

TREES — PART 2

This week, we want to handle trees that can represent mathematical functions. We will handle usual mathematical functions of a variable. For example, we want to be able to express :

1. $f(x) = \sqrt{x} + \frac{3}{\sqrt{x}}$
2. $g(x) = (x - 2) \cdot e^x$
3. $h(x) = \frac{15\,000}{1 + 39 \cdot e^{-0.75t}}$

Last time we used the Tree class, see Listing 1.

```

1 class Tree:
2     def __init__(self, data, left=None, right=None):
3         self.data = data
4         self.left = left
5         self.right = right
6
7     def __str__(self):
8         return str(self.data)

```

Listing 1: The Tree data structure.

This week, we want to use different types of nodes for our trees:

1. Operations (+, -, *, / and ^): they have two children (left and right).
2. Leaves: numbers (1, 0.75...) or variables (most of the type x , sometimes t): they do not have children (i.e., left and right are both equal to None).
3. Functions (cos, sin, tan, sqrt, exp, ln...): they have one child (we'll use the convention that this child is left, and that right has to be None).

Listing 2 is an example of use that defines $\sqrt{x} + \frac{3}{\sqrt{x}}$. You can download it from http://www.bar-samian.am/2023-2024/S7ICTC/TP10_Trees.py.

```

1 x = Tree("x")
2 three = Tree(3)
3 sqrt_x = Tree("sqrt", x)
4 sub_tree = Tree("/", three, sqrt_x)
5 f = Tree("+", sqrt_x, sub_tree)

```

Listing 2: Mathematical expression $\sqrt{x} + \frac{3}{\sqrt{x}}$.

1. Draw the tree associated to the variable `f`. Here, the sub-tree `sqrt_x` is present twice in the tree. You can draw it twice or once, it does not change the result.
2. Write a function `evaluate_tree` that takes 3 arguments `tree`, `variable` and `value`, and computes the mathematical result of the expression contained in the tree, where the variable is replaced by the value. You can use the function `compute_tree` which is given in the python file this week, enhanced from your work last time.

For example, `evaluate_tree(f, "x", 9)` should be equal to $\sqrt{9} + \frac{3}{\sqrt{9}} = 4$.

3. Create in python the variable `g` which is the tree associated to $g(x)$. Please use a node that represents the exponential function (`exp`) with `x` as left child, instead of using the exponentiation operator `^` with `2.718281828` and `x` as children.

Make sure that the `evaluate_tree` function from the previous question also works here. For example, check that `evaluate_tree(g, "x", 1)` is equal to ≈ -2.718281828 .

4. Our goal is now to write a `derive_tree` function that computes derivatives. This function has two arguments `tree`, `variable`. It will be a list of "ifs", where we'll check each possibility for the data. In most of the cases, it is possible to simplify the writings of the functions. I do not ask to simplify trees that you will have, I just ask that the mathematical expression obtained is correct.

- (a) Constants: what is the derivative of a constant? Write this case in python.

Test it: derive the tree `three`.

- (b) Variable alone: what is the derivative of x ? Write this case in python.

Test it: derive the tree `x`.

- (c) The `+`: if f and g are two functions, how do you compute $(f(x) + g(x))'$? Write this case in python.

Test it: write the tree `x + 3` and derive it.

- (d) The `-`: if f and g are two functions, how do you compute $(f(x) - g(x))'$? Write this case in python.

Test it: write the tree `4 - x` and derive it.

- (e) The `*`: if f and g are two functions, how do you compute $(f(x) * g(x))'$? Write this case in python.

Test it: write the tree `(x + 3) * (4 - x)` and derive it.

- (f) The `/`: if f and g are two functions, how do you compute $\left(\frac{f(x)}{g(x)}\right)'$? Write this case in python.

Test it: write the tree `1/x` and derive it.

- (g) The `^`: we'll only cover the case where we have either a^b where a and b are constants, or x^n where x is the variable and n is an integer constant. How do you compute $(a^b)'$? How do you compute $(x^n)'$? Write these cases in python.

Test it: write the trees `2^4` and `x^3` and derive them.

- (h) The `sqrt`: if f is a function, how do you compute $\left(\sqrt{f(x)}\right)'$? Write this case in python.

Test it: derive the `f` tree.

- (i) The `exp`: if f is a function, how do you compute $(e^{f(x)})'$? Write this case in python.

Test it: derive the `g` tree.

5. In this last question, we'll try to make some simplifications. We'll write a function `simplify_tree` that outputs another tree, equivalent to the one given as argument, but easier.

For instance, it will convert the tree `0 + ...` into just `...`. It will convert the tree `1 * ...` into just `...`. It will convert `0 * ...` into just `0`. And so on.