Announcements

- HW4 is posted due next Monday.
- Later this week-sample midterm 1.
- Next Tues or Wed in class review followed by in class midterm.
Expressions & Statements

Expressions are things that compute a value.

Ex: `word` + 2*`a` - string
    a + b * c - numeric expression
    `name` + `space` - string expression

These are operations on any type that compute a value.
Functions

We call functions and give in input parameters.

Ex: \( \text{round}(\text{fltn}) \) # returns rounded float

However, \text{round} is also an object!

Ex: \( X = \text{round} \) # valid!

\[ a = X(n) \] # now calls \text{round}(x)
Chaining Calls

Functions & methods are essentially the same.

They are attached to a class/object.

Ex: mylist.sort()

You can combine them in many ways:

Ex: s.lower().find('a')

# returns true if 'a' or 'A' is in s

round(x + 3.5)
dramatic(s.lower())
Return Types

All functions must have a return type. If no return, then defaults to None. (This can actually be useful later on...)
Statements & Expressions

Statements are any stand-alone executable. (An expression on a line by itself is a statement.)

Some are useful for side effects:

```python
mylist.sort()  # returns None
```

Statements are generally one line, but can extend:

```python
a = (b + c + d)  # or [77 and 33]
```
Back to Practice problem

Make gcd function based on our algorithm from last week.

(See code.)

Can import functions to python:
from filename import function
Type checking

If we write a function, it's good to be able to check that a user is sending valid inputs.

Can avoid strange error messages or deal with issues more directly.

Ex: `gcd('Hello', 'Goodbye')`

Error shows up in our function, even though our code is just fine.
Type Checking

So, to check if input is correct type, use: `isinstance(variable, (type1, type2))`

Returns True if variable is of a type in the tuple `(type1, type2,...)` (and False otherwise).

If wrong type, we can raise an error.
Raising errors

What type of errors have we seen so far?

TypeError, NameError, ValueError

To raise our own errors, use raise command.

Ex: if (something is wrong type)
raise TypeError('error message')
Practice 5.31

Write a function `yesOrNo(prompt)` that asks the user a question (specified in the string `prompt`) and demands a response of `yes` or `no`.

If wrong input is received, ask user again for yes or no answer (repeat until you get one).

Return `true` if user says `yes`, `false` if user says `no`. 