Lecture 11

Topics to be covered:

Chapter 8:
- Section 8.1 If-else branches (general)
- Section 8.2 If-else statement
- Section 8.3 More if-else
- Section 8.4 Equality and relational operators

If-else branches (general)

In many circumstances when we write a program we need the ability to check conditions and change the behavior of the program accordingly.

Selection statements or *conditional statements*, give us this ability.

If-else branches (general)

In many circumstances when we write a program we need the ability to check conditions and change the behavior of the program accordingly.

Selection statements or *conditional statements*, give us this ability.

Example: Let's look through the following code

```
if my_class_average > 1:
  print(“I passed the class! Hooray!”)
else:
  print("Bummer! I will have to re-take this
class!”)
```

If-else branches (general)

Consider another code fragment:

```
x = int(input("Enter an integer value:"))
y = int(input("Enter another integer value:"))

if x > y:
   a = x

if x < y:
   a = y
else:
   print("They are equal!")
```

If-else branches (general)

Consider another code fragment:

```
x = int(input("Enter an integer value:"))
y = int(input("Enter another integer value:"))

if x > y:
   a = x

if x < y:
   a = y
else:
   print("They are equal!")
```

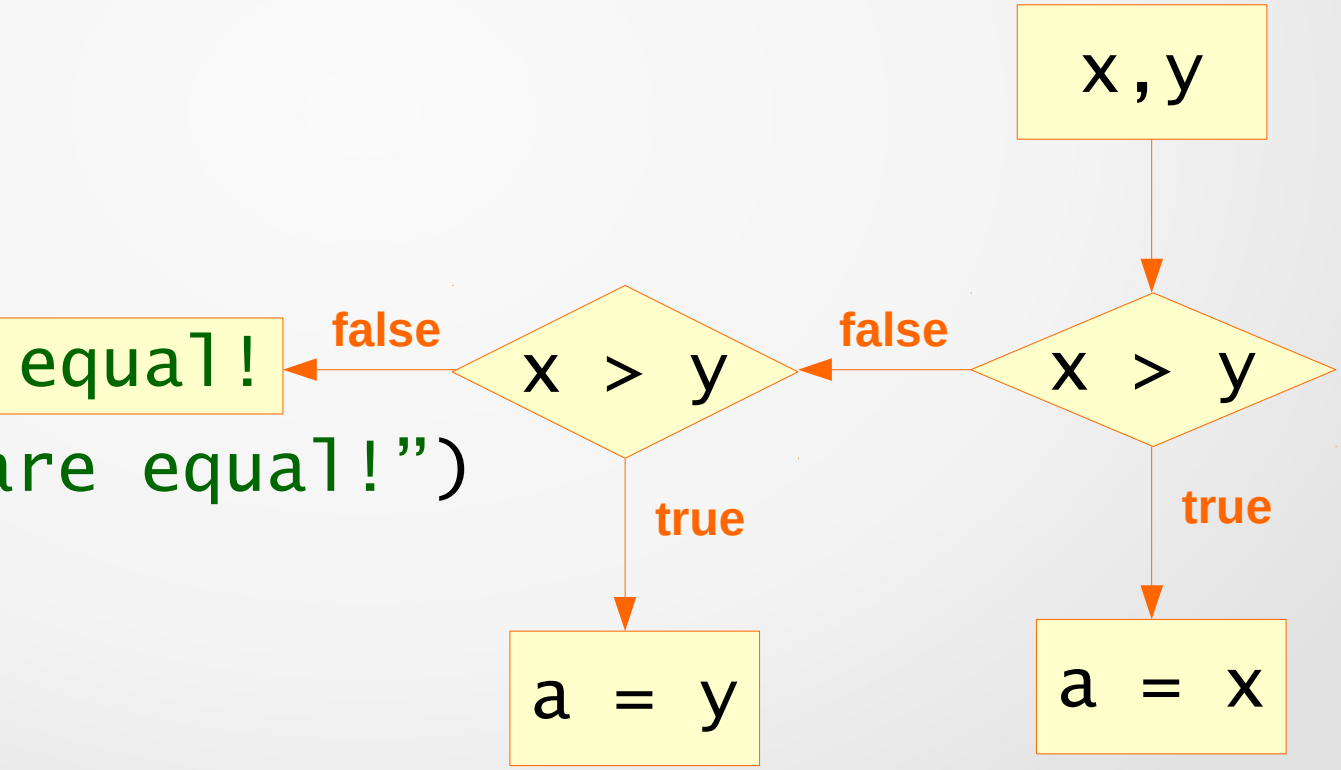
conditions
(evaluated to a Boolean value: True or False)

If-else branches (general)

```
x = int(input("Enter an integer value:"))
y = int(input("Enter another integer value:"))

if x > y:
   a = x

if x < y:
   a = y
else: They are equal!
   print("They are equal!")
```



If-else branches (general)

If we type the following commands in the Python shell, we will get the responses highlighted with blue

```
>>> 2==2
True

>>> 2<3
True

>>> 3>7
False

>>> 5>9 or 2<3
True
```

If-else statement

Multi-branch if-else statements

Let's write a program that will report the grade for the test, given a numeric score.

```
test_score = float(input("Enter test score:"))
if test_score >= 90:
   print("This is an A grade!")
if 80 <= test_score < 90:
   print("This is a B grade!")
if 70 <= test_score < 80:
   print("This is a C grade!")
if 60 <= test_score < 70:
   print("This is a D grade!")
else: print("Unfortunately this is an F grade")
```

If-else statement

Multi-branch if-else statements

Let's write a program that will report the grade for the test, given a numeric score.

```
test_score = float(input("Enter test score:"))
if test_score >= 90:
   print("This is an A grade!")
if 80 <= test_score < 90:
   print("This is a B grade!")
if 70 <= test_score < 80:
   print("This is a C grade!")
if 60 <= test_score < 70:
   print("This is a D grade!")
else: print("Unfortunately this is an F grade")
```

If-else statement

Multi-branch if-else statements

Let's write a program that will report the grade for the test, given a numeric score.

```
test_score = float(input("Enter test score:"))
if test_score >= 90:
   print("This is an A grade!")
elif 80 <= test_score < 90:
   print("This is a B grade!")
elif 70 <= test_score < 80:
   print("This is a C grade!")
elif 60 <= test_score < 70:
   print("This is a D grade!")
else: print("Unfortunately this is an F grade")
```

only one branch will execute!

Equality and relational operators

Equality operators

An equality operator checks whether two operands' values are the same (**==**) or different (**!=**).
Note that equality is **==**, not just **=**.

Equality operators	Description	Example (assume x is 3)
==	a **==** b means a is equal to b	x == 3 is true x == 4 is false
!=	a **!=** b means a is not equal to b	x != 3 is false x != 4 is true

An expression evaluates to a *Boolean value*.
A Boolean is a type that has just two values: True or False.

Equality and relational operators

Relational operators

A relational operator checks how one operand's value relates to another, like being greater than.

Relational operators	Description	Example (assume x is 3)
<	a **<** b means a is less than b	x < 4 is true x < 3 is false
>	a **>** b means a is greater than b	x > 2 is true x > 3 is false
<=	a **<=** b means a is less than or equal to b	x <= 4 is true x <= 3 is true x <= 2 is false
>=	a **>=** b means a is greater than or equal to b	x >= 2 is true x >= 3 is true x >= 4 is false

Equality and relational operators

Operator chaining

Python supports *operator chaining*.

Example: `a < b < c`
determines whether **b** is greater-than **a** but less-than **c**.

Chaining performs comparisons left to right,
evaluating `a < b` first.

- If the result is true, then `b < c` is evaluated next.
- If the result of the first comparison `a < b` is false, then there is no need to continue evaluating the rest of the expression.

Equality and relational operators

In-class work:

See items 1-5 in the handout