

Lab 3

- Review
 - Lab 2
 - Loops
 - Functions

Lab 2 Feedback

- Getting a little tougher in grading
 - Paying more attention to style (e.g., variable names), efficiency, readability, good output
 - High-level descriptions
 - More strict on adhering to problem specification
 - Demonstrate program **more than once** if gets input from user or outcome changes when run again
 - Find errors before I do!

Testing Discussion

- Consider what inputs could allow you to see different behaviors
- Consider how easily you can validate

Lab 2 Feedback: Common Issues

Which solution is more efficient?

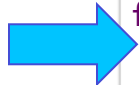
```
i=6
for j in range(1, 9):
    result = i % j
    print(i, "%", j, "=", result)
```

vs

```
for j in range(1, 9):
    i=6
    result = i % j
    print(i, "%", j, "=", result)
```

Lab 2 Feedback: Common Issues

Which solution is more efficient?



```
i=6
for j in range(1, 9):
    result = i % j
    print( i, "%", j, "=", result)
```

vs

```
for j in range(1, 9):
    i=6 ← Additional assignment each time through loop
    result = i % j
    print(i, "%", j, "=", result)
```

Lab 2 Feedback: Common Issues

Which solution is simpler?

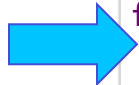
```
i=6
for j in range(1, 9):
    result = i % j
    print( i, "%", j, "=", result)
```

vs

```
i=6
j=0
for x in range(8):
    j = x + 1
    result = i % j
    print( i, "%", j, "=", result)
```

Lab 2 Feedback: Common Issues

Which solution is simpler?



```
i=6
for j in range(1, 9):
    result = i % j
    print( i, "%", j, "=", result)
```

vs

```
i=6
j=0
for x in range(8):
    j = x + 1
    result = i % j
    print( i, "%", j, "=", result)
```

More code makes
solution more difficult
to understand

Text's setText("text") method

- Instead of creating multiple Text objects, just use setText mutator method.
- For example:

```
text = Text( anchorPoint, "original directions")
...
text.setText("new directions")
```

Better Naming

- Consider which variable name is better:

```
circle = Circle(midPoint, 50)
```

```
bodyTop = Circle(midPoint, 50)
```

More Hints

- Debugging practices
 - Trace through the program as if you are the computer
 - Similar to some exam problems
 - Use print statements to display variables' values
 - Or, use Python visualizer to show how variables' values change

Review

- How do we call functions?
- How can we access functions from a module?

Problem: Animate Moving to User Click

- Use combinations of the method `move` and the function `sleep`
 - Need to `sleep` so that humans can see the graphics moving
 - Computer would process the `moves` too fast!
- `sleep` is part of the `time` module
 - Takes a `float` parameter representing *seconds* and pauses for that amount of time

Review: Designing for Change: Constants

- Special variables whose values are defined once and never changed
 - By convention, not enforced by interpreter
- By convention
 - A constant's name is all caps
 - Typically defined at top of program → easy to find, change
- Examples:
 - `NUMBER_OF_INPUTS = 5`

Review: Accumulator Design Pattern

1. Initialize accumulator variable
2. Loop until done
 - Update the value of the accumulator
3. Display result

Recall our example of adding up the user inputs...

Computational Thinking

- Learning how to think
 - Learning how to learn
 - Learning how to solve problems
- Process
 - Practice!
 - Review slides and examples after class
 - Run them in Python visualizer
 - Finding answers
 - Previous labs, handouts, ...
 - Asking questions
 - We talk you through our process

Drill good practice in early on smaller problems so that you are well-poised to handle the big problems!

Lab 3 Overview

- Practice Python programming
 - Loops
 - Constants
 - Animation with Graphics API
 - Functions

Exam 1 Questions?