625-600: Programming Exercise

Read every page very carefully before you begin.

1. Implement `deriv` to support:
   addition, subtraction, unary minus, multiplication, division, sqrt, log, exp, sin, cos, tan.
   → HINT: use slide02 page 44 as a skeleton.
2. Implement simplification routines `splus` etc. for all operators and integrate it into `derivplus`, etc.
   → HINT: Integrate code in slide02 page 45 into code in page 44. (Code available on course web page, under the src/ directory.)
3. Write a simple function `deriv-eval` to assign a numerical value to the variable and get a single number corresponding to the resulting derivative:
   `(deriv-eval '(+ (* x x) (- 2 x)) 'x 20)`
   * You must use recursion.
4. Write a recursive simplification function `simplify` that could simplify expressions like `(+ 0 (+ (* x 0) x))`

Programming Exercise 1: other conditions

1. Use only one variable (say `X`). Other symbols should be treated as constants (e.g. `Y, Z, ...`).
2. All operators should be binary operators:
   i.e. expressions like `(+ 1 2 3 4 5)` do not need to be supported. Only those in the form of `(+ 1 2)` are expected to be used.
3. The only exception is the unary minus operator `(- 10)`, which only has one argument.
4. You must check for division by zero and print an error message in case such an event occurs, especially for the `deriv-eval` function.

Programming Exercise 1: Example Inputs and Outputs

1. `(deriv '(* (+ x 4) (+ x 5)) 'x)`
   `-> (+ (+ X 4) (+ X 5)))`
2. `(deriv '(/ (+ x 1) x) 'x)`
   `-> (/ (- X (+ X 1)) (* X X))`
3. `(deriv-eval '(* (+ x 4) (+ x 5)) 'x 10)`
   `-> 29`
4. `(deriv-eval '(/ (+ x 1) x) 'x 5)`
   `-> -1/25`

Programming Exercise 1: Things to Try

- Program code (**deriv.lsp**): put it in a single text file.
  – Ample indentation and documentation is **required**.
- Sample inputs and outputs
  – 10 non-trivial examples, each containing a combination of more than 5 operators. Provide examples for `deriv`, `deriv-eval`, and `simplify`.

Programming Exercise 1: Important Grading Information

- Since the deriv functions call the simplification functions splus etc., if the simplification routine is broken, regardless of the deriv functions being correct, your call will result in an error. If this happens, **both** deriv and simplification will be graded as malfunctioning.
- If you got deriv functions to work, but if simplification is not working, take out the simplification code from your deriv functions so that at least your deriv functions work.

Programming Exercise 1: Submission

- You don't need to submit anything.

Differentiation rules

c: constant; $f(x), g(x)$: functions of x; Lisp `(expt x y)` = x^y.

$$\frac{d(f/g)}{dx} = \frac{1}{g^2}\left(g\frac{df}{dx} - f\frac{dg}{dx}\right)$$

$$\frac{df^c}{dx} = cf^{c-1}\frac{df}{dx}, \qquad \frac{d\sqrt{f}}{dx} = \frac{1}{2\sqrt{f}}\frac{df}{dx}$$

$$\frac{d\log(f)}{dx} = \frac{1}{f}\frac{df}{dx}, \qquad \frac{d\exp(f)}{dx} = \exp(f)\frac{df}{dx}$$

$$\frac{d\sin(f)}{dx} = \cos(f)\frac{df}{dx}, \qquad \frac{d\cos(f)}{dx} = -\sin(f)\frac{df}{dx}$$

$$\frac{d\tan(f)}{dx} = (1 + tan^2(f))\frac{df}{dx}$$