Announcements

- Course evaluations today. Need people to handle them, I am supposed to be out of the room.
- Review for final on Friday.
- Final Monday 8-10 AM. Bring a Scantron. Open notes.

Classes

- Modules often define new kinds of objects.
- These are called classes – like data types but defined by a programmer, not built into Python.
- If you want to understand the code in some existing module, or maybe change it, it will help to understand how classes are made.

Instance

- There can be lots of objects of type string, and similarly lots of objects of class Button.
- Each object is an instance of its class.
  - conditions = "Showers"
  - todayBut = Button(content, text="today")
  - conditions is an instance of string.
  - todayBut is an instance of Button

Factory Functions

- Create instances of the class.
- Often these are the only functions in the module.
- Anything that works with an existing instance is given as a method.
- This is the difference between a function and a method.

```python
todayBut = Button(content, text="today")
todayBut.configure(command=go)
```
Attributes

- A class can store data; for instance, the text associated with a Button, or the function it calls.
- Data stored in a class is its attributes.

```python
todayBut.configure(command=go)
```

- This method changes the command attribute of the Button instance todayBut.

Deck of cards class

- Attribute will be a list of cards
- A card is a suit and a number – stored as a tuple.

```python
# factory function; always called __init__
def __init__(self):
    self.cards = []
    for num in range(1,13):
        for suit in ["H","C","S","D"]:
            # append a suit,num tuple
            self.cards.append( (suit,num) )
```

Attributes

- Begin with “self”, like `self.cards`.
- Global within the class.
- Invisible outside it (local to the class).

String representation

- Used by the print function; a string that shows the world the data in your class. Not always used.

```python
# string function; always called __str__
def __str__(self):
    outStr = ""
    for card in self.cards:
        suit = card[0]
        num = card[1]
        outStr = outStr + " " + suit+str(num)
    return outStr
```

Methods

- Other function definitions inside the class definition.

```python
# a method!
def shuffle(self):
    shuffle(self.cards)
```

Object Oriented Programming

- A philosophy about how to organize programs.
- Keep data organized into objects, along with methods to use or modify that data.
- Many languages, including Python, have a lot of syntactic sugar to make object oriented programming easier.
- Lots of modules (like GUI) are organized as collections of classes, so it’s nice to know what’s going on under the hood.
Dealing some hands