Lec09-functions-RyanSponzilli

September 24, 2024

1 ASTR 310 Lecture 09: functions

1.0.1 1. Euclid's Algorithm

Euclid's algorithm finds the greatest common divisor (gcd) of two integers a and b. The procedure is as follows: beginning with $r_{-1} = a$ and $r_0 = b$, choose successive values

$r_{k+1} = r_{k-1} \bmod r_k$; $k = 0,...$

until for some $k = N$ you find $r_N = 0$. The gcd of a and b is then r_{N-1}.

Write a Python function to compute the gcd of two numbers. Ensure that you begin the iteration with integers (i.e. by casting floats to integer).

Test your function on the pairs:

- `gcd(35, 42)` $= 7$
- `gcd(1071, 462)` $= 21$
- `gcd(2232, 5952)` $= 744$
- `gcd(6831, 15384)` $= 3$

[10 pts total]

[33]:

```python
def gcd(a: int, b: int):
    r_p = a
    r = b

    while(r != 0):
        temp = r
        r = r_p % r
        r_p = temp

    return r_p
```

[34]:

```python
print(gcd(35, 42))
print(gcd(1071, 462))
print(gcd(2232, 5952))
print(gcd(6831, 15384))
```

```
7
21
```

```
744
3
```

1.0.2 2. Optional Arguments and Multiple Return Values

Write a Python function that will take a list of numbers and return various statistics. The function should take one required argument (the list of numbers) and one optional argument (a tuple or list of strings naming the statistics to compute). The default value for the optional argument should be `("mean",)`. The allowed strings are as follows.

- `"mean"` return the mean value of the list
- `"sum"` return the sum of the list values
- `"rms"` return the root-mean-square
- `"max"` return the maximum
- `"min"` return the minimum

If the list has N elements L_i, the mean is $\frac{1}{N}\sum_{i=0}^{N-1} L_i$ and the rms is $\sqrt{\frac{1}{N}\sum_{i=0}^{N-1} L_i^2}$.

The return value should be a tuple containing the requested statistics in the same order as they appear in the request tuple.

Test your function on the list: `[-34.2, 17.1, 3.8, 1.4, 22.6, 19.4, 15.9]` with different combinations of requested statistics (or no request).

Numpy arrays are still illegal! Check Reading 7 for operations on lists. Avoid overriding the builtins (e.g. don't create a new variable called "sum").

[10 pts total]

[37]:

```
from math import *
def stats(l: list, stats: tuple = ("mean",)):
    l_dict = dict()
    for stat in stats:
        if stat == "mean":
            l_dict["mean"] = sum(l) / len(l)
        elif stat == "sum":
            l_dict["sum"] = sum(l)
        elif stat == "rms":
            l_dict["rms"] = sqrt(sum([i**2 for i in l]) / len(l))
        elif stat == "max":
            l_dict["max"] = max(l)
        elif stat == "min":
            l_dict["min"] = min(l)

    return tuple(l_dict.values())
```

[38]:

```
print(stats([-34.2, 17.1, 3.8, 1.4, 22.6, 19.4, 15.9]))
print(stats([-34.2, 17.1, 3.8, 1.4, 22.6, 19.4, 15.9], stats = ("min", "sum")))
print(stats([-34.2, 17.1, 3.8, 1.4, 22.6, 19.4, 15.9], stats = ("min", "sum",␣
  ↪"mean", "rms")))
```

```python
print(stats([-34.2, 17.1, 3.8, 1.4, 22.6, 19.4, 15.9], stats = ("mean", "sum", "max")))
print(stats([-34.2, 17.1, 3.8, 1.4, 22.6, 19.4, 15.9], stats = ("rms", "min", "max", "mean", "sum")))
```

```
(6.571428571428571,)
(-34.2, 46.0)
(-34.2, 46.0, 6.571428571428571, 19.340483078617392)
(6.571428571428571, 46.0, 22.6)
(19.340483078617392, -34.2, 22.6, 6.571428571428571, 46.0)
```