Lab 6

- Review Lab 5
- Review relevant content
- Lab 6

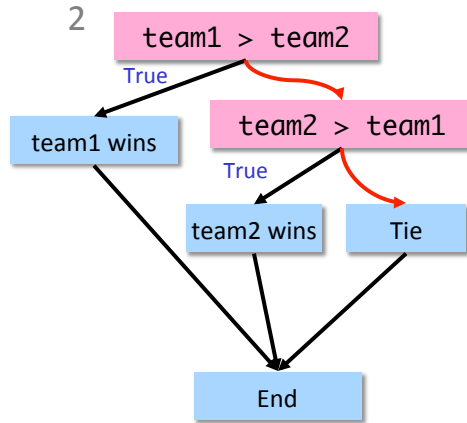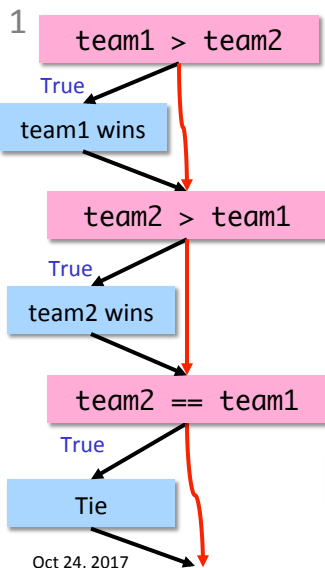# Common Issue: Inefficiency

```
if team1Score > team2Score:
    print("Team 1 wins!")
else:
    if team2Score < team1Score:
        print("Team 2 wins!")
    else:
        if team1Score == team2Score:
            print("They tied! We're going to overtime!")
```

Extra if statement, not necessary
Know when hit second else that the only possibility is a tie
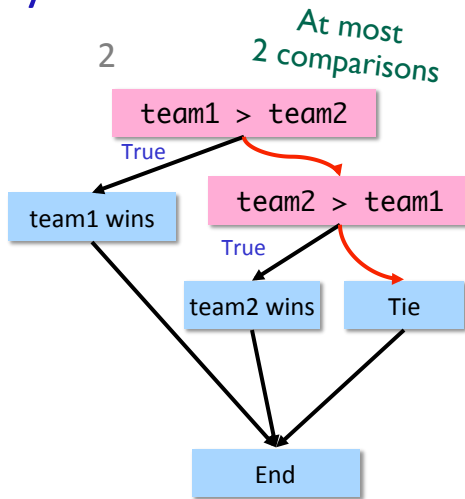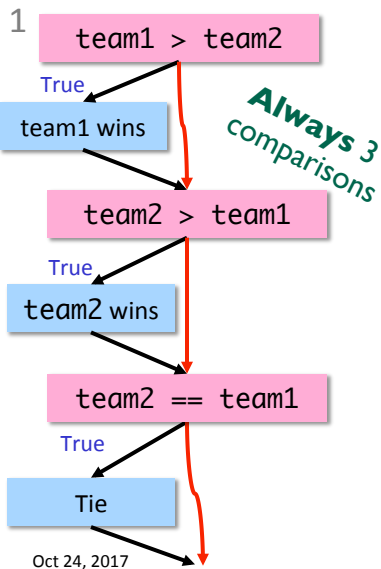
```
if team1Score > team2Score:
    print("Team 1 wins!")
else:
    if team2Score < team1Score:
        print("Team 2 wins!")
    if team1Score == team2Score:
        print("They tied! We're going to overtime!")
```

Problem 1, 2 Efficiency

1

team1 > team2

True

team1 wins

team2 > team1

True

team2 wins

team2 == team1

True

Tie

2

team1 > team2

True

team1 wins

team2 > team1

True

team2 wins   Tie

End

• How many conditions evaluated?

Problem 1, 2 Efficiency

1

team1 > team2

True

team1 wins

Always 3 comparisons

team2 > team1

True

team2 wins

team2 == team1

True

Tie

2

At most 2 comparisons

team1 > team2

True

team1 wins

team2 > team1

True

team2 wins   Tie

End

Problem 2 (& 3) Efficiency

Which tends to be more efficient?
How many conditions to evaluate?

```
team1 == team2
```
True

Tie

```
team2 > team1
```
True

2 wins   1 wins

End

```
team1 > team2
```
True

1 wins

```
team2 > team1
```
True

2 wins   Tie
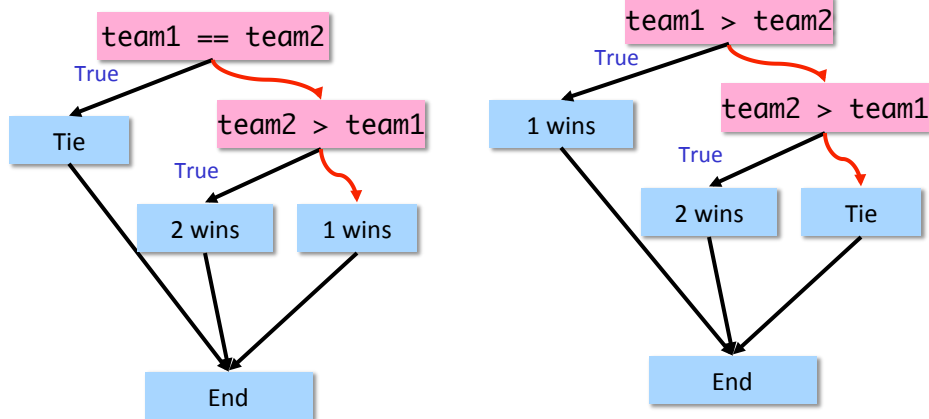
End
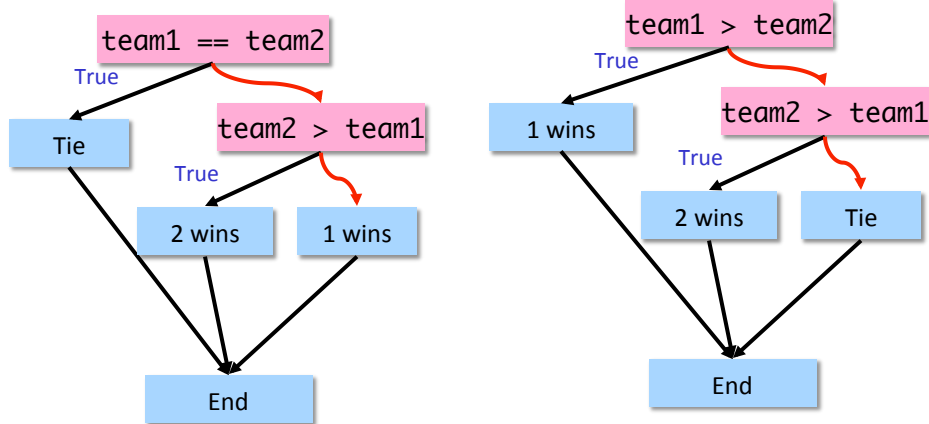
Problem 2 (& 3) Efficiency

Equality is a rare condition; on average, will always need to check second condition.

More common case. May only need to check one condition.

```
team1 == team2
```
True

Tie

```
team2 > team1
```
True

2 wins   1 wins

End

```
team1 > team2
```
True

1 wins

```
team2 > team1
```
True

2 wins   Tie

End

Adding to Development Process

- Last development step:
  - ➤ Assess your program again after it works
  - ➤ Is it efficient? Is it readable? Can I simplify?

Lab 4 – Greatest Hits: Less-Complicated Approaches for Customized Display

- Correct but more complicated solution to handling customized display

Other, similar examples in submissions

```
if albums == 1 and extraTracks == 0:
    print("Your album requires", albums, "cd")
elif albums == 1 and extraTracks > 0:
    print("Your album requires", albums, "cd")
    print(extraTracks, "tracks will have to wait for
                        the next Greatest Hits album")
elif albums > 1 and extraTracks > 0:
    print("Your album requires", albums, "cds")
    print(extraTracks, "tracks will have to wait for
                        the next Greatest Hits album")
elif albums > 1 and extraTracks == 0:
    print("Your album requires", albums, "cds")
```

Lab 4 – Greatest Hits: Less-Complicated Approaches for Customized Display

- Less complicated solution
  - Simpler logic, conditions
  - Less duplicated code

```python
if albums == 1:
    print("Your album requires", albums, "CD.")
else:
    print("Your album requires", albums, "CDs")

if extraTracks > 1:
    print(extraTracks, "tracks will have to wait for
                    the next Greatest Hits album")
elif extraTracks==1:
    print(extraTracks, "track will have to wait for
                    the next Greatest Hits album")
```

Relational Operators

- Reminder: instead of, for example,

$$num < 0 \text{ or } num > 0$$

can use

$$num \mathrel{!=} 0$$

World Series Extensions

- Simulate innings
  - Simulate outs
- Change odds based on home/visiting team
- Dynamically change odds based on who won/lost already in the series

- Today: could stop the series after a team reaches four wins. How?

A new trick: sys module

- sys.exit(<status code>)
  - You can typically use a status code of 0.
- Example:

```
import sys

age = int(input("What is your age? "))
if age < 0:
    print("You must enter a positive number.")
    sys.exit(0)
…
```

str Review

- How can we combine strings?
- How can we find out how long a string is?
- How can you tell if one string is contained in another string?
- How can we find out the character at a certain position?
- How can we iterate through a string?
- How do you call a method on a string?

Lab 6

- Advanced conditions
- Indefinite Loops
- Text-based problems