Part 2.3:
Python: Flow Control, Conditional Statements, and Dictionaries

Dana L Carper and Travis J Lawrence

Quantitative and Systems Biology

University of California, Merced

Flow Control

- Flow control is the process of making decisions based on the value of one or more variables.

Flow Control

- Flow control is the process of making decisions based on the value of one or more variables.
- This allows you to run different blocks of code based on these decisions

Flow Control

- Flow control is the process of making decisions based on the value of one or more variables.
- This allows you to run different blocks of code based on these decisions
- We have already introduced you to one method of flow control

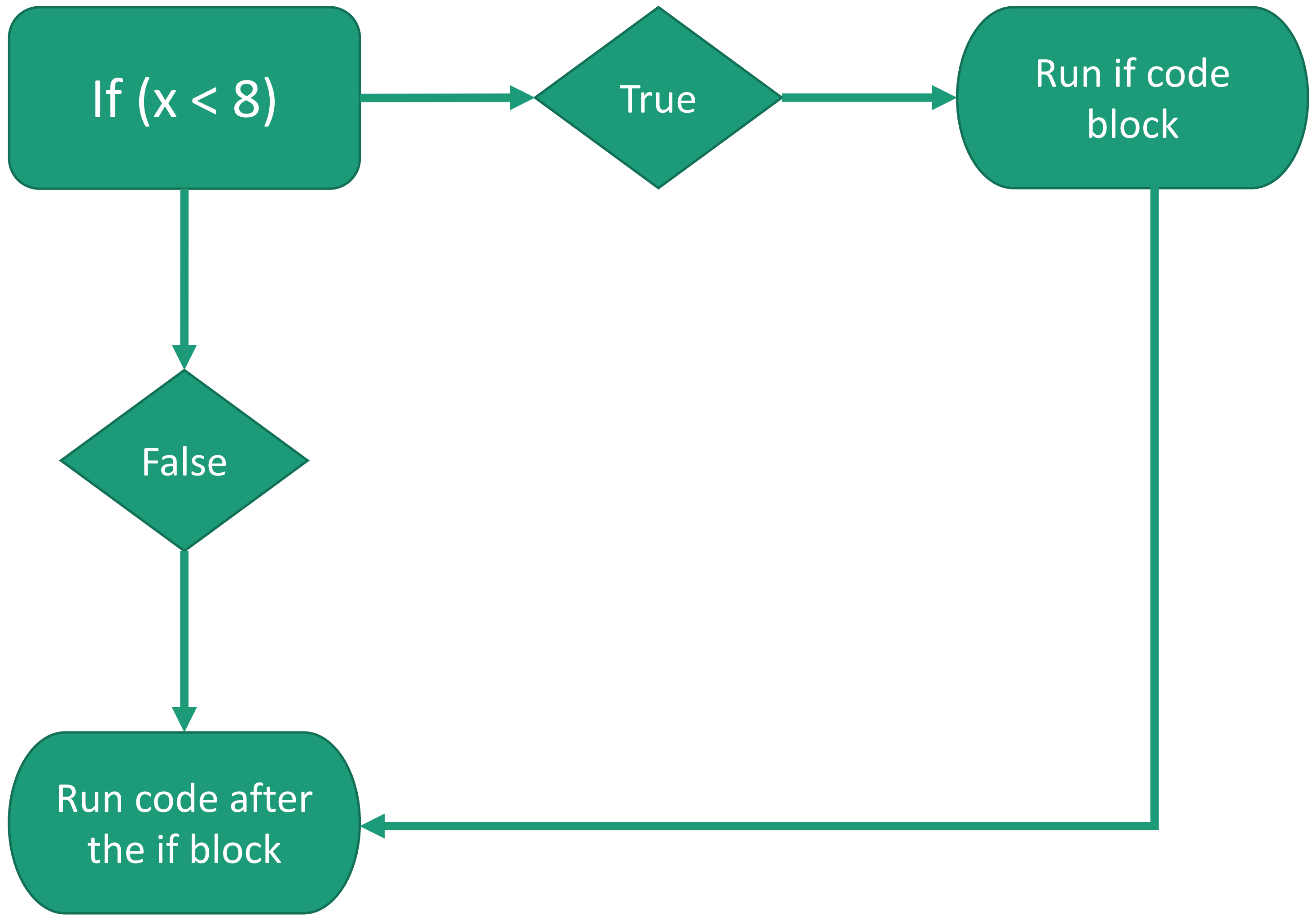
Flow Control

- Flow control is the process of making decisions based on the value of one or more variables.
- This allows you to run different blocks of code based on these decisions
- We have already introduced you to one method of flow control
    - Loops

Flow Control

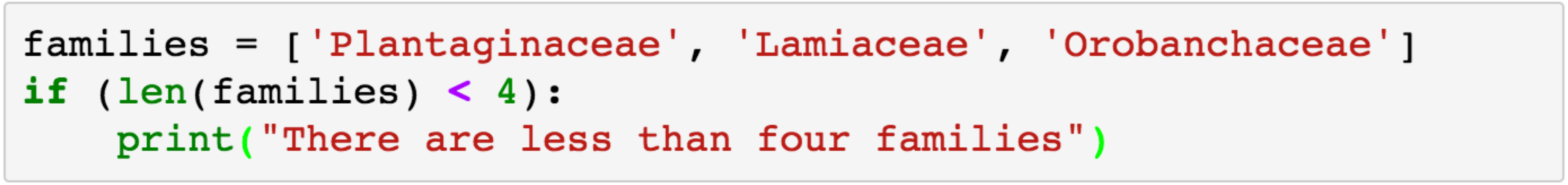
- Flow control is the process of making decisions based on the value of one or more variables.
- This allows you to run different blocks of code based on these decisions
- We have already introduced you to one method of flow control
  - loops
  - If/elif/else statements

Flow Control: if statements



Flow Control: if statements

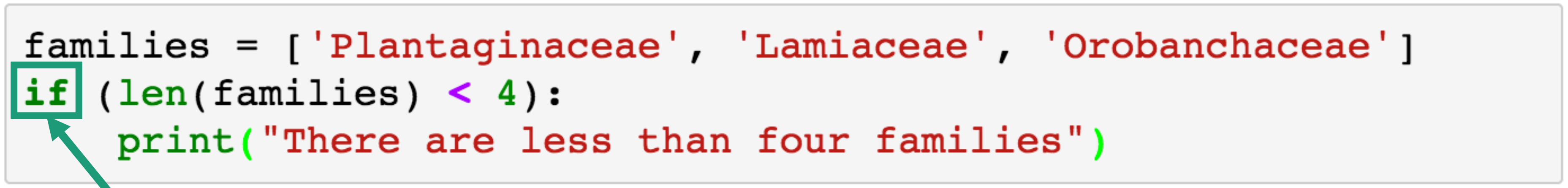
- if statement example



```python
families = ['Plantaginaceae', 'Lamiaceae', 'Orobanchaceae']
if (len(families) < 4):
    print("There are less than four families")
```

Flow Control: if statements

- if statement example

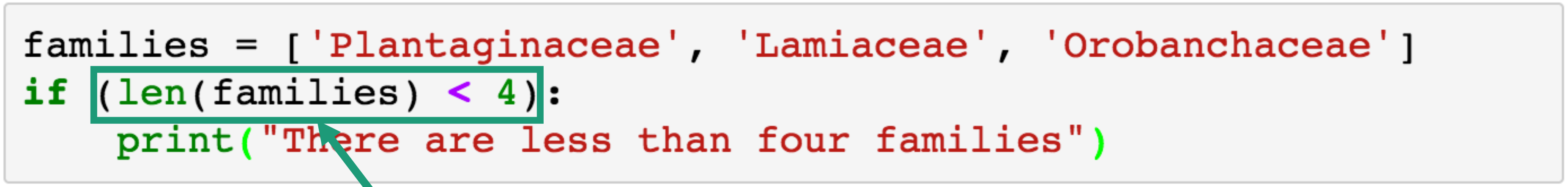


```
families = ['Plantaginaceae', 'Lamiaceae', 'Orobanchaceae']
if (len(families) < 4):
    print("There are less than four families")
```

All if statements must begin with the keyword `if`.

Flow Control: if statements

- if statement example

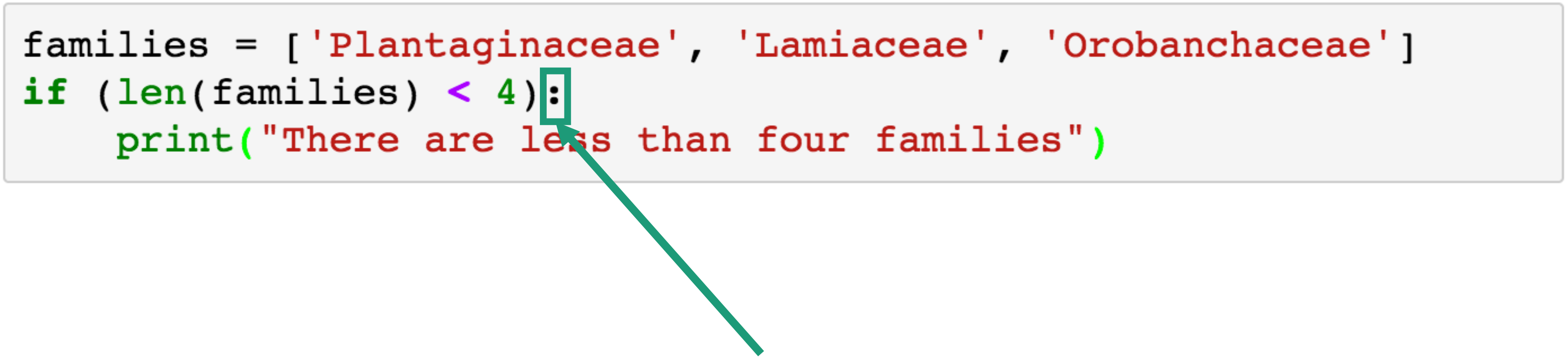


```
families = ['Plantaginaceae', 'Lamiaceae', 'Orobanchaceae']
if (len(families) < 4):
    print("There are less than four families")
```

The conditional statement is surrounded by parenthesis.

Flow Control: if statements

- if statement example

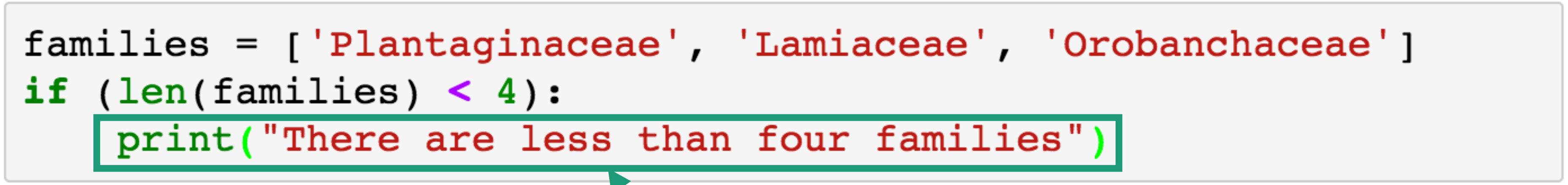


```python
families = ['Plantaginaceae', 'Lamiaceae', 'Orobanchaceae']
if (len(families) < 4):
    print("There are less than four families")
```

A colon marks the beginning of an indented code block.

Flow Control: if statements

- if statement example

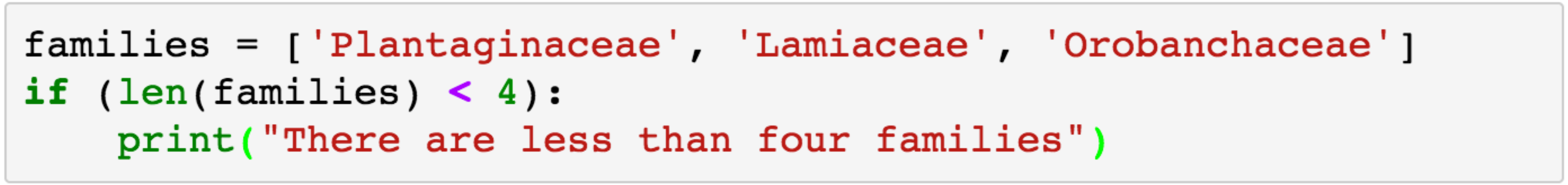


```
families = ['Plantaginaceae', 'Lamiaceae', 'Orobanchaceae']
if (len(families) < 4):
    print("There are less than four families")
```

Indented code block that is executed if the conditional statement is True
(four spaces are used to indent the block)

Flow Control: if statements

- if statement example



```python
families = ['Plantaginaceae', 'Lamiaceae', 'Orobanchaceae']
if (len(families) < 4):
    print("There are less than four families")
```

- Results

Flow Control: if statements

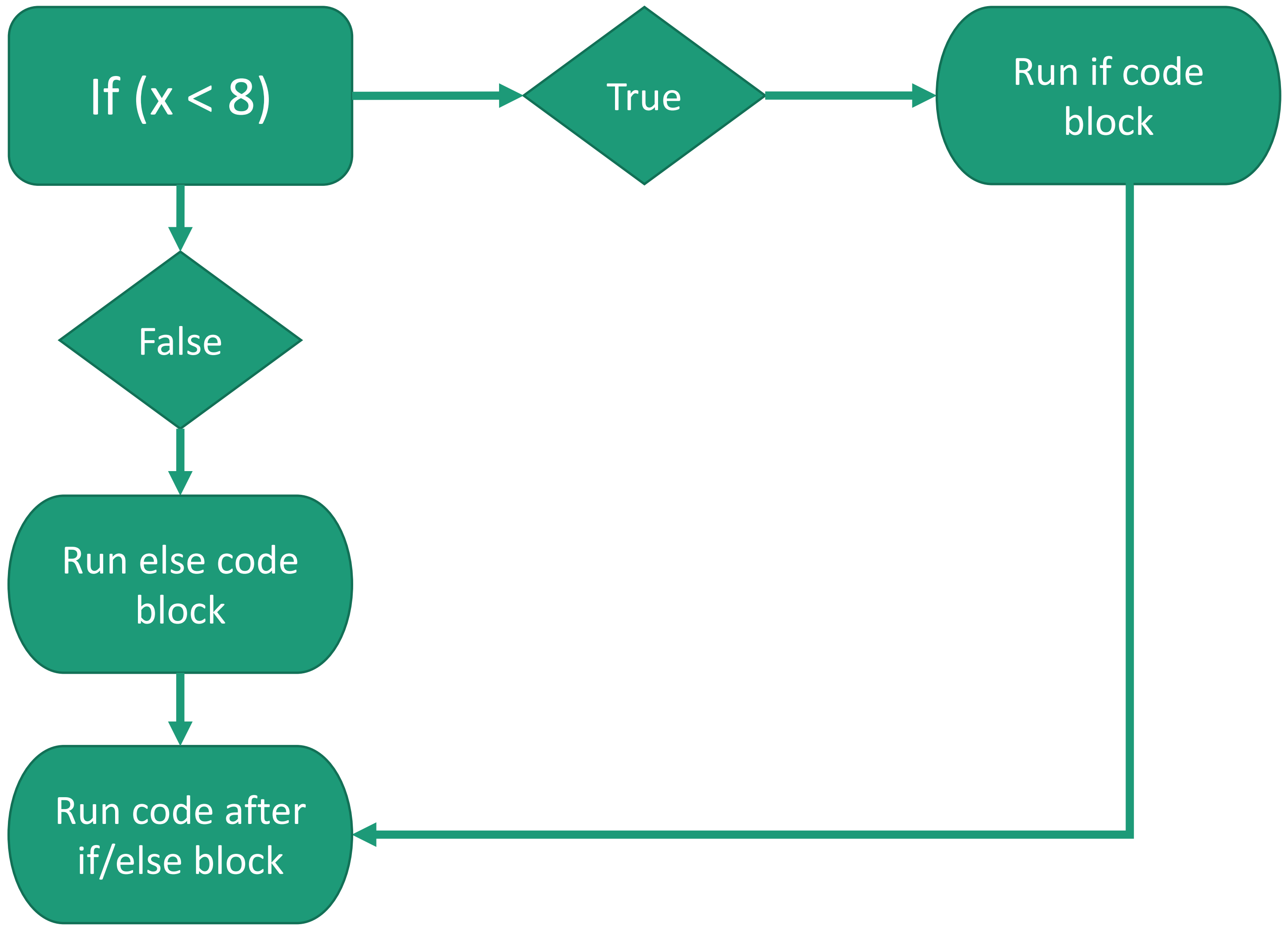
- if statement example

```
families = ['Plantaginaceae', 'Lamiaceae', 'Orobanchaceae']
if (len(families) < 4):
    print("There are less than four families")
```

- Results

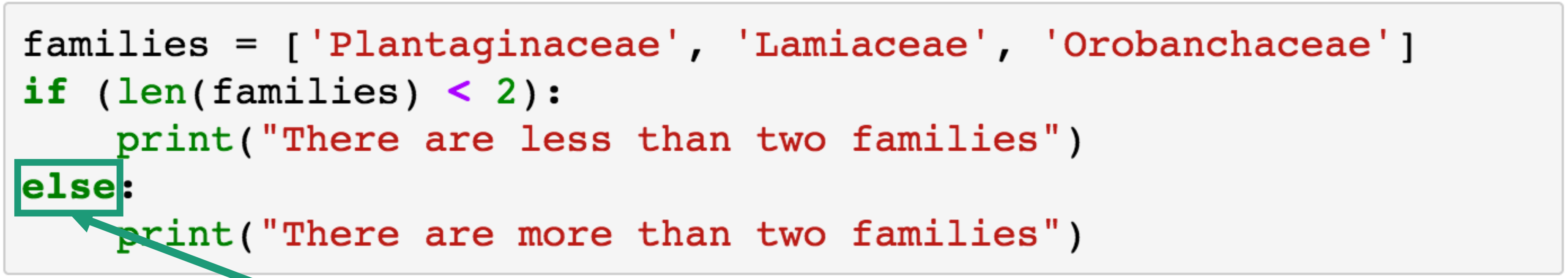
```
"There are less than four families"
```

Flow Control: else statements



Flow Control: else statements

- If/else statement example

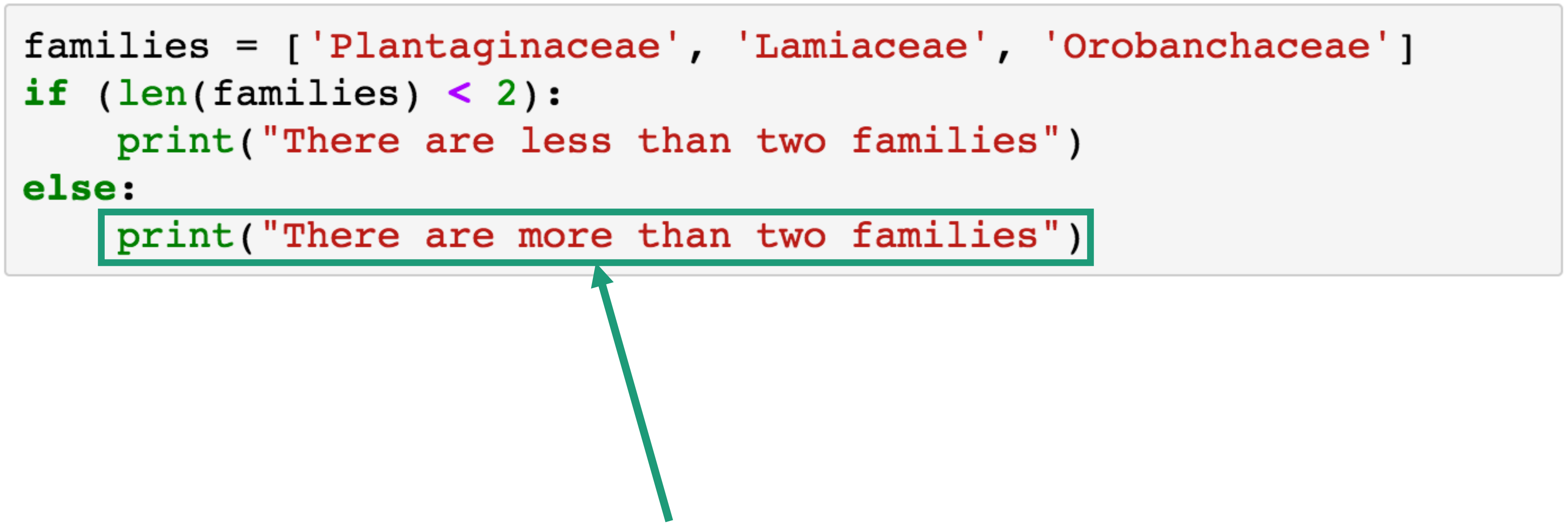


```python
families = ['Plantaginaceae', 'Lamiaceae', 'Orobanchaceae']
if (len(families) < 2):
    print("There are less than two families")
else:
    print("There are more than two families")
```

Else statement keyword. This indicates the beginning of an else statement. This is followed by a colon to mark the start of an indented code block.

Flow Control: else statements

- If/else statement example



```
families = ['Plantaginaceae', 'Lamiaceae', 'Orobanchaceae']
if (len(families) < 2):
    print("There are less than two families")
else:
    print("There are more than two families")
```

Indented code block that is executed when the conditional statement of the if block is False.
(four spaces are used to indent the block)

Flow Control: else statements

- If/else statement example

```python
families = ['Plantaginaceae', 'Lamiaceae', 'Orobanchaceae']
if (len(families) < 2):
    print("There are less than two families")
else:
    print("There are more than two families")
```

- Results

Flow Control: else statements

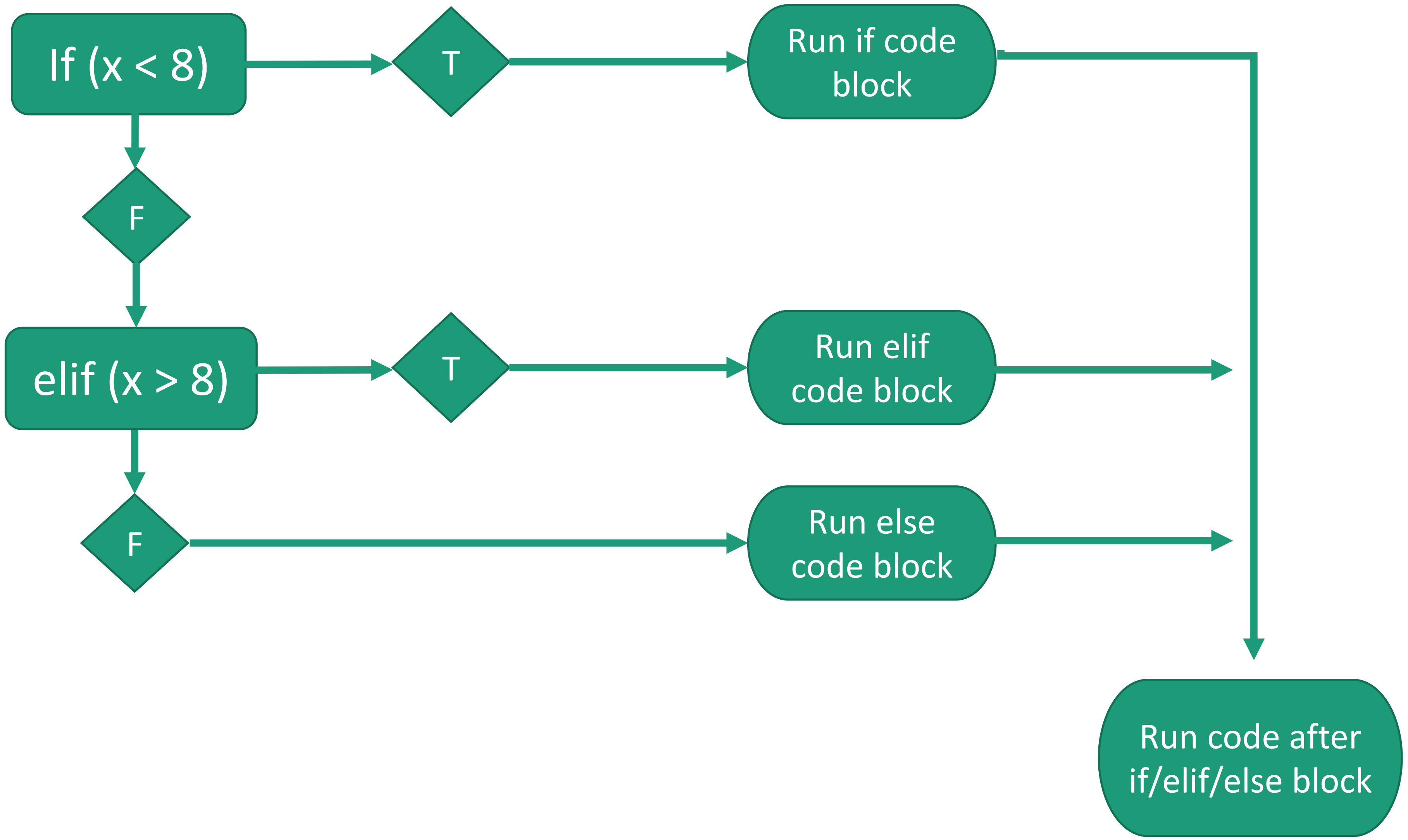
- If/else statement example

```
families = ['Plantaginaceae', 'Lamiaceae', 'Orobanchaceae']
if (len(families) < 2):
    print("There are less than two families")
else:
    print("There are more than two families")
```

- Results

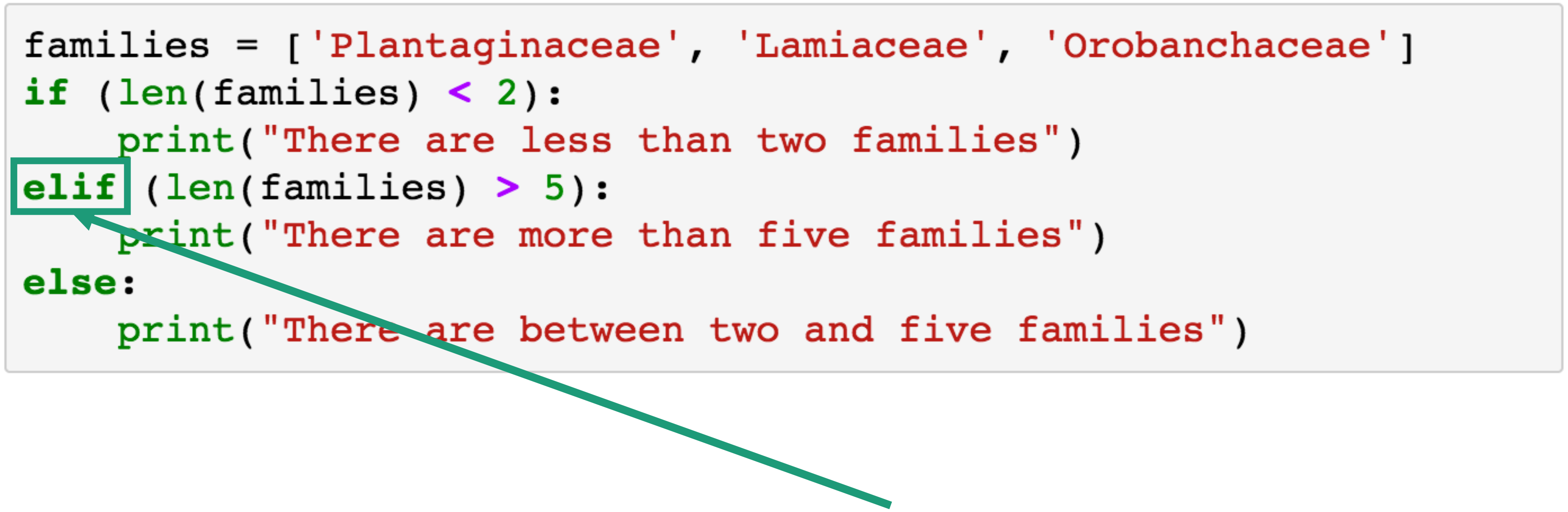
```
"There are more than two families"
```

Flow Control: elif statements



Flow Control: elif statements

- If/elif/else statement example

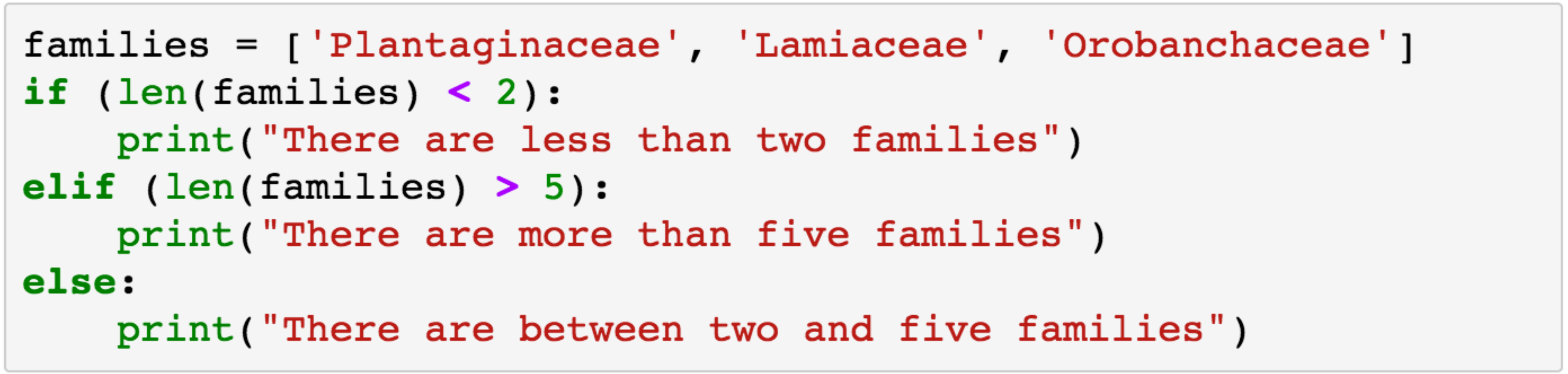


```
families = ['Plantaginaceae', 'Lamiaceae', 'Orobanchaceae']
if (len(families) < 2):
    print("There are less than two families")
elif (len(families) > 5):
    print("There are more than five families")
else:
    print("There are between two and five families")
```

Keyword indicating the start of an else if statement. This is followed by a conditional statement and a colon to begin the indented block. The indented block is executed only if the previous if and else if statements were False.

Flow Control: elif statements

- If/elif/else statement example



```
families = ['Plantaginaceae', 'Lamiaceae', 'Orobanchaceae']
if (len(families) < 2):
    print("There are less than two families")
elif (len(families) > 5):
    print("There are more than five families")
else:
    print("There are between two and five families")
```

- Results

Flow Control: elif statements

- If/elif/else statement example

```python
families = ['Plantaginaceae', 'Lamiaceae', 'Orobanchaceae']
if (len(families) < 2):
    print("There are less than two families")
elif (len(families) > 5):
    print("There are more than five families")
else:
    print("There are between two and five families")
```

- Results

```
"There are between two and five families"
```

Conditional Expressions

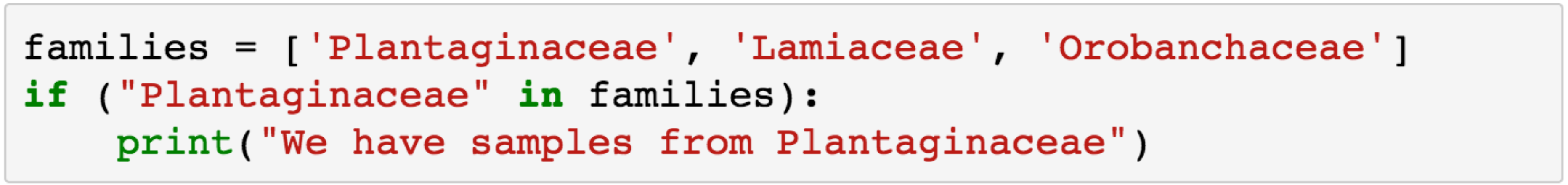
- >   greater than
- >= greater than or equal
- <   less than
- <= less than or equal
- == equal
- not
- or
- and

Booleans

- Booleans are a type of variable that can be set to either True or False
- Conditional expressions return a booleans

Conditional Expressions: membership testing

- The keyword ‘in’ is used for membership testing.
- This is used to test if a value is present in a Python collection (e.g. list)
- Example:

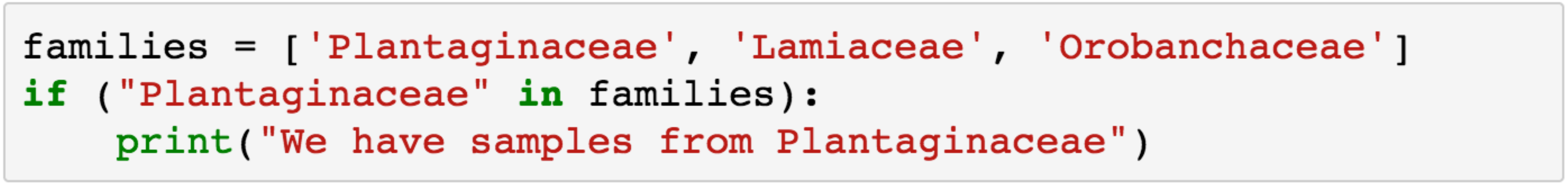


```
families = ['Plantaginaceae', 'Lamiaceae', 'Orobanchaceae']
if ("Plantaginaceae" in families):
    print("We have samples from Plantaginaceae")
```

- Result:

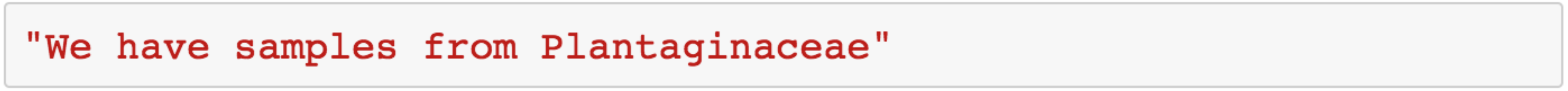
Conditional Expressions: membership testing

- The keyword ‘in’ is used for membership testing.
- This is used to test if a value is present in a Python collection (e.g. list)
- Example:



```
families = ['Plantaginaceae', 'Lamiaceae', 'Orobanchaceae']
if ("Plantaginaceae" in families):
    print("We have samples from Plantaginaceae")
```

- Result:



```
"We have samples from Plantaginaceae"
```

Data Structure: Dictionary

- Dictionaries consist of key:value pairs
- Keys can be any kind of variable but must be unique
- Values can by any kind of variable including another dictionary

Keys/Words	Values/Definitions
“mRNA”	56
“Genes”	[“Gene1”,”Gene2”,]
5	True
“Plantaginaceae”	[“Penstemon”]

Data Structure: Dictionary

- Creating a dictionary

```
example = {"mRNA":56, "Genes":["Gene1","Gene2"], 5:True, "Plantaginaceae":["Penstemon"]}
```

When creating a dictionary the key:value pairs are surrounded by curly braces

Data Structure: Dictionary

- Creating a dictionary

```
example = {"mRNA":56, "Genes":["Gene1","Gene2"], 5:True, "Plantaginaceae":["Penstemon"]}
```

Keys and their values are separated with a colon

Data Structure: Dictionary

- Creating a dictionary

```
example = {"mRNA":56, "Genes":["Gene1","Gene2"], 5:True, "Plantaginaceae":["Penstemon"]}
```

Key:value pairs are separated by commas

Data Structure: Dictionary

- Creating a dictionary

```
example = {"mRNA":56, "Genes":["Gene1","Gene2"], 5:True, "Plantaginaceae":["Penstemon"]}
```

- Accessing Values

```
example["mRNA"] #56
example["Plantaginaceae"] #["Penstemon"]
```

To access the value of a key use the dictionary's name followed by the key enclosed by square brackets .

Data Structure: Dictionary

- Creating a dictionary

```
example = {"mRNA":56, "Genes":["Gene1","Gene2"], 5:True, "Plantaginaceae":["Penstemon"]}
```

- Accessing Values

```
example["mRNA"] #56
example["Plantaginaceae"] #["Penstemon"]
```

- Updating and adding new key:value pairs

```
example["mRNA"] = 57 #57
example["Plantaginaceae"].append("Plantago") #["Penstemon", "Plantago"]
example["new key"] = "new value"
```