Activities: Functions

May 15, 2020

(1) Create a function that calculates the n-th harmonic number `harmonic(n)`. Have your function check that the argument n is positive.

(2) Write a function that takes a number between 1 and 100 and returns an American style letter grade: 'A' for numbers ≥ 90, 'B' for numbers, 'C' for numbers ≥ 65, 'D' for numbers ≥ 55 and 'F' else.

(3) Write a function `num_der(function, delta = 0.000001)` that takes a function and calculates its numerical derivative using the symmetric difference formula $\frac{df}{dx} = \frac{f(x+\delta) - f(x-\delta)}{2 \cdot \delta}$. Notice that we gave delta a default value that can be overwritten by the user of the function. Then try out the function on math.sin and math.cos.

(4) Write a recursive function `fibonacci(n)` that strictly uses the following definition:

$$\text{fibonacci}(n) = \begin{cases} n & \text{if } n \leq 1 \\ \text{fibonacci}(n-1) + \text{fibonacci}(n-2) & \text{otherwise} \end{cases}.$$

(a) Try this function out for $n = 0, n = 1, n = 2, n = 3, n = 4, n = 20, n = 25, n = 30\ldots$ Around 35, the execution will be really slow because of the number of recursive calls.

(b) Import the lru_cache decorator from the built-in module func_tools: `from func_tools import lru_cache`. Then use the decorator `@lru_cache(maxsize = None)` on your function. Compare the behavior.

(c) Next week, we are going to build our own decorator and also use a much more efficient way of implementing the Fibonacci number.