Boolean Values

In Python, *any* expression can be interpreted as either `True` or `False`. Knowing this feature helps you avoid common programming mistakes.

Model 1 Truthy or Falsy

In Python, most values are considered to be "true" when used as a Boolean. However, some values are considered to be "false." **Read the first two questions before completing the table.**

Python value	Predicted output	Actual output
`123`		
`0`		
`"abc"`		
`4.56`		
`""`		
`"0"`		
`True`		
`"True"`		
`False`		
`"False"`		
`0.0`		
`"0.0"`		

Questions (15 min) **Start time:**

1. Run the following code snippet, and use the output to complete the first row of the table.

```
value = 123
if value:
    print("Truthy")
else:
    print("Falsy")
```

2. As a team, predict the output (Truthy or Falsy) for the other values in the table. Then test each value using the code snippet from the previous question. Record your results in the table.

Manager: Spend no more than 5 minutes on this question.

3. Using complete sentences, summarize what kinds of values are considered to be `True` in Python (and what kinds of values are considered to be `False`).

4. Predict the value (`True` or `False`) of the following expressions. Then check your answers using a Python Shell.

Python value	**Predicted output**	**Actual output**
`bool(5)`		
`bool("")`		
`not 5`		
`not ""`		

5. Rewrite the following `if` statements without using `==` or `!=` (or any other operator).

a)
```
if value != 0:
    print("Not zero")
```

b)
```
if happy == True:
    print("You know it!")
```

c)
```
if absent == False:
    print("Good job")
```

d)
```
if answer != "":
    print("Thank you")
```

e)
```
if answer == "":
    print("Try again")
```