Last class recap & today

Today:

- practice writing simple programs (SIPO)

- modulo op

- constants: why useful & best practice

- practice using `math` module

- int, float types & mixed arithmetic

- more ‘print’ tricks

Third program: dhms2s

Task: convert a time given in days, hours, minutes, and seconds to the equivalent number of seconds

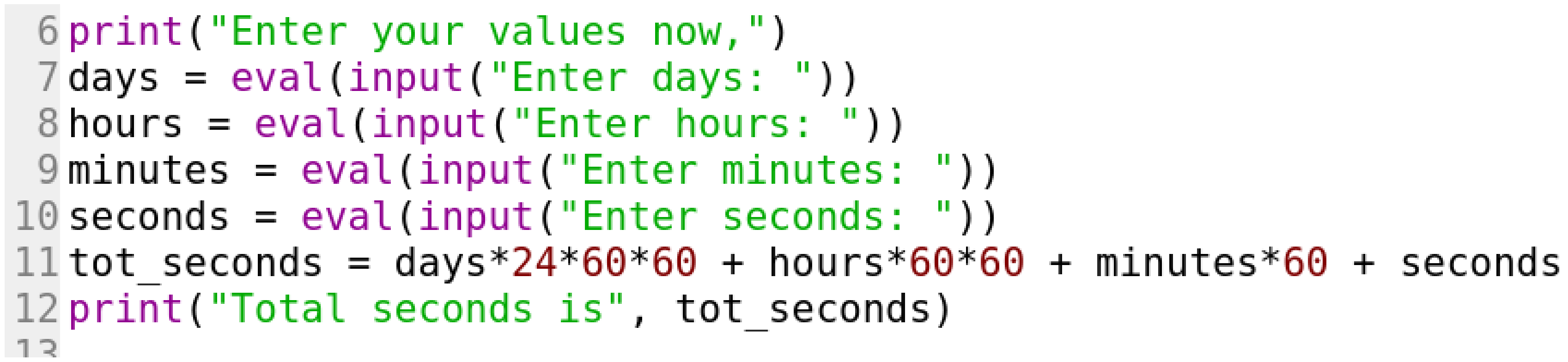
- do it by hand for a few input cases

- think what input, output should be what needs to be calculated

- translate into python

- check if code reproduces calc’s done by-hand

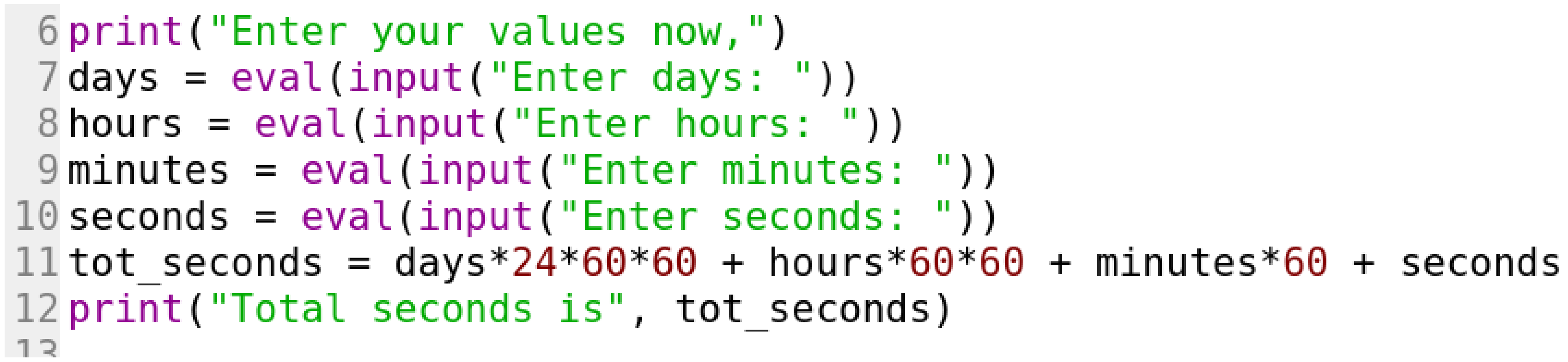
Third program: dhms2s



```python
print("Enter your values now,")
days = eval(input("Enter days: "))
hours = eval(input("Enter hours: "))
minutes = eval(input("Enter minutes: "))
seconds = eval(input("Enter seconds: "))
tot_seconds = days*24*60*60 + hours*60*60 + minutes*60 + seconds
print("Total seconds is", tot_seconds)
```

Can we make this code faster?

Third program: dhms2s



```python
print("Enter your values now,")
days = eval(input("Enter days: "))
hours = eval(input("Enter hours: "))
minutes = eval(input("Enter minutes: "))
seconds = eval(input("Enter seconds: "))
tot_seconds = days*24*60*60 + hours*60*60 + minutes*60 + seconds
print("Total seconds is", tot_seconds)
```

Can we make this code faster?

tot_seconds = ((days*24 + hours)*60 + minutes)*60 + seconds

s2dhms

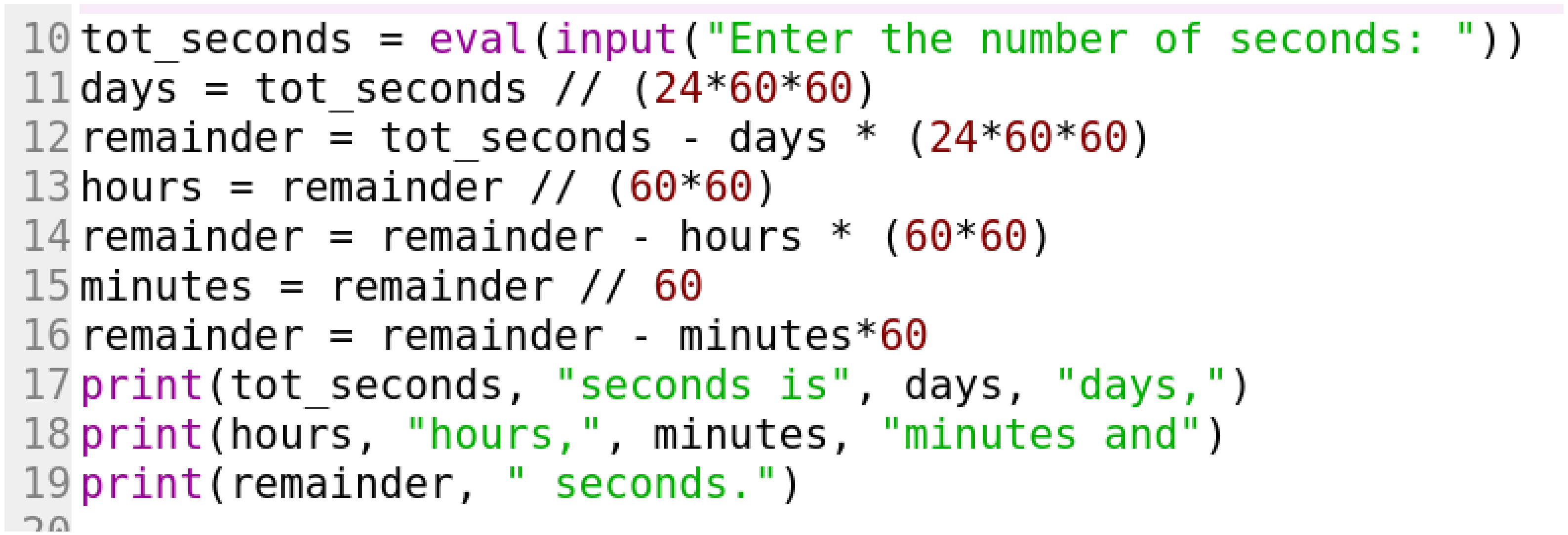
Now lets write a program that does the inverse calculation:

take a large number of seconds and find the equivalent number of days, hours, minutes, and seconds

- do it by hand (consider 200,000 seconds)

- determine: input, output, processing steps

s2dhms



```python
tot_seconds = eval(input("Enter the number of seconds: "))
days = tot_seconds // (24*60*60)
remainder = tot_seconds - days * (24*60*60)
hours = remainder // (60*60)
remainder = remainder - hours * (60*60)
minutes = remainder // 60
remainder = remainder - minutes*60
print(tot_seconds, "seconds is", days, "days,")
print(hours, "hours,", minutes, "minutes and")
print(remainder, " seconds.")
```

Let’s improve this code!

s2dhms

days = 200,000/(24*60*60) = 2
remainder = 200,000 - 2*(24*60*60) = 27,200
hours = 27,200/(60*60) = 7
remainder = 27,200 - 7x(60*60) = 2000
minutes = 2,000/60 = 33
seconds = 2,000 - 33*60 = 20

- the **input** is the number of seconds (200,000),
- the **processing** consists of several steps to find the number of days, hours, minutes and seconds, including intermediate steps to find the remainders along the way,
- the **output** is the number of days, hours, minutes and seconds.

modulo: 5th arithmetic operator

- yields the remainder of a division

- denoted by ‘%’

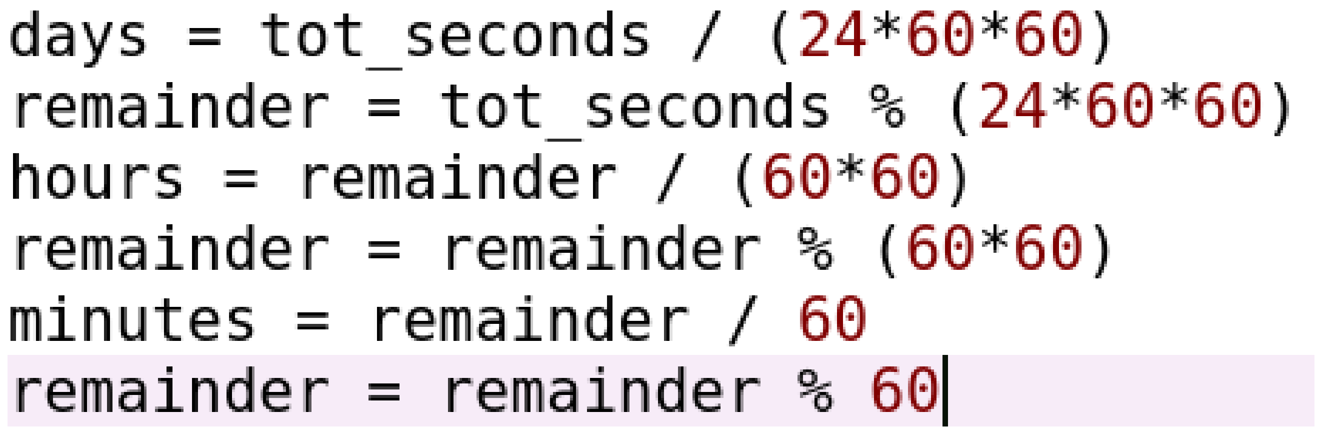
example: 7%3 = remainder of 7/3 = 1

- everyday arithmetic: -,+,/,*

- remainder calc is needed in many computat’ns

- exist in most programming languages

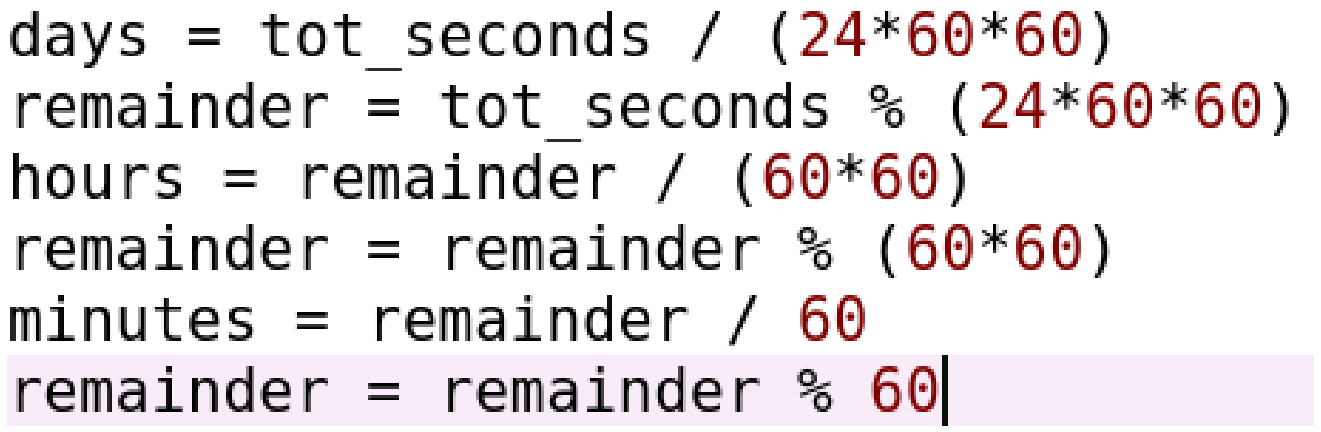
s2dhms: rewrite using modulo



```
days = tot_seconds / (24*60*60)
remainder = tot_seconds % (24*60*60)
hours = remainder / (60*60)
remainder = remainder % (60*60)
minutes = remainder / 60
remainder = remainder % 60
```

Can we do better?

s2dhms: improve via symbolic constants



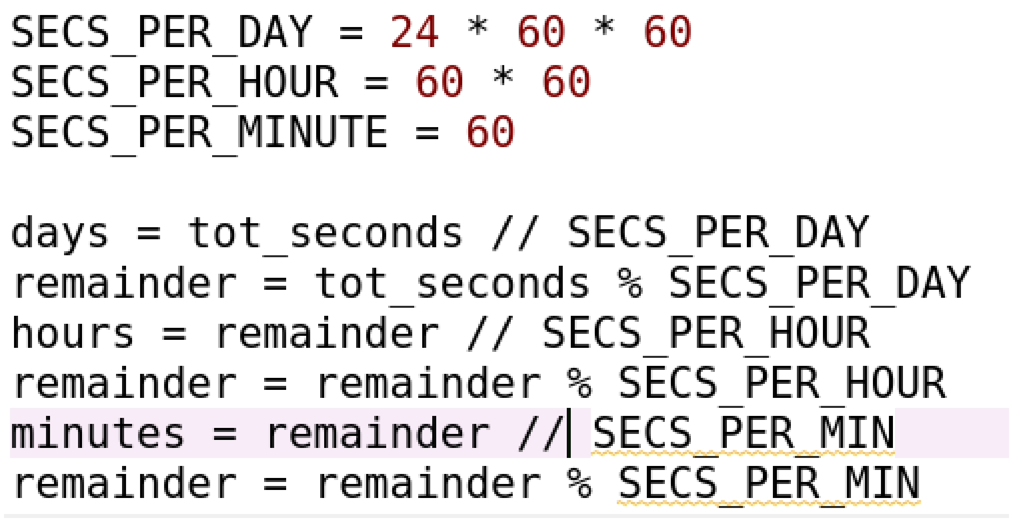
```
days = tot_seconds / (24*60*60)
remainder = tot_seconds % (24*60*60)
hours = remainder / (60*60)
remainder = remainder % (60*60)
minutes = remainder / 60
remainder = remainder % 60
```

We calculate the same thing more than once,
hence there inefficiency.

Also, its not clear why are we multiplying certain numbers etc.

Let’s avoid these using symbolic constants

s2dhms: improve via symbolic constants



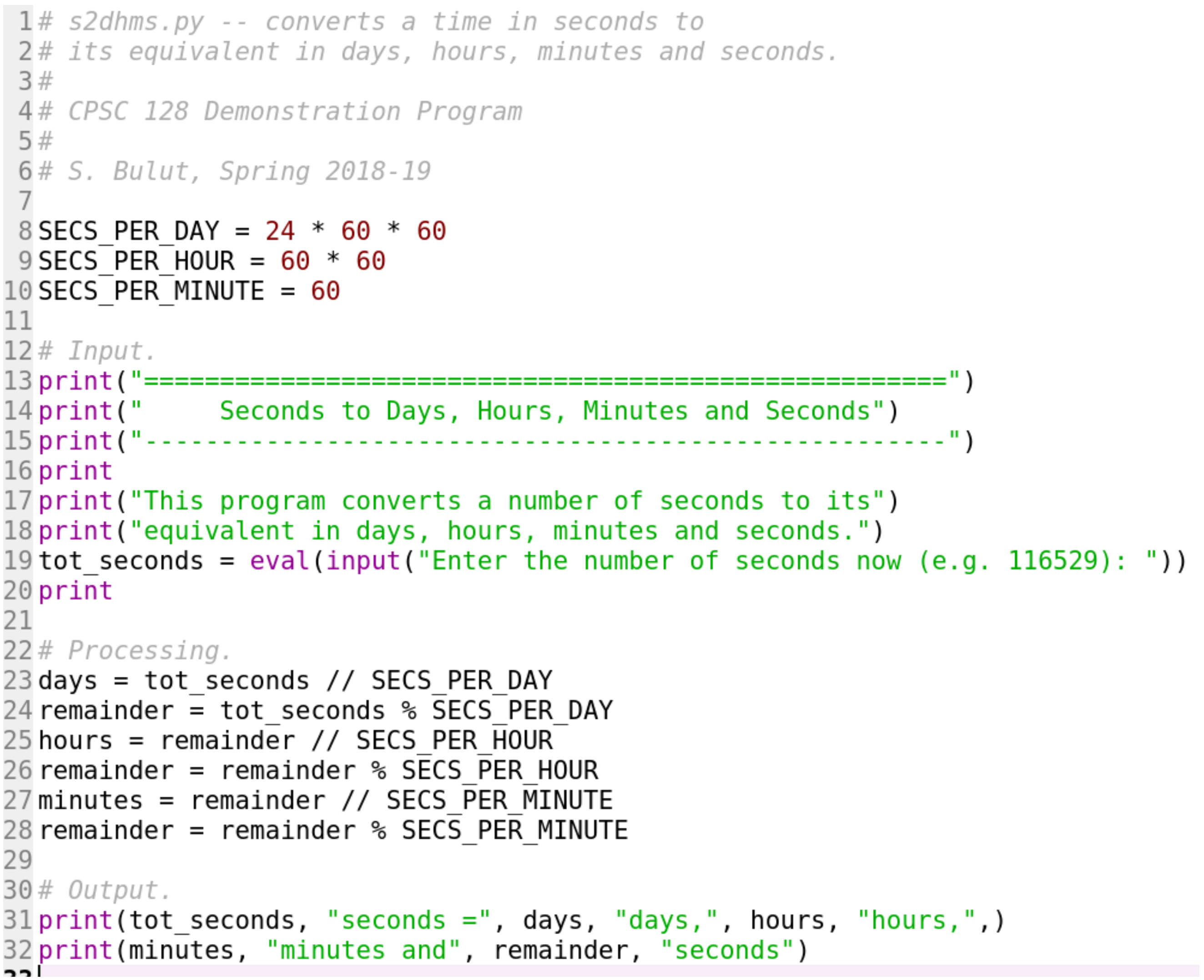
```
SECS_PER_DAY = 24 * 60 * 60
SECS_PER_HOUR = 60 * 60
SECS_PER_MINUTE = 60

days = tot_seconds // SECS_PER_DAY
remainder = tot_seconds % SECS_PER_DAY
hours = remainder // SECS_PER_HOUR
remainder = remainder % SECS_PER_HOUR
minutes = remainder // SECS_PER_MIN
remainder = remainder % SECS_PER_MIN
```

- Calculate once, use many times

- pick sym. constant names in a descriptive way to increase clarity

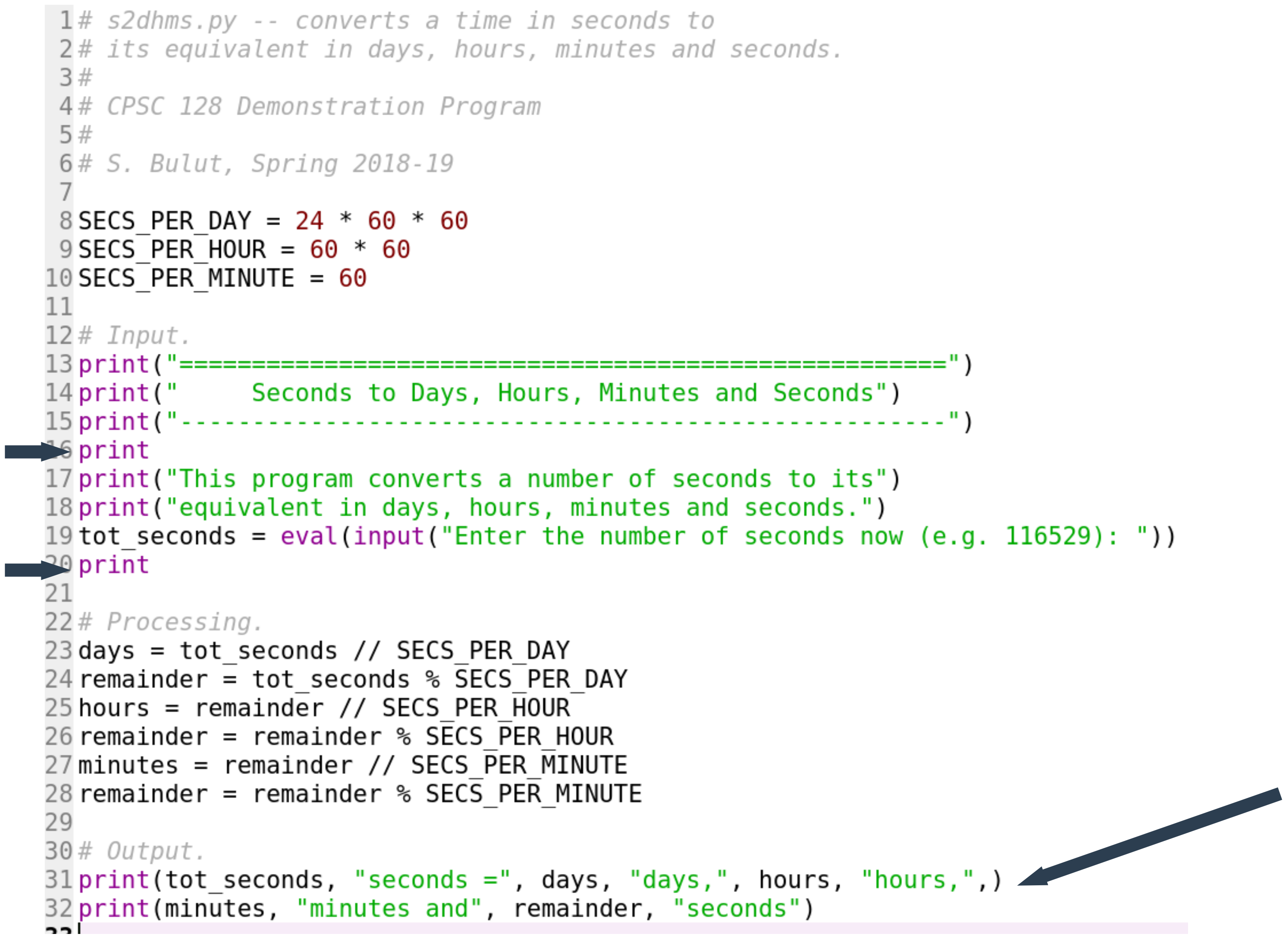
- Python convention: use capitals for constants: a cue to the reader

s2dhms: putting it all together



```python
# s2dhms.py -- converts a time in seconds to
# its equivalent in days, hours, minutes and seconds.
#
# CPSC 128 Demonstration Program
#
# S. Bulut, Spring 2018-19

SECS_PER_DAY = 24 * 60 * 60
SECS_PER_HOUR = 60 * 60
SECS_PER_MINUTE = 60

# Input.
print("=====================================================")
print("     Seconds to Days, Hours, Minutes and Seconds")
print("-----------------------------------------------------")
print
print("This program converts a number of seconds to its")
print("equivalent in days, hours, minutes and seconds.")
tot_seconds = eval(input("Enter the number of seconds now (e.g. 116529): "))
print

# Processing.
days = tot_seconds // SECS_PER_DAY
remainder = tot_seconds % SECS_PER_DAY
hours = remainder // SECS_PER_HOUR
remainder = remainder % SECS_PER_HOUR
minutes = remainder // SECS_PER_MINUTE
remainder = remainder % SECS_PER_MINUTE

# Output.
print(tot_seconds, "seconds =", days, "days,", hours, "hours,",)
print(minutes, "minutes and", remainder, "seconds")
```

s2dhms: putting it all together



```python
# s2dhms.py -- converts a time in seconds to
# its equivalent in days, hours, minutes and seconds.
#
# CPSC 128 Demonstration Program
#
# S. Bulut, Spring 2018-19

SECS_PER_DAY = 24 * 60 * 60
SECS_PER_HOUR = 60 * 60
SECS_PER_MINUTE = 60

# Input.
print("=====================================================")
print("     Seconds to Days, Hours, Minutes and Seconds")
print("-----------------------------------------------------")
print
print("This program converts a number of seconds to its")
print("equivalent in days, hours, minutes and seconds.")
tot_seconds = eval(input("Enter the number of seconds now (e.g. 116529): "))
print

# Processing.
days = tot_seconds // SECS_PER_DAY
remainder = tot_seconds % SECS_PER_DAY
hours = remainder // SECS_PER_HOUR
remainder = remainder % SECS_PER_HOUR
minutes = remainder // SECS_PER_MINUTE
remainder = remainder % SECS_PER_MINUTE

# Output.
print(tot_seconds, "seconds =", days, "days,", hours, "hours,",)
print(minutes, "minutes and", remainder, "seconds")
```

blank lines

Line continuation

Data types: int and float

- most common data types to store numbers

- **int** -> integers (positive & negative whole numbers without integers

- **float** -> floating point values (numbers with decimals)

- “5” is not the same as “5.” or “5.0” in Python and other lang’s

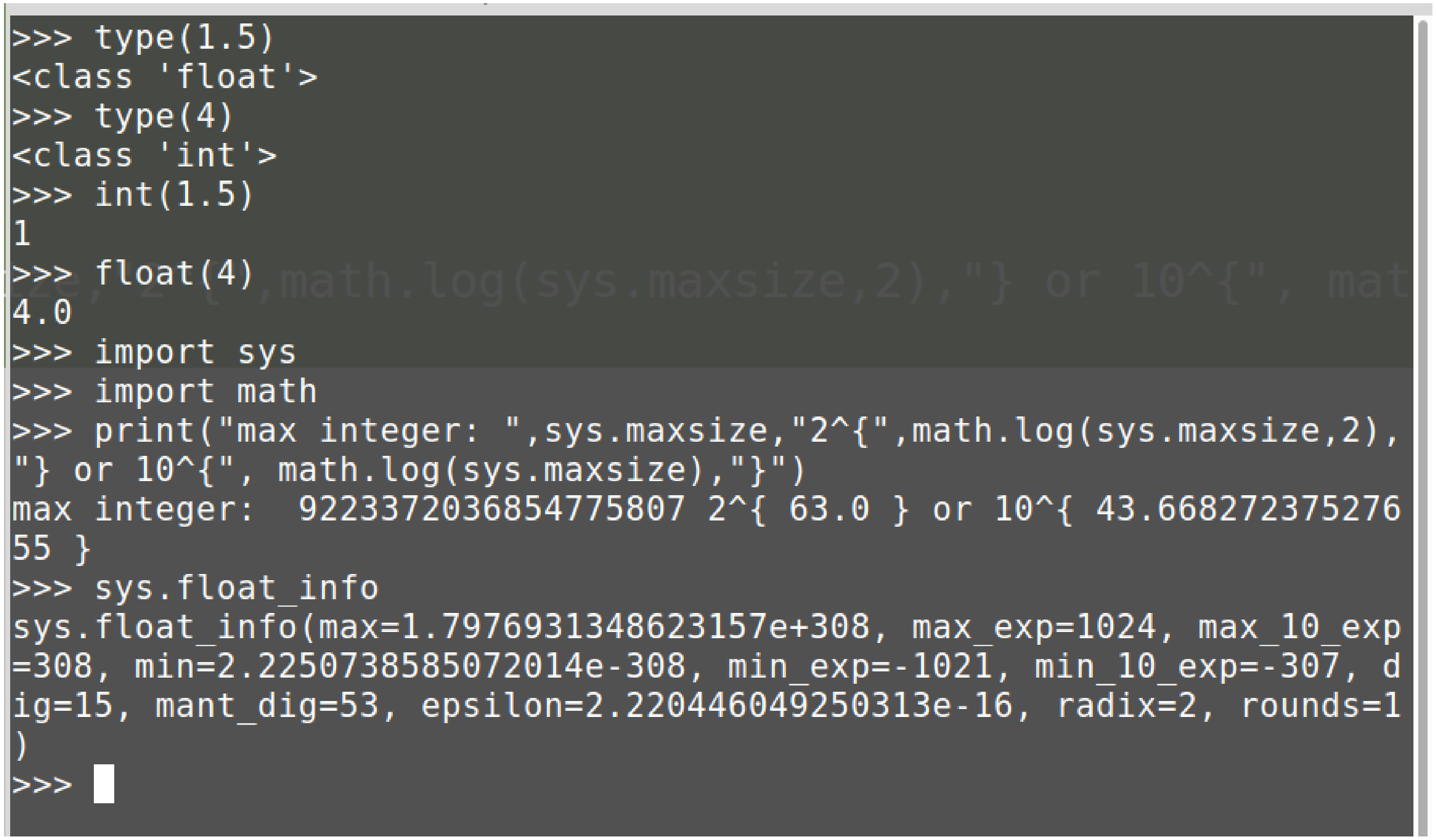
- use **float** for measurements etc, use **int** for counts

- avoid mixing **int**’s and **float**’s (especially in python2)
    although Python does its best to convert things to a more general type
    i.e. int -> float , float -> complex
    - subtle problems can arise
    - conversion takes extra time

- Not Good: (-40 + 32.0)* 5 / 9 **Good:** (-40.0 + 32.0)*5.0/9.0

- Casting: int(3.14) -> 3 float(4) -> 4.0

Data types: int and float (Aside)



```
>>> type(1.5)
<class 'float'>
>>> type(4)
<class 'int'>
>>> int(1.5)
1
>>> float(4)
4.0
>>> import sys
>>> import math
>>> print("max integer: ",sys.maxsize,"2^{",math.log(sys.maxsize,2),
"} or 10^{", math.log(sys.maxsize),"}")
max integer:  9223372036854775807 2^{ 63.0 } or 10^{ 43.668272375276
55 }
>>> sys.float_info
sys.float_info(max=1.7976931348623157e+308, max_exp=1024, max_10_exp
=308, min=2.2250738585072014e-308, min_exp=-1021, min_10_exp=-307, d
ig=15, mant_dig=53, epsilon=2.220446049250313e-16, radix=2, rounds=1
)
>>>
```

use math module:
calculate diameter of a tree

Scott used a tape measure to measure the circumferences of all the trees in a regeneration plot, but the database wants him to enter diameters instead of circumferences. Assuming the trees are roughly circular in cross-section we can calculate the diameter from the circumference, because the diameter is twice the radius and the radius is the circumference divided by 2π. Write a program that will let him enter the circumference and will print the corresponding diameter. A sample run might look like:

use math module: calculate diameter of a tree

```python
import math
import math as m
from math import pi
#from math import *

circ = eval(input("Enter circumference:"))
print()
diameter = circ/math.pi
diameter2 = circ/m.pi
diameter3 = circ/pi

print("Diamter is ",diameter)
print("Diamter is : %1.2f" % diameter)
print("Diamter is : %1.2f" % diameter2)
print("Diamter is : %1.2f" % diameter3)

```