DATA 8

Spring 2022

Lecture 13

Conditionals and Iteration

Announcements

Comparison and Booleans

Comparison Operators

The result of a comparison expression is a `bool` value

```
x = 2                y = 3
```

Assignment statements

```
x > 1                x > y                y >= 3

x == y               x != 2               2 < x < 5
```

Comparison expressions

Aggregating Comparisons

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

```
1    + 0     + 1             == 2
True + False + True          == 2
sum([1    , 0     , 1    ])  == 2
sum([True, False, True])     == 2
```

(Demo)

Applying a Function to a Row

Rows

A row of a table has items and can be aggregated.

```
r = t.row(0) # r is the row at index 0
r.item(1)    # item can take an index or label
sum(r)       # Also: np.average, min, max, etc.
```

Apply with One Argument

`t.apply(f)` for a table `t` and function `f` creates an array of the results of applying `f` to each *row* of `t`.

E.g., `t.apply(sum)` would return the sum of each row as an array.

(Demo)

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 functions that choose different behavior based on their arguments

(Demo)

Random Selection

Random Selection

`np.random.choice`

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

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

(Demo)

Appending Arrays

A Longer Array

- **`np.append(array_1, value)`**
  - new array with `value` appended to `array_1`
  - `value` has to be of the same type as elements of `array_1`
- **`np.append(array_1, array_2)`**
  - new array with `array_2` appended to `array_1`
  - `array_2` elements must have the same type as `array_1` elements

(Demo)

Iteration

`for` Statements

- `for` is a keyword that begins a multiline `for` statement.
- Executing a `for` statement performs a computation for every element in a list or array.
- A common special case is to perform a computation a fixed number of times.

(Demo)

Anatomy of a for loop

Example:

variable name

array of values

```
for item in some_array:
    print(item)
```

indent

code to evaluate in each iteration of for loop

Optional: Advanced `where`

A Closer Look at `where`

`t.where(array_of_bool_values)`

returns a table
with only the rows of `t` for which
the corresponding `bool` is `True`.

(Demo)