Day 1 Exercises

18 December 2018

The factorial function

Implement a function `factorial(x)` that takes in a nonnegative integer `x` and returns its factorial.

The exponential function

Implement a function `exp(x, tol)` that uses a Taylor expansion to calculate e^x to an accuracy tolerance of `tol`. If only one argument is supplied, assume that the argument is `x` and have `tol` default to 10^{-8}. You can ensure accurate expansions by considering an n^{th} order expansion and making sure that this agrees with the $(n+1)^{\text{st}}$ order expansion to within `tol`.

You may wish to use the `factorial()` function, which you have defined in the previous problem. Start by calculating the 0^{th} order expansion, followed by the 1^{st}, and then comparing the two. After that, compare the 1^{st} with the 2^{nd}, and so on. Once you find that the values agree to within `tol`, return the result.

Matrix multiplication

In this problem, we will represent a matrix as a list of lists. If the dimension of the matrix is $m \times n$, the size of the outer list will be m and the size of each inner list will be n. For instance, the 4×3 matrix

$$\boldsymbol{M} = \begin{bmatrix} 0 & 9 & 1 \\ 7 & 2 & 6 \\ 3 & 5 & 4 \\ 6 & 0 & 3 \end{bmatrix}$$

will be represented as

```
M = [[0, 9, 1],
     [7, 2, 6],
     [3, 5, 4],
     [6, 0, 3]]
```

Write a function `matmul(A, B)` that implements matrix multiplication between two matrices `A` and `B`. You will need to have a way to check if the dimensions of the matrices agree. For instance, if they do not agree, you can return `None` or raise an `Exception`.

The rectified linear unit function

The rectified linear unit function $\text{ReLU}(x)$, which is commonly used in machine learning, returns x if x is positive and 0 otherwise. If the input is a vector or matrix, it is applied elementwise. As a concrete example,

$$\text{ReLU}\left(\begin{bmatrix} 8 & -18 & 4 & \pi \end{bmatrix}^\top\right) = \begin{bmatrix} \text{ReLU}(8) & \text{ReLU}(-18) & \text{ReLU}(4) & \text{ReLU}(\pi) \end{bmatrix}^\top = \begin{bmatrix} 8 & 0 & 4 & \pi \end{bmatrix}^\top.$$

Write a function that calculates $\text{ReLU}(x)$ using at most five lines of code. Create a new line after every colon and do not use semicolons. Make sure that this function should work for numbers, lists of numbers, and matrices.

Object-oriented programming

```
class Animal():
  def __init__(self, age):
    self.age = age
    self.lifespan = 25
  def bark(self):
    print("...")
  def show_avg_life_left(self):
    print("I have", lifespan - age, "years left in this world")

class Dog(Animal):
  def __init__(self, age):
    # 1 human year is 7 dog years
    self.age = 7 * age
  def bark(self):
    print("Woof")
  def show_avg_life_left(self):
    print("Woof, looks like I've only got", 7 * (lifespan - age), "years left")

class Cat(Animal):
  def bark(self):
    print("Meow")
  def show_avg_life_left(self):
    print("Meow, I have but", lifespan - age, "years left, meow")
  def nyan(self):
    print("Nyan nyan nyan nyan nyan")

spot = Dog(2)
iris = Cat(1)
marc = Animal(17)
```

Answer the following questions without running the code.

1. What is the result of running `spot.lifespan()`?
2. What is the result of running `iris.lifespan()`?
3. What is the result of running `iris.show_avg_life_left()`?
4. What is the result of running `spot.bark()`?
5. In implementing the function `bark()`, why does it contain `self` as an input, but does not take in any input at runtime?
6. What is the result of running `marc.bark()`?
7. What is the result of running `marc.nyan()`?