The Monty Hall Problem
Monty Hall Problem
A. Switch?
B. Stay?
C. Doesn’t matter
Comparison
Comparison Operators

The result of a comparison expression is a bool value

```
x = 2           y = 3
x > 1           x > y          y >= 3
x == y          x != 2         2 < x < 5
```

Assignment statements

Comparison expressions

(Demo)
George Boole

*The Laws of Thought* (1854)

No general method for the solution of questions in the theory of probabilities can be established which does not explicitly recognise, not only the special numerical bases of the science, but also those universal laws of thought which are the basis of all reasoning, and which, whatever they may be as to their essence, are at least mathematical as to their form.
Combining Comparisons

Boolean operators can be applied to `bool` values

\[
\begin{align*}
a & = \text{True} & b & = \text{False} \\
\text{not } b & \quad a \text{ or } b & a \text{ and not } b \\
a \text{ and } b & \quad \text{not (a or b)} & b \text{ and } b
\end{align*}
\]

Evaluate to True

Evaluate to False

(Demo)
Aggregating Comparisons

Summing an array or list of bool values will count the True values only.

\[
1 + 0 + 1 = 2 \\
\text{True} + \text{False} + \text{True} = 2 \\
\text{sum([1 , 0 , 1 ])} = 2 \\
\text{sum([True, False, True])} = 2
\]

(Demo)
Random Selection
Random Selection

`np.random.choice`

- Selects at random
- with replacement
- from an array
- a specified number of times

```
np.random.choice(some_array, sample_size)
```

(Demo)
Discussion Question

\[
d6 = \text{np.arange}(1, 6+1)
\]

What results from evaluating the following 2 expressions? Are they the same? Do they describe the same process?

\[
\text{np.random.choice}(d6, 1000) + \text{np.random.choice}(d6, 1000) \\
2 \times \text{np.random.choice}(d6, 1000)
\]
Control Statements
Control Statements

These statements *control* the sequence of computations that are performed in a program

- The keywords *if* and *for* begin control statements
- The purpose of *if* is to define computations that can choose different behaviors
- The purpose of *for* is to perform a computation for every element in a collection

(Demo)