## Lab 3 Feedback

- Continuing to get 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
  - Constants
  - Demonstrate program **more than once** if gets input from user or outcome changes when run again
    - Find errors before I do!

## Lab 3: Feedback

- Comments, good variable names on the OO graphics programming
- Compare:

```
# Draw the snow person's body
body2.draw(win)
```

vs

```
# Draw the second circle
circle2.draw(win)
```

## Program Organization

```
# high-level description
# author name

import statements

CONSTANT_DEFNS = …

program_statements ...
program_statements ...
program_statements …
```

## Lab Reminders

- Student assistants and professor are here to help BUT you should know how to think/learn
  - Review the slides and examples
  - Narrow down the issues so they're not too broad
  - Find a clear symptom of the error before asking how to fix
- You should NOT simply look at your neighbor's code and write down the answer
  - You can ask your neighbor questions and discuss solutions but copying answers is an honor code violation
- Grappling with a problem is part of the process
  - We will give advice on the process and how to approach problems

## Lab Learning Process

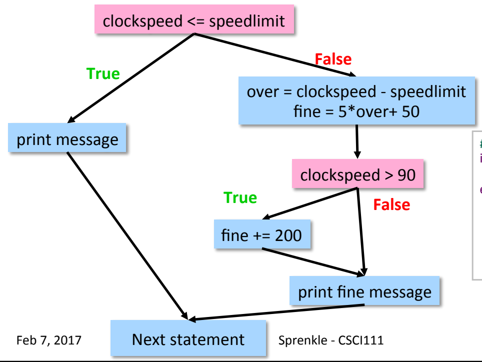
- More struggle does not mean higher grades
- BUT struggle does help you improve your process to arrive at solutions faster
  - What didn't work before? Don't do that again. Try something different
  - What did work before? Repeat that
- Approaches that work
  - Break problem into smaller pieces
    - don't try to solve whole problem at once
  - Good test cases
  - Asking questions: why was that output generated? Why did that behave like that, …

## Review

- How do we tell Python to make decisions?
- What is the syntax for such statements?
  - What are the alternatives and their meaning?
- How do you write a condition that is true only if two conditions are true?
- How do you write a condition that is true if at least one of two conditions is true?
- How do you write a condition that is true if a condition is false?

## Review: Testing with `if` Statements

- Make sure *at least* have test cases that execute each branch in control flow diagram
  - i.e., Each execution path is "covered"

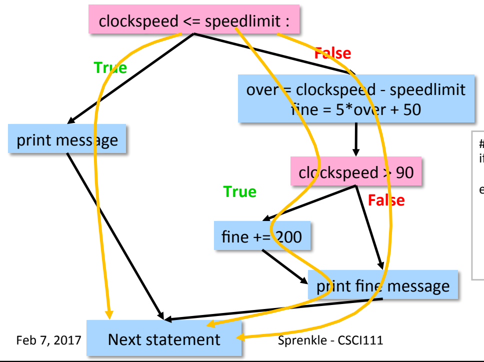


Three execution paths

```
# get user input
if clockspeed <= speedlimit:
   print "Continue …"
else:
   diff = clockspeed - speedlimit
   fine = 50 + 5 * diff
   if clockspeed > 90:
      fine += 200
   print "fined $" + str(fine) + "."
```

## Review: Testing with `if` Statements

- Make sure have test cases that execute each branch in control flow diagram
  - i.e., Each execution path is "covered"



Three execution paths

```
# get user input
if clockspeed <= speedlimit:
   print "Continue …"
else:
   diff = clockspeed - speedlimit
   fine = 50 + 5 * diff
   if clockspeed > 90:
      fine += 200
   print "fined $" + str(fine) + "."
```

## Review: Efficiency of `if` statements

- Efficiency: how much does the computer need to compute
- Which is more efficient?

```
if x < 0:
   print(x, "is negative")
if x >= 0:
   print(x, "is 0 or positive")
```

```
if x < 0:
   print(x, "is negative")
else:
   print(x, "is 0 or positive")
```

## Review: Efficiency of `if` statements

- Which is more efficient?

```
if x < 0:
   print(x, "is negative")
if x >= 0:
   print(x, "is 0 or positive")
```

Additional computation

```
if x < 0:
   print(x, "is negative")
else:
   print(x, "is 0 or positive")
```

More efficient

## Lab 4 Overview

- Conditional problems
- Due tomorrow