2 2 ... 2)
```

   with 256 arguments. However, there are many more elegant solutions. One way to proceed is to suppose that you have found an expression `<E>` that evaluates to 2^{128}. Then the following expression evaluates to 2^{256}:

```
(let ((x128 <E>))
  (* x128 x128))
```

   Now, you can use this as the seed of a simple expression for 2^{256}.)

5. What is the value of the following Scheme expression? (solve these without the computer first, as you will not have a computer during the exam):

   (a) `(- 15 5)`

   (b) `(+ (/ 1 2) (/ 1 3))`

   (c) `(* 18 8 10)`

   (d) `(- (+ 10 4) (* 4 1))`

   (e) `(* (+ 2 2) (/ (* (+ 3 5) (/ 30 10)) 2))`