Data Scientist
Codecademy
Python Fundamentals
Python Control Flow

Control Flow

Introduction to Control Flow

In Python, the script will execute from the top down, until there is nothing left to run. It is our job to include gateways, known as conditional statements, to tell the computer when it should execute certain blocks of code. If these conditions are met, then run this function.

Boolean Expressions

In order to build control flow, we need to check if something is true or not. A Boolean expression is a statement that can either be True or False.

Relational Operators

We can create a Boolean expression by using relational operators.

Relational operators compare two items and return either True or False. For this reason they are also called comparators.

- Equals `==`
- Not Equals `!=`

These operators compare two items and return True or False if they are equal or not.

While using relational operators be mindful about the data type.

Boolean Variables

True and False have a special type: bool

True and False are the only bool types, and any variable assigned one of these values is called a Boolean variable.

If Statement

if <condition == true>, then <do something>

In Python we use : instead of then

```
if is_raining:
    print("Bring an umbrella")
```

More Relational Operators

- `>` greater than
- `>=` greater than or equal to
- `<` less than
- `<=` less than or equal to

Boolean Operators: and

Often we would like to check multiple conditions which will require multiple Boolean expression which we can use to make a larger Boolean expression using Boolean operators.

These operators also known as logical operators combine smaller Boolean expressions into larger Boolean expressions.

There are basically three Boolean operators:

- And
- Or
- Not

and combines two Boolean expressions and evaluates as True, if both its components are True, but False otherwise.

Boolean Operators: or

The Boolean operator or combines two expressions into a larger expression that is True if either component is True.

Boolean Operators: not

This operator on applying reverses the Boolean value.

Else Statements

Using multiple if statements makes code cluttered and clunky.

else statements allow us to elegantly describe what we want our code to do when certain conditions are not met.

else statements always appear in conjunction with if statements.

```
if age >= 13:
  print("Access granted.")
else:
  print("Sorry, you must be 13 or older to watch this movie.")
```

Else if Statements

Like if and else we also have elif statements. An elif statement checks another condition after the previous if statements conditions aren’t met and is short for else if.

```
print("Thank you for the donation!")

if donation >= 1000:
  print("You've achieved platinum status")
elif donation >= 500:
  print("You've achieved gold donor status")
elif donation >= 100:
  print("You've achieved silver donor status")
else:
  print("You've achieved bronze donor status")
```