Python for Economics

Richard Lawrence

Afternoon Session, 9/10/2021

"Algorithms"

Table of Contents

(This course is divided into numbered Lessons)

6. Operations
7. Control Structures
8. Errors and Files

- Wrap up

Lesson 6
Operations

Using math and logic in expressions and statements

Learning Objectives

- Apply mathematical knowledge to Python programming
- Evaluate complex expressions
- Use integer arithmetic to solve problems

Bring math concepts to Python

- Operators
- Order of operations
- Common arithmetic operators
- Integer arithmetic
- Comparison operators
- Logic operators

Assignment “Arithmetic and Comparisons”

Go to google classroom assignment “Arithmetic and Comparisons”

Tasks

- Read and try the examples
- Exercise: evaluate arithmetic expressions
- Exercise: use integer arithmetic to solve problems
- Exercise: evaluate logical expressions

Homework

Practice with Operations.

- Assignment “Units of Time”
    - Fun Activity for Operations with Numpy Datetime
- Assignment “Lesson 6 Quiz”
    - Quiz on Operations

Break time reminder slide

break 10 minutes

Lesson 7
Control Structures

How to make a program do more than one thing

Learning Objectives

- Understand blocks and whitespace
- Use control structures
  - Functions
  - Conditionals
  - Loops
- Compose control structures for efficient code

Lesson Primer

To make a program versatile, it is necessary to write code that may get executed some number of times - *undetermined* at the time the code is written.

A short lecture will explain some key concepts.

Anatomy of a Control Structure

We have already seen the `for` statement. This is an example of a **control structure**.

```
for x in range():
    print()
```

Observations

- The `for` **control statement** ends with a colon ":"
- The next line is **indented** (some amount of space on the left)

Indentation

The amount of whitespace at the beginning a line is called the indentation.

whitespace `statement`

Common indentation levels: 2 spaces, 4 spaces, 8 spaces, etc

Warning: *Spaces* and *tabs* are both whitespace, but tabs don’t look the same in every text editor so it can be a "gotcha".

Blocks

In Python, programs are structured into **blocks**. A block is a group of statements that are executed together.

Statements in a block have the **same** indentation.

```
block 1
block 1
    block 2
    block 2
```

Nested Blocks

```
block 1
block 1
    block 2
    block 2
        block 3
    block 2
        block 4
    block 2
block 1
```

- Blocks can contain blocks with greater indentation

Example (left):

- All the statements with no indentation are part of the main block (block 1)
- block 1 contains all the other blocks

Nested Blocks

```
block 1
block 1
    block 2
    block 2
        block 3
    block 2
        block 4
    block 2
block 1
```

- Blocks can contain blocks with greater indentation

Example (left):

- **four** lines are part of block 2 because they're separated from each other by statements with *greater* indentation (blocks 3 and 4).

Nested Blocks

```
block 1
block 1
    block 2
    block 2
        block 3
    block 2
        block 4
    block 2
block 1
```

- Blocks can contain blocks with greater indentation

Example (left):

- block 3 and block 4 are *different* blocks because they're separated by a statement with *less* indentation (block 2).

Control Statements

A block can be executed once, multiple times, or not at all.

A **control statement** determines when, why, and how this occurs.

Control statements *precede* the block and end in a colon "`:`".

```
block 1
block 1
control statement:
    block 2
    block 2
    control statement:
        block 3
    control statement:
        block 4
    block 2
block 1
```

Assignment “Functions”

Go to classroom assignment Lesson 7 “Functions”

Tasks

- Read and try the examples
- Exercises: use functions for task repetition
- Exercises: use functions for good programming habits

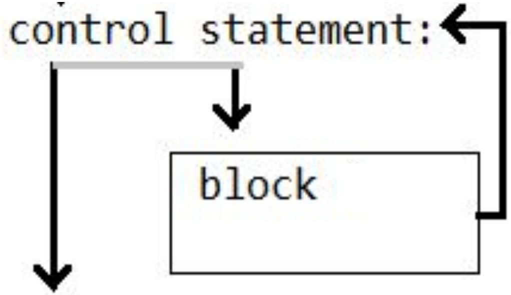
Flow Control

The *order* in which statements are executed is called Flow. Control statements determine where flow goes next.

Each control statement can either

- send flow *into* its block

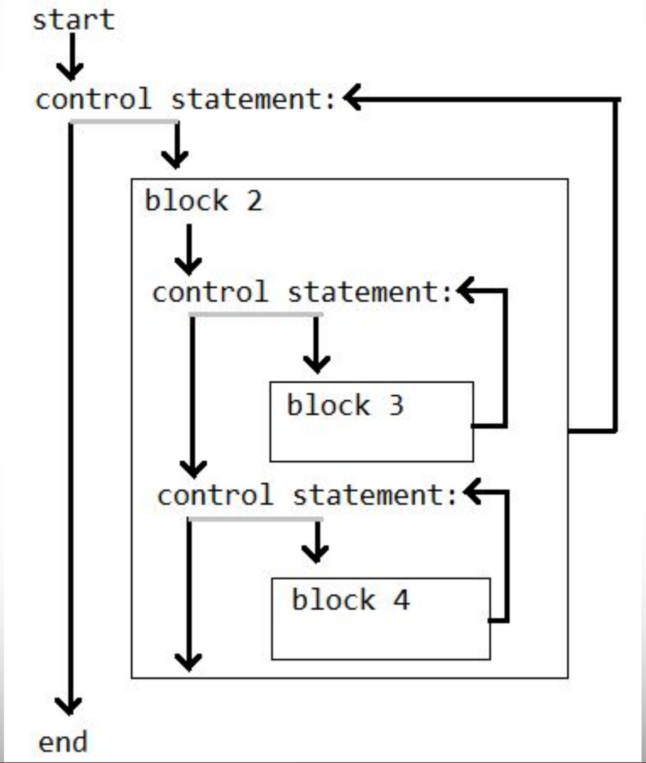
  or
- pass to the statement *after* its block.



When flow reaches the *end* of a block, it returns to the control statement above that block.

Nested Flow Control Diagram Example

```
block 1
block 1
control statement:
    block 2
    block 2
    control statement:
        block 3
    control statement:
        block 4
    block 2
block 1
```



Assignment “Conditionals”

Go to google classroom assignment “Conditionals”

Tasks

- Read and try the examples
- Exercises: conditionals for program flow
- Exercises: use conditionals to performs math tests

Break time reminder slide

break 10 minutes

Assignment “More conditionals”

Go to google classroom assignment “More conditionals”

Tasks

- Read and try the examples
- Exercises: more cases with Else and Elif
- Exercises: while loops

Homework

Extra practice with control structures assigned as homework.

- Assignment “Compute Pi”
  - Fun Activity using both Conditionals and Loops
- Assignment “Lesson 6 Quiz”

Lesson 8
Errors and Files

When you can’t trust the system...

Lesson Learning Objectives

- Establish good habits for file handles
- Safeguard untrusted actions
- Catch and handle exceptions

Homework

This lesson will not be covered in class; the assignments are homework.

- Assignment “Errors and Files”
    - Lecture and Exercises
- Assignment “Calculator”
    - Fun activity

Day 1 wrap-up

almost time to go home

Practice for next week

Most important skills to master

- Notebook interface
- Data types
- Conditions

Slides from today are available in Google Classroom

Summary of Homework Assignments

- Lesson 2: “Lesson 2 Quiz”
- Lesson 3: “Text files”
- Lesson 4: “The droid”
- Lesson 5: “User Input”, “Story Generator”, “Lesson 5 Quiz”
- Lesson 6: “Units of Time”, “Lesson 6 Quiz”
- Lesson 7: “Compute Pi”, “Lesson 6 Quiz”
- Lesson 8: “Errors and Files”, “Calculator”

Please complete your homework before class next Friday.

Office Hours Details

Please come to our office hours for assistance

- `M` 10-11 am Blocker 219B
- `T` 10-11 am (on Zoom)
- `W` 2-3 pm Blocker 219B
- `R` 2-3 pm Blocker 219B

Please join our slack channel for discussion

- Workspace `sweeterworkspace.slack.com`
- Channel `hprc-econ-fall-21` (private channel)

End of day Survey

This slide blank.