CIS 122

Throwing you for a loop
Survey Results

- **Pace - 5.4**
  - General agreement

- **Difficulty - 5.4**
  - More spread out

- **Clarity - 8.8**
Survey Results - Improvements

- More coding examples
  - More lecture examples?
  - More hands-on IDLE examples?

- More emphasis on syntax
  - Use Python syntax in speech

- Harder / Interesting / Extra Credit Problems
  - We now have the background to tackle bigger problems
  - If you want more, ask me personally

- Better dry erase pens
  - Check
Where are we?

- We can store values in variables
- We can store code in functions
- We can use recursion to run code arbitrary number of times
- We can compute anything that is computable!
Beyond Recursion

- Some problems are naturally recursive
  - Drawing fractals

- But sometimes, you just want to repeat a command
  - Drawing squares

- That's what loops are for
  - Repeatedly execute a block of code
Loops

- Two types of loops
  - while loops
    - Useful for more general problems
    - Loop until some condition is met
    - Can loop forever
  - for loops
    - Useful for more specific problems
    - Iterate over a sequence
    - Repeat a specific number of times
    - Much harder to loop forever
While Loops

- While some condition is true
  - Keep running block of code

- Very similar to if statement
  - If statement runs block once if condition is true
  - While loop runs block repeatedly while condition is true
Anatomy of a while loop

\[ x = 0 \quad \text{Initialization} \]

\[ \textbf{while } x < 10: \quad \text{Loop Condition} \]
\[ \ \quad \textbf{print } x \quad \text{Loop body} \]
\[ x = x + 1 \]
While Loops

● While condition is True, keep running body
  ○ What if condition is always true?

● Infinite loop
  ○ Similar to infinite recursion
  ○ But no limit on number of loops

● Sometimes an infinite loop is a good thing
  ○ IDLE shell
  ○ Operating systems

```python
x = 0
while x >= 0:
    print x
    x = x + 1
```

```python
x = 0
while True:
    print x
    x = x + 1
```
While Loops

- What if you need to break out of a loop early?
  - Use the break keyword
  - Stop running whatever loop you're in

```python
x = 0
while True:
    print(x)
    x = x + 1
    if x == 10:
        break
```
While Loops

● Avoid using break statements when you can
  ○ Tend to make code less clear
  ○ A good loop condition is far more readable

● If you use break statements, comment them well

```python
x = 0
while x < 10:
    print x
    x = x + 1

x = 0
while True:
    print x
    x = x + 1
    if x == 10:
        break
```
While Loop Practice

● Let's try solving an old problem in a new way

● Write a factorial function using a while loop
  ○ factorial(x) = 1 * 2 * 3 * ... * x

● What's the plan?
  ○ Need a task we can repeat
While Loop Practice

- **Initialization**
  - Initialize a counter variable to 1
  - Initialize a total variable to 1

- **Loop**
  - Multiply total by counter
  - Increment counter

- **Condition**
  - Stop when counter gets to x
def factorial(x):
    """Compute the product of all numbers from 1 to x"""
    # Initialization
    counter = 1
    total = 1
    while counter <= x:  # Condition
        total = total * counter  # Body
        counter = counter + 1
    return total  # After the loop finishes
While Loop Practice

● Your turn

● Implement $\text{collatz}(x)$ using a while loop
  ○ How many times do we need to perform HOTPO on $x$ before it reaches 1

● HOTPO
  ○ even $x \rightarrow x/2$
  ○ odd $x \rightarrow 3*x+1$