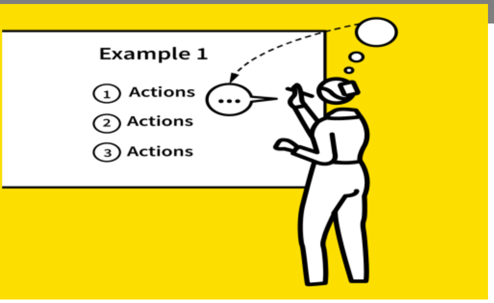
Learning Goals/Objectives

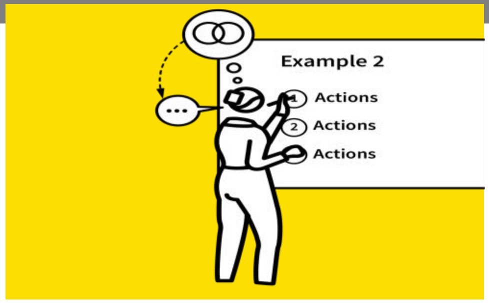
Be able to read, comprehend, trace, adapt and create Python code using selection that:

- Uses Boolean operators with multiple conditions.
- Checks if a number is inside or outside a range.
- Nests selection statements inside each other.

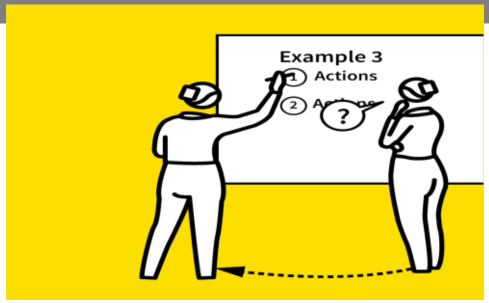
HOW THE TASKS HELP YOU DEVELOP YOUR SKILLS



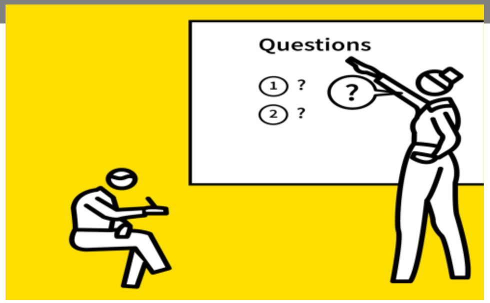
FULLY WORKED TO INTRODUCE THE METHOD OR IDEAS



FULLY WORKED FOR REINFORCEMENT



PARTIALLY WORKED FOR STUDENTS TO FINISH OFF



CUED START FOR STUDENT COMPLETION

COMPLETED INDEPENDENTLY

PREDICT & RUN → INVESTIGATE → MODIFY → MAKE

NOT MY CODE → PARTLY MY CODE → ALL MY CODE

More than one Boolean Operator

Resources created by Andy Colley (@MrAColley)

Boolean Operators

and - Checks at least two conditions. Returns true if **all** conditions are true

```
if hair == ‘blonde’ and eyes != ‘blue’
and hasPet == true:
    Stand up
else:
    Sit down
```

Boolean Operators

or - Checks at least two conditions. Returns true if **at least one** condition is true

```
if hair == ‘blonde’ or eyes != ‘blue’ or
hasPet == true:
   Stand up
else:
   Sit down
```

repl.it

Boolean Operators

not - Inverts the value from the condition. Returns false if the condition is true and true if the condition is false.

```
if not (hair == ‘brown’):
    Stand up
else:
    Sit down
```

Multiple Booleans - How To Code

Use **and** or **or** between the conditions.

1. Set your data, either by assignment or input.

2. Put the Boolean operator between the conditions in your code

```
num1 = 23

if num1 < 50 and num1 < 100:
```

Resources created by Andy Colley (@MrAColley)

Numbers in a Range (Validation)

repl.it

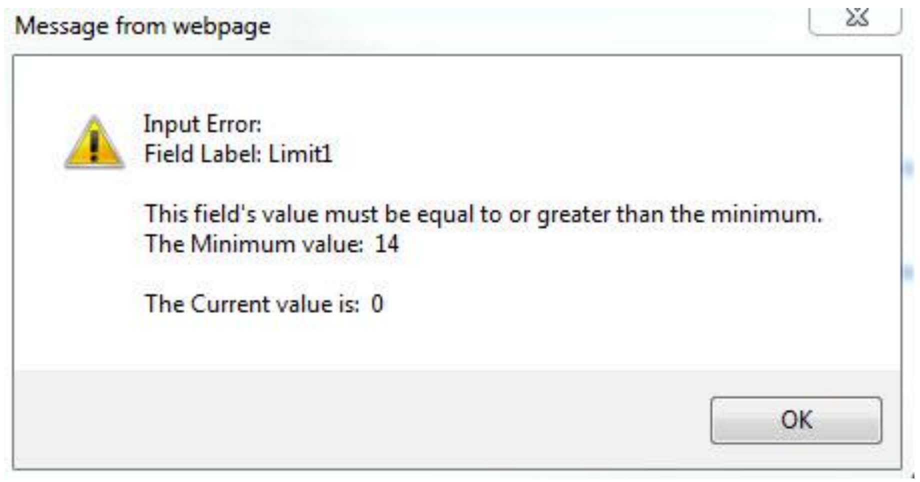
Resources created by Andy Colley (@MrAColley)

Validation

Checking whether data entered is **reasonable, sensible** and **allowable.**

Sometimes you only want to allow the user to input certain numbers.

We can use selection and Boolean conditions to produce error messages if they don't.



Input Error:
Field Label: Limit1

This field's value must be equal to or greater than the minimum.
The Minimum value: 14

The Current value is: 0

Validation inside a range - how to code

Using selection will allow you to create success/error messages. It **won't** make the user try again - we're building up to that.

1. Get the input.

```
num1 = input("Enter a number between 1
and 10")

if num1 >= 1 and num1 <= 10:
    print("Number in range")
else:
    print("Number not in range")
```

2. Use >= the lowest allowable number **and** <= the highest allowable number.

3. Success message if it is in the range. Else error message.

Validation outside a range - how to code

Using selection will allow you to create success/error messages. It **won’t** make the user try again - we’re building up to that.

1. Get the input.

```
num1 = input(“Enter a number between 1
and 10”)

if num1 < 1 or num1 > 10:
    print(“Number not in rang
else:
    print(“Number in range”)
```

2. Use < the lowest allowable number **or** > the highest allowable number.

3. Error message if it is not in the range. Else success message.

Nesting

Resources created by Andy Colley (@MrAColley)

Nesting

Nesting is putting programming structures inside each other.

For example, an if inside an if.

```
num1 = 53
if num1 > 10:
     print(“More than 10”)

     if num1 > 30:
          print(“ and more than 30”)
     else:
          print(“ but not more than 30”)
```

Nesting Selection - How To Code

1. Create your first selection statement as normal.

```
num1 = 53
if num1 > 10:
    print(“More than 10”)

    if num1 > 30:
        print(“ and more than 30”)
    else:
        print(“ but not more than 30”)
```

TAB

2. Nest the next statement inside the fist by using ‘Tab’ to indent it.

3. This statement will only run if the condition in the first ‘if’ is true.

repl.it

Resources created by Andy Colley (@MrAColley)