Function Examples
Announcements
Twenty-One Environment Diagram
**Twenty-One Rules**

Two players alternate turns, on which they can add 1, 2, or 3 to the current total

The total starts at 0

The game end whenever the total is 21 or more

The last player to add to the total loses

Some states are good; some are bad

(Demo)
Function Implementation Practice
A Slight Variant of Fall 2022 Midterm 1 3(b)

Implement nearest_prime, which takes an integer n above 5. It returns the nearest prime number to n. If two prime numbers are equally close to n, return the larger one. Assume is_prime(n) is implemented already.

```python
def nearest_prime(n):
    """Return the nearest prime number to n. In a tie, return the larger one.
    >>> nearest_prime(8)
    7
    >>> nearest_prime(11)
    11
    >>> nearest_prime(21)
    23
    """
    ___
    while True:
        if ___:
            ___
            if ___:
                k = -k
            else:
                ___
```

From the videos:
Read the description
Verify the examples & pick a simple one
Read the template
Annotate names with values from your chosen example
Write code to compute the result
Did you really return the right thing?
Check your solution with the other examples

From discussion:
Describe a process (in English) that computes the output from the input using simple steps.
Figure out what additional names you'll need to carry out this process.
Implement the process in code using those additional names.
Currying
Function Currying

```python
def make_adder(n):
    return lambda k: n + k
```

```python
>>> make_adder(2)(3)
5
>>> add(2, 3)
5
```

**Curry**: Transform a multi-argument function into a single-argument, higher-order function.
Example: Newton's Method (OPTIONAL)
Newton's Method Background

Quickly finds accurate approximations to zeroes of differentiable (smooth) functions

Application: a method for computing square roots, cube roots, etc.

The positive zero of \( f(x) = x^2 - a \) is \( \sqrt{a} \). (We're solving the equation \( x^2 = a \).)
Newton's Method

Given a function $f$ and initial guess $x$,

Repeatedly improve $x$:

1. Compute the value of $f$ at the guess: $f(x)$
2. Compute the slope of $f$ at the guess: $\text{slope}(f, x)$
3. Update guess $x$ to be:
   \[ x = x - \frac{f(x)}{\text{slope}(f, x)} \]

Finish when $f(x) = 0$ (or close enough)

How to find the square root of 2?

```python
>>> f = lambda x: x**2 - 2
>>> find_zero(f)
1.4142135623730951
```

Applies Newton's method