CSE1729 - Introduction to Programming October 7, 2020

Prelim 1

This is a 50 minute exam. The only reference material you may use during the exam, are the course textbook or notes. You may not use any additional resources (websites, Google, discussion boards, assistance from other humans) during the exam. *Please read every question carefully before answering* and express your answers as neatly and completely as you can.

There are 4 questions on the exam, each worth 10 points; **you may choose *any three* to complete**. Thus, the highest score that can be achieved on the exam is 30 points. Please indicate which 3 questions you wish to be graded in the check boxes next to the question numbers below.

Name: ______________________________ Section: __________

Question	Points	Grade?	Score
1	10		
2	10		
3	10		
4	10		
Total	30		

1. Expressions and Functions

   (a) [2 points] Write a SCHEME expression which defines a variable named `pi` and uses the following approximation as a value for the variable:

$$\pi \approx \left(228 + \frac{16}{1329}\right)^{\frac{1}{41}} + 2$$

   (b) [2 points] Define a scheme function, named `(in-hours h m s)`, which accepts a measured time in hours, h, minutes, m, and seconds, s, and returns the length of time in hours only. Recall, a minute is $\frac{1}{60}$ of an hour and a second is $\frac{1}{60}$ of a minute.

   (c) [3 points] Define a SCHEME function `crazy` which takes two arguments: a function, f, and a number, x. Called with f and x, the function should return

$$\frac{f(x+10) + f(x+10) + 10}{f(x+10)}.$$

   (For full credit, your function `crazy` should only call f *once* on $x + 10$. Use a `let` statement!)

   (d) [3 points] Define a SCHEME function, named `(num-digits n)`, which accepts an integer, n, as an argument and computes the number of digits in the integer. Note: you did this in Lab 2.

2. A Munchhausen number is a number in which the sum of all the individual digits raised to themselves equals the original number.
For example, 3435 is a Munchhausen number because

$$3^3 + 4^4 + 3^3 + 5^5 = 3435$$

These numbers are named this way as it evokes the story of Baron Munchausen raising himself up by his own ponytail because each digit is raised to the power of itself.

(a) [2 points] It so happens that the expression 0^0 is undefined. If we assume $0^0 = 0$, then more integers satisfy this definition of Munchausen numbers. Unfortunately, the SCHEME builtin function `expt` returns the value 1 for 0^0. Define a SCHEME function, named `(pow b e)`, which computes b^e with 0^0 defined to be 0. That is, define the power function, but your function should evaluate to 0 when called with 0 as both the base and the exponent. So,

```
> (pow 0 0)
0
```

(b) [8 points] Create a function `(munchausen? n)` that returns `#t` if the integer n is a Munchhausen number and `#f` otherwise. That is, your function should return `#t` when the sum of all the digits, raised to themselves, is equal to n.

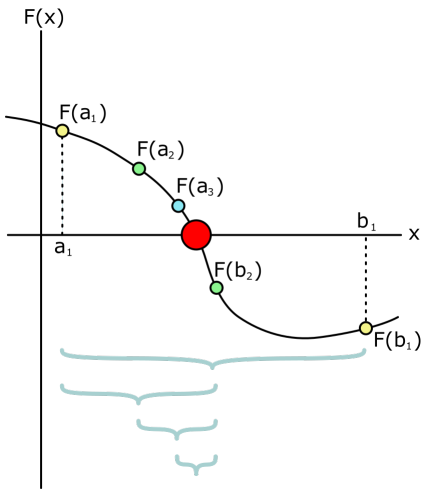
3. 

Figure 1: An example of several iterations of the bisection method. By Bisection_method.svg: Tokuchanderivative work: Tokuchan (talk) - Bisection_method.svg, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=9382140

The bisection method is a root-finding method that applies to any continuous functions for which one knows two values which the function gives results with opposite signs when applied to each of those two values. The method consists of repeatedly bisecting the interval defined by these values and then selecting the subinterval in which the function changes sign, and therefore must contain a root. It is a very simple and robust method, but it can be relatively slow. The input for the method is a continuous function f, an interval minimum, a, and maximum, b. The function values are of opposite sign (there is at least one zero crossing within the interval). Each iteration performs these steps:

1. Calculate c, the midpoint of the interval, $c = \frac{a+b}{2}$
2. Calculate the function value at the midpoint, $f(c)$.
3. If convergence is satisfactory (that is, $|f(c)|$ is sufficiently small), return c and stop iterating.
4. Examine the sign of $f(c)$ and replace either a or b with c so that there is a zero crossing within the new interval.

(a) [3 points] Two real numbers have the same sign if their product is positive. Define a SCHEME function, named (`same-sign? a b`) which accepts two real numbers, a and b, and evaluates to `#t` if both a and b have the same sign.

(b) [7 points] Define a SCHEME function, named (`bisection f a b tol`), which accepts a continuous function, `f`, the minimum a and maximum b of an interval in which $f(a)$ and $f(b)$ have opposite signs, and the desired tolerance for the final estimate of the zero of that function, `tol`. Your function should return an estimate of a zero of the function f that is within the desired accuracy (tolerance) specified by `tol` using the bisection method outlined above.

4. [10 points] The hyperbolic cosine function $\cosh(x)$ is defined by the power series

$$\cosh(x) = \sum_{k=0}^{\infty} \frac{x^{2k}}{(2k)!} = 1 + \frac{x^2}{2!} + \frac{x^4}{4!} + \frac{x^6}{6!} + \cdots$$

Write a recursive SCHEME function which, given n, computes the expansion up to term n (which is a sum of $n+1$ terms):

$$\sum_{k=0}^{n} \frac{x^{2k}}{(2k)!}.$$

For full credit, your implementation **must not** use the built-in function `expt` nor rely on an implementation of factorial. You may, however, assume the existence of a function `(sum term a b)`:

```
(define (sum term a b) ;; Given f, a and b, compute f(a)+f(a+1)+ ... +f(b).
    (if (> a b)
        0
        (+ (term a)
           (sum term (+ a 1)  b))))
```