CSE 1729 - Introduction to Principles of Programming — Due: January 31, 2022

Problem Set 1

1. Re-write the following arithmetic expressions as SCHEME expressions and show the result of the SCHEME interpreter when invoked on your expressions.
   (a) $(22+42) \times (54 \times 99)$.
   (b) $((22+42) \times 54) \times 99$.
   (c) $64 \times 102 + 16 \times (44/22)$.
   (d) Is the expression in the following limerick, written by recreational mathematician James Mercer, correct?

      A dozen, a gross, and a score
      Plus three times the square root of four
      Divided by seven
      Plus five times eleven
      Is nine squared and not a bit more

      That is, define a variable, named `limerick`, bound to the result of evaluating the following SCHEME expression:

      $$\frac{12+144+20+3\sqrt{4}}{7} + (5 \times 11)$$

      You can check the validity of this limerick by evaluating the expression

      ```
      limerick
      ```

      Note: You can use the built-in SCHEME function `sqrt` to compute the square root of 4 as follows: `(sqrt 4)`

2. Reflect on the expressions above.
   (a) Of course, the first two expressions evaluate to the same number. In what sense are they different? How is this reflected in the SCHEME expression?
   (b) In an unparenthesized infix arithmetic expression, like $3+4*5$, we rely on a *convention* to determine which operation we apply first (rules of precedence). Are rules of precedence necessary for arithmetic operations SCHEME ?

3. Write SCHEME definitions for the functions below. Use the interpreter to try them out on a couple of test cases to check that they work.
   (a) `inc`, the function $\texttt{inc}(x) = x+1$.
   (b) `inc2`, the function $\texttt{inc2}(x) = x+2$. Show how to write `inc2` using your definition of `inc` instead of $+$.
   (c) `cube`, the function $\texttt{cube}(x) = x^3$.
   (d) `p`, the polynomial function $\texttt{p}(x) = (x^5 + 16x^4 + 22x^3 + x + 9)^2$.
   (e) Using the function `cube`, write the function $\texttt{ninth}(x) = x^9$.
   (f) Using the function `ninth`, write the function $\texttt{eighty-first}(x) = x^{81}$. Recall that $81 = 9 \times 9$. (You may want to test this on an input relatively close to 1, such as 1.01.)

Remark SCHEME provides built-in support for exponentiation (via the `expt` function, defined so that `(expt x y)` yields x^y). For the exercises above, however, please construct the functions $x \mapsto x^k$ using only $*$ and function application.

4. Reflect on your definition of `eighty-first` above. What would have been the difficulty of defining this merely in terms of $*$?

5. An International Standard Book Number is a ten-digit or thirteen-digit number meant to uniquely identify commercially sold books. For ISBN-10 numbers, the first nine digits encode information such as the publisher and the title of the book. The tenth digit is a check-digit and is used to find typographical errors or errors in the transmission of an ISBN-10 number. Given nine digits representing the information related to a particular book,

$$x_{10}, x_9, x_8, x_7, x_6, x_5, x_4, x_3, x_2$$

determine the check-digit which completes the ISBN-10 number for that book. Start by computing the following sum:

$$10 \cdot x_{10} + 9 \cdot x_9 + 8 \cdot x_8 + 7 \cdot x_7 + 6 \cdot x_6 + 5 \cdot x_5 + 4 \cdot x_4 + 3 \cdot x_3 + 2 \cdot x_2$$

One useful notation for representing the sum of terms such as these in a concise way is sigma notation. For instance, the same sum expressed in sigma notation is

$$\sum_{i=2}^{10} i \cdot x_i$$

The sigma character indicates we will sum these terms. The index we will use for the terms is i and the range of values we will sum over is the integers from 2 to 10. Finally, each term will be of the form $i \cdot x_i$. We will be using sigma notation for some of the problems in this course.

To find the check-digit, we need to find the smallest value we can add to the sum above which gives a result that is a multiple of 11. SCHEME provides the `modulo` function which evaluates to the remainder of integer division. For example:

```
> (modulo 112 11)
2
```

If we take the sum above modulo 11, we get the remainder of division by 11. We need the value to add to this to get to the next multiple of 11. So, we can just subtract this value from 11. Unfortunately, there is a problem with this solution. If the sum above, let's call it s, happens to be a multiple of 11 already, we should get 0 for the check-digit. However, `(modulo s 11)` will evaluate to 0 and $11-0$ is 11. We can correct this by using the modulo operation one more time.

If you like, you can view the very first Numberphile video on YouTube which describes 10 digit ISBN numbers here. It starts by talking about the date the video was released (11-11-11). But, eventually, they do describe ISBN-10 numbers.

(a) Write a SCHEME function `(isbn10-checkdigit x10 x9 x8 x7 x6 x5 x4 x3 x2)` that takes the nine digits corresponding to an ISBN-10 number and computes the value of the corresponding check-digit.

(b) We can also check that a series of digits encodes a valid ISBN-10 number. Write a SCHEME function, named `(is-isbn10? x10 x9 x8 x7 x6 x5 x4 x3 x2 x1)`, which evaluates to `#t` if the digits form a valid ISBN-10 number, and `#f` otherwise.

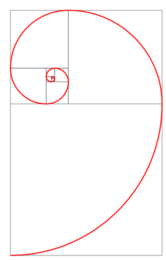
Remember DRY (Don't Repeat Yourself). Your solution to this part should not duplicate any of the code you wrote in part a. In fact, you should apply the function you wrote in part a to determine if the supplied digits correspond to a valid ISBN-10 number.

6. The Fibonacci spiral, shown below, is a rather remarkable mathematical object which mysteriously appears in a number of places in biology. It is described by the polar equation

$$r(\theta) = \phi^{\theta \cdot (2/\pi)}.$$

Here ϕ is a constant called the *golden ratio* which you may approximate by 1.618. Likewise π is the familiar constant which you may approximate by 3.142. In order to carry out the exponentiation, please use the built-in function `expt` defined so that `(expt x y)` returns x^y. Using these, write a SCHEME function `fspiral` so that `(fspiral theta)` returns the value $r($`theta`$)$.

You may enjoy watching the three-part series of videos by Vi Hart which start here



7. Remember the *Quadratic formula*, which can be used to find the roots of a quadratic eguation? For a quadratic equation $ax^2 + bx + c = 0, a \neq 0$, the formula is

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Notice that this gives us two different roots (because of the $\pm$) whenever $b^2 - 4ac \neq 0$.

Write the following SCHEME functions.

(a) `(root1 a b c)` that gives us the root corresponding to the plus in the $\pm$ in the quadratic formula (that is, calculate $\frac{-b+\sqrt{b^2-4ac}}{2a}$).

(b) `(root2 a b c)` that gives us the root corresponding to the minus in the $\pm$ in the quadratic formula (that is, calculate $\frac{-b-\sqrt{b^2-4ac}}{2a}$).

(c) `(number-of-roots a b c)` which calculates the number of distinct roots to the equation $ax^2 + bx + c = 0, a \neq 0$ (which will either be 1 or 2).

(d) `(real-roots? a b c)` is a boolean function that evaluates to `#t` when the roots of $ax^2 + bx + c = 0, a \neq 0$ are real numbers. Note that you do not have to calculate the roots to determine whether they are real or complex numbers.

Note: there are some common calculations done in the above functions; it may make sense to write some auxiliary functions to make your code simpler.