lec05

October 1, 2018

1 Python "Goodies," Weirdness

2 Built-in Types

- numbers: `int`, `float`, `bool`
- strings: `str` (and `bytes`)
- collections: `list`, `tuple`, `set`, `frozenset`
- associative arrays: `dict`

Each of these types has a number of useful built-in operations (see https://docs.python.org/3/library/stdtypes.html)

3 Pythonic Looping

```
In [ ]: for i in [0, 1, 2, 3, 4, 5]:
            print(i**2)
```

```
In [ ]: colors = ['red', 'green', 'blue', 'yellow', 'chartreuse', 'periwinkle']
        names = ['karl', 'duane', 'adam', 'pete']

        i = 0
        while i < len(colors):
            print(colors[i])
            i += 1
```

```
In [ ]: for i in range(len(colors)-1, -1, -1):
            print(colors[i])
```

```
In [ ]: for i in range(len(colors)):
            print(i, colors[i])
```

```
In [ ]: n = min(len(names), len(colors))
        for i in range(n):
            print(names[i], colors[i])
```

```
In [ ]: def poly_evaluate(coeffs, val):
            out = 0
            for i in range(len(coeffs)):
                out += coeffs[i] * val ** i
            return out

        print(poly_evaluate([7, 1, 2], 20))
```

```
In [ ]: for color in sorted(colors):
            print(color, end=' ')

        print()

        for color in sorted(colors, reverse=True):
            print(color, end=' ')

        # sort based on length?
```

4 Expressive One-Liners

```
In [ ]: result = 0
        for i in range(20):
            s = i**2
            result += s
        print(result)
```

```
In [ ]: def average(x):
            total = 0
            count = 0
            for i in x:
                total += i
                count += 1
            if count == 0:
                return 0
            else:
                return total / count

        average([1, 2, 3, 4])
```

```
In [ ]: a = [1, 2, 3, 4, 5]

        def nonnegative(x):
            sofar = True
            for i in x:
                if i < 0:
                    sofar = False
            return sofar


        def nondecreasing(x):
            sofar = True
            for i in range(len(x)-1):
                if x[i+1] <= x[i]:
                    sofar = False
            return sofar
```

```
In [ ]: # what is different about each of the forms below?

        a = [i**2 for i in range(10)]
        b = (i**2 for i in range(10))
        c = {i**2 for i in range(10)}
        d = {i:i**2 for i in range(10)}

        print(a)
        print(b)
        print(c)
        print(d)
```

```
In [ ]: o = ['cold', 'cord', 'word', 'ward', 'warm']
        all((len(a)==len(b) and sum(i!=j for i,j in zip(a, b))==1)
            for a, b in zip(o,o[1:]))
```

5 Aside: Tuple Unpacking

```
In [ ]: # swap two variables

        a = 7
        b = 8

        temp = a
        a = b
        b = temp

        print(a, b)

        def fibonacci(n):
```

```
        # state update
        x = 0
        y = 1
        for i in range(n):
            temp = y
            y = x + y
            x = temp
        return x

    fibonacci(7)
```

6 Some Weird (at first glance) Stuff

```
In [ ]: for n in range(2, 20):
            is_prime = True
            for x in range(2, n):
                if n % x == 0:
                    print(n, 'equals', x, '*', n//x)
                    is_prime = False
                    break
            if is_prime:
                print(n, 'is prime')
```

```
In [ ]: try:
            x = 'hello' + 1
        except:
            print('ERROR A')
```

7 Another Highlight: Complex Numbers

```
In [ ]: import math

        y = (-7)**0.5
        x = 2+3j

        print(y)
        print(y.real)
        print(y.imag)
        print(abs(y))

        # discrete Fourier transform of a given signal x[n]
        def dft(x):
            N = len(x)
            return [1/N * sum(x[n] * math.e**(-1j*2*math.pi*n/N*k) for n in range(N)) for k in

        dft([0, 1, 0, 0])
```