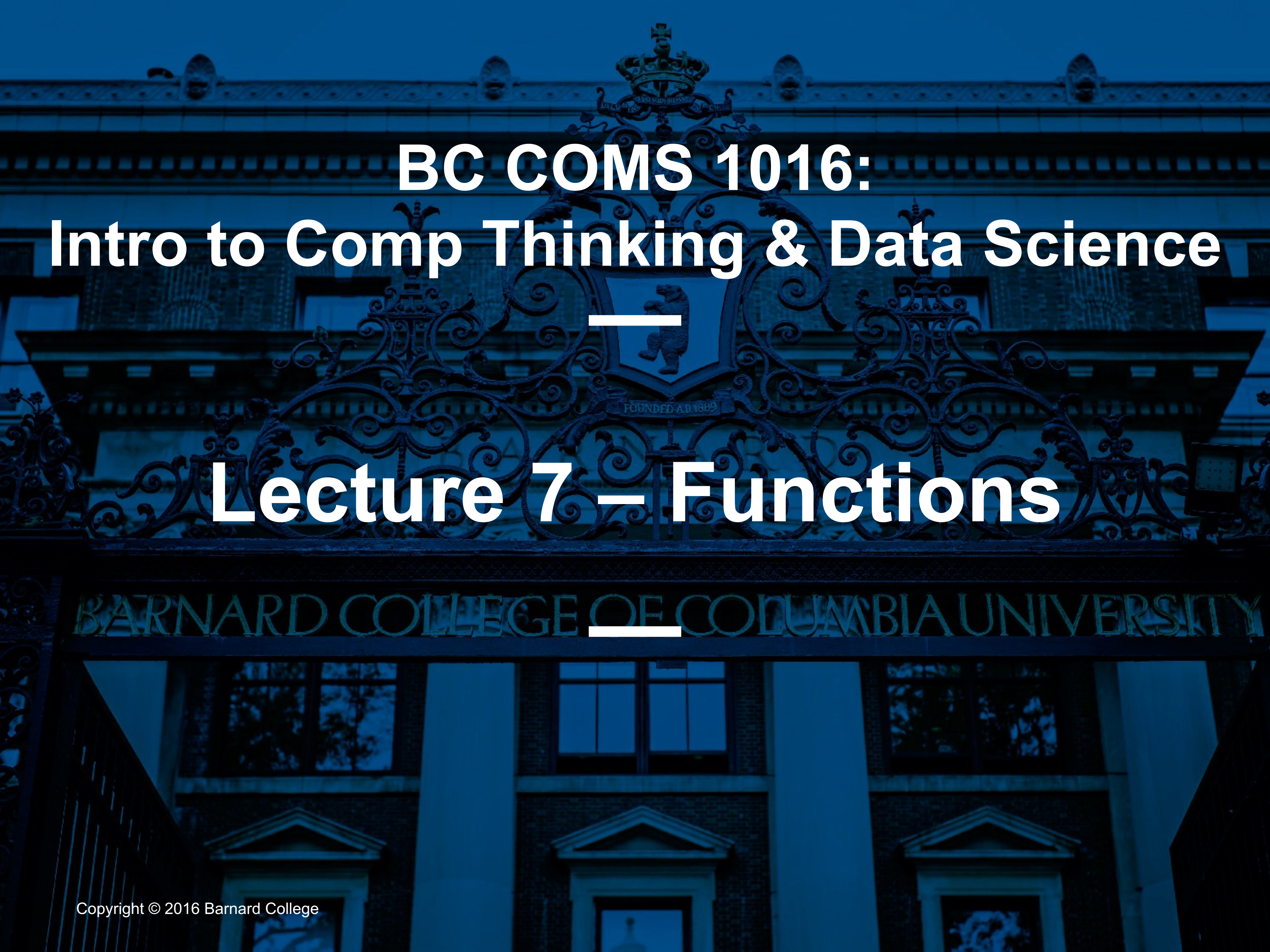


BC COMS 1016: Intro to Comp Thinking & Data Science

Lecture 7 – Functions

Announcements

- HW02 - Table Manipulation & Visualization:
  - Due Monday (02/14)
- Lab 03 - Functions and Visualizations
  - Due Monday (02/14)
- HW03 - Functions, Histograms, and Groups
  - Due Monday (02/21)
- Checkpoint/Project 1:
  - Paired assignment that covers the previous section of the course material
  - Released tonight or tomorrow and due 2 weeks (Friday 02/25)

Functions

Anatomy of a Function

- Name
- Parameters / Argument Names
- Body
- Return Expression

Example Function

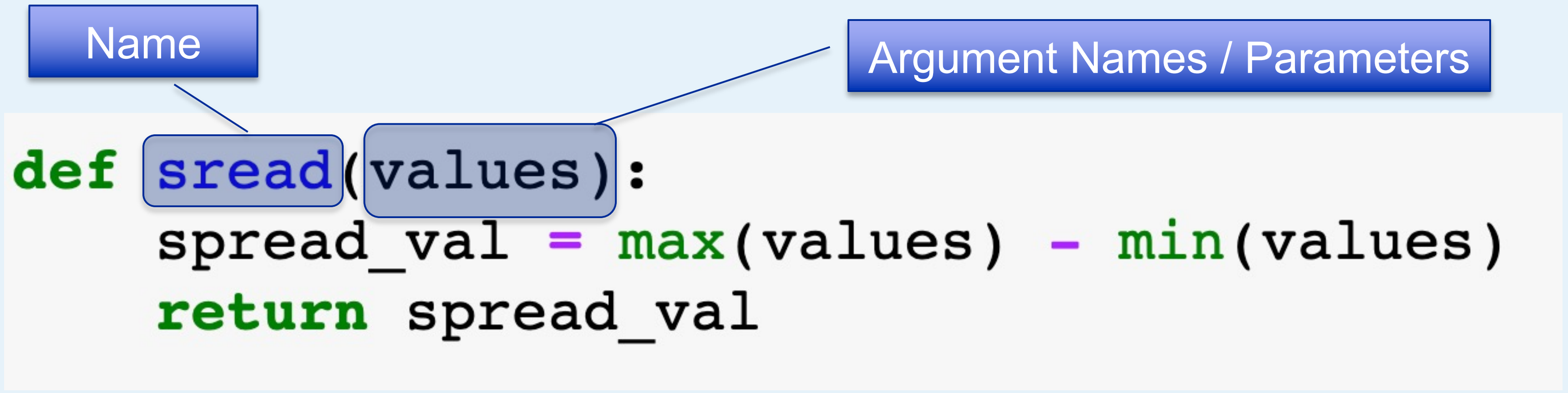
```python
def sread(values):
    spread_val = max(values) - min(values)
    return spread_val
```

Example Function - Name

Name

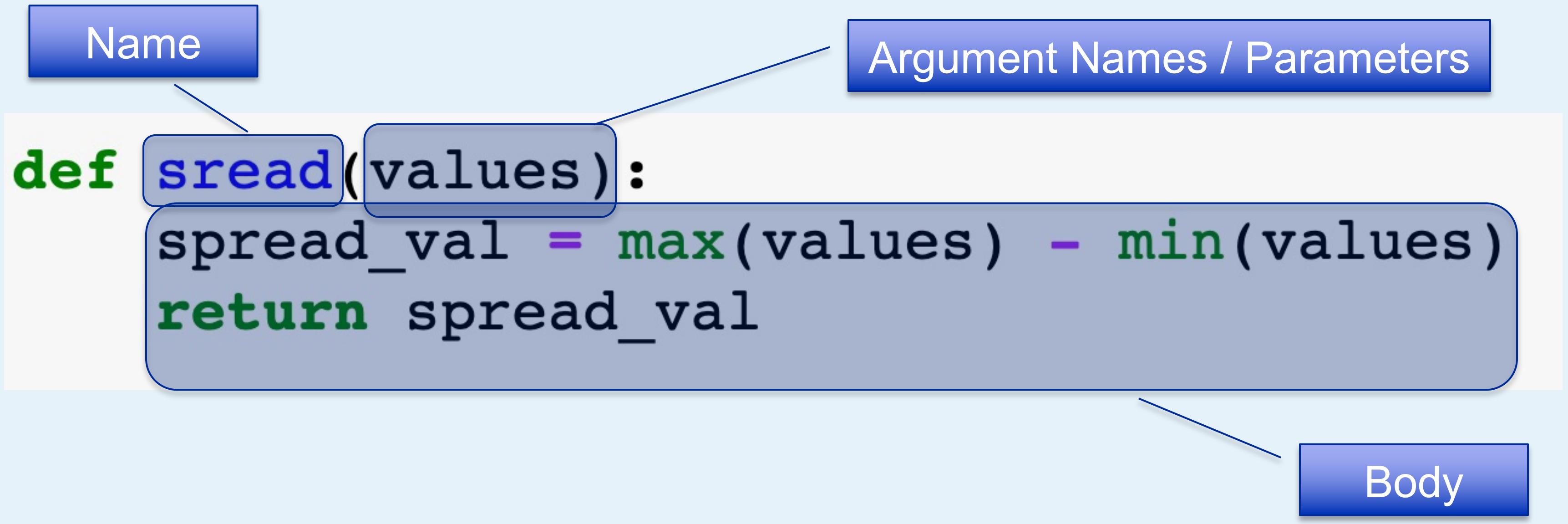
```
def sread(values):
    spread_val = max(values) - min(values)
    return spread_val
```

Example Function – Argument Names/Parameters



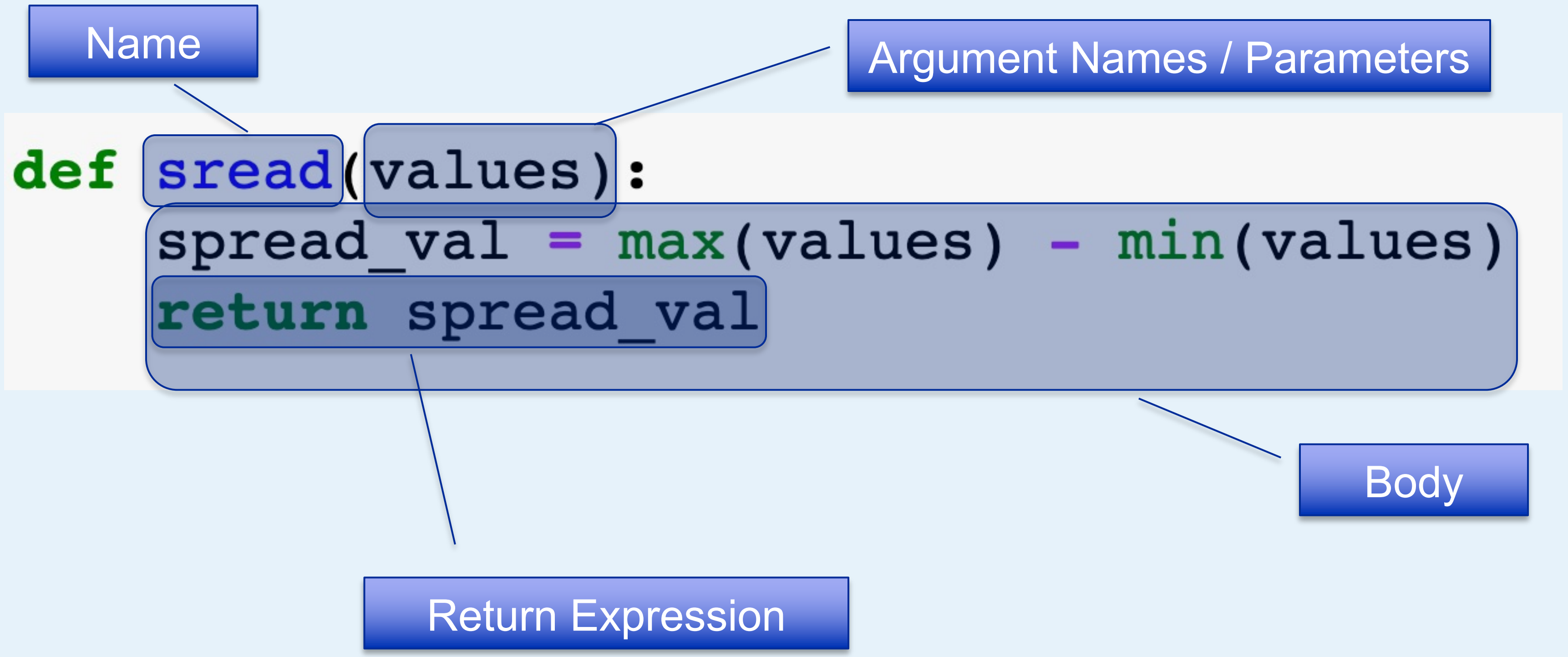
```python
def sread(values):
    spread_val = max(values) - min(values)
    return spread_val
```

Example Function - Body



```python
def sread(values):
    spread_val = max(values) - min(values)
    return spread_val
```

Example Function – Return expression


```
def sread(values):
    spread_val = max(values) - min(values)
    return spread_val
```

What does this function do?

```
def f(s):
    return np.round(s / sum(s) * 100, 2)
```

- What kind of input does it take?
- What output will it give?
- What's a reasonable name?

Applying Functions to Columns

The apply method creates an array by calling a function on every element in input column(s)

- First argument: Function to apply
- Other arguments: The input column(s)

```
table_name.apply(function_name, 'column_label')
```

Grouping by One Column

The group method aggregates all rows with the same value for a column into a single row in the resulting table.

- First argument: Which column to group by
- Second argument: (Optional) How to combine values

- len — number of grouped values (default)
- list — list of all grouped values
- sum — total of all grouped values

Lists as Generic Sequences

A list is a sequence of values (just like an array), but the values can all have different types

```
[2+3, 'four', Table().with_column('K', [3, 4])]
```

- Lists can be used to create table rows.
- If you create a table column from a list, it will be converted to an array automatically

Grouping by Multiple Columns

The group method can also aggregate all rows that share the combination of values in multiple columns

- First argument: A list of which columns to group by
- Second argument: (Optional) How to combine values

Pivot Tables

- Cross-classifies according to two categorical variables
- Produces a grid of counts or aggregated values
- Two required arguments:
  - First: variable that forms column labels of grid
  - Second: variable that forms row labels of grid
- Two optional arguments (include **both** or **neither**)
- values=’column_label_to_aggregate’
- collect=function_to_aggregate_with

Group vs Pivot

Pivot

- One combo of grouping variables **per entry**
- **Two** grouping variables: columns and rows
- Aggregate values of **values column**
- Missing combos **= 0 (or empty string)**

Group

- One combo of grouping variables **per row**
- **Any number** of grouping variables
- Aggregate values of **all other columns** in table
- Missing combos **absent**

Joining Two Tables

```
tblA.join(colA, tblB, colB)

tblA.join(colA, tblB)
```

Tomorrow’s Reading

- Chapter 9.1 – 9.3
- Conditionals & Randomness