(http://csmatters.org/pd-new) P - 02

0bP - 0b10

Python: Values, Variables, and Loops

Unit Programming

Revision Date: Sep 07, 2019
Duration: 165 50-minute sessions

Lesson Summary

Summary:

In this blended lesson, teachers will learn about good coding practices, the basics of Python variables, boolean logic, and mathematical operations. They will also be introduced to loops and the Python range function. The online portion of the lesson will allow teachers to experience the interactive learning environment and review basic Python programming concepts. The in-person portion will use pair programming to explore the content and consider options for pairing students effectively.

Outcomes:

- Teachers will learn the basic syntax and operations of Python.
- Teachers will learn the basics of relational syntax.
- Teachers will gain an understanding of good coding practices.

Overview (this lesson may span multiple sessions and include both content and pedagogy of programming instruction):

Session 1:

- Basic arithmetic and boolean logic for Python (75 min)
    1. Teacher vs. Student Roles
    2. Coding Style
    3. Python Console
    4. Integer Division and Modulo
    5. Working with Arithmetic Operators
    6. Atomic (Basic) Data Types
    7. Relational vs. Logical Operators
    8. More Practice with Logic!
    9. Declaring and Creating Variables
    10. Exam Reference Sheet

Session 2:

- Loops (90 min)
    1. Introduction to Loops
    2. Range Practice
    3. More Loops Practice
    4. Homework

Learning Objectives

Key Concepts

Teachers should understand:

- Practice pair programming and make guidelines for pair programming.
- Learn how arithmetic operations work in Python.
- Learn atomic data types in Python.
- Learn the differences between conditional equality (==) and assignment (=) in Python.
- How to properly format code for readability.

Teacher Resources

Student computer usage for this lesson is: **required**

PROG02_Python Values, Variables, and Loops Folder (https://drive.google.com/open?id=0B5vAY-fhOT-iRWlqbTY2akJRTVU)

Lesson Plan

Session 1 (75 min)

TOTAL: 75 min

Materials: The instructor should follow the Values and Variables (https://drive.google.com/open?id=1AJqqu4EZP_nlvPWFOp7oU-s5LrrGIT4dgMMtScrlnlA) presentation for Session 1. The teachers should also have access to a computer and the Internet for accessing *Runestone Interactive*. For the links in the presentation, the presenter can either give the teachers access to the presentation so that they can click on the links on the slides or give the teachers those links.

Python Console: PyCharm

Pair teachers based on programming experience with Python.

1. Teacher vs. Student roles

This section uses slide 2 of the values and variables powerpoint.

Remind teachers of the many roles they will occupy during this workshop. In this particular session, some of them are learning this content for the first time, so they will naturally be in more of a student role. For teachers who are already familiar with the content, they should still try their best to go through the activities as their students would so they can start thinking about best practices for their own classroom.

2. Coding Style

This section uses slides 3-6 of the values and variables powerpoint.

1. Depending on the levels of experience teachers have with programming, the instructor can have teachers brainstorm coding style rules/conventions first. Work as a whole group to complete the list.
2. Next, briefly go over the suggested coding conventions:
    - Consistency with spaces vs. tabs for indenting
    - For Python, you have to indent with a tab. (Hence, Python does not require braces: the indenting determines the scope).
    - Old editors convert tabs into spaces; however, newer ones do not.
    - Keep lines relatively short (no longer than ~79 characters)
    - Name variables descriptively
        - use camelCase
        - Avoid using characters that look like other characters (1, i, l).
    - Be mindful of white space
        - Both horizontal and vertical
    - Comments are for the people writing it! Not so much for the teacher. (In a year, will you still be able to tell what this code does?)

Teachers should keep a copy of this list for their own coding practice later in the workshop.

3. Python Console

This section uses slide 7 of the values and variables powerpoint.

In addition to IDE's that one can download (e.g. PyCharm), there are online editing environments for programming in Python.

4. Integer Division and Modulo

This section uses slides 8-9 of the values and variables powerpoint.

Teachers read the *Runestone Interactive* page (http://interactivepython.org/runestone/static/thinkcspy/SimplePythonData/OperatorsandOperands.html) on the two operators and write definitions for the operators in their own words.

Notes:

- **Integer division //** yields the greatest integer of the quotient of any two floating point numbers (the quotient rounded down to the nearest integer). Integer division // is a Python operation (for other languages, there are different syntaxes).
- **Modulo %** produces the remainder portion of the quotient of any two floating point numbers (don't need to be integers).

The modulo operator can be very useful. For example, you can use it to tell if a number is even or odd by (num % 2). What is 5 % 2? 4 % 2?
The modulo operator % is fairly universal.

5. Working with Arithmetic Operations

This section uses slides 10-12 of the values and variables powerpoint

- Teach the teachers like they are students: show them a list of code, have them guess what happens, have them try each command on the console individually, and then demonstrate what happens on the projector.
- If the facilitators print out the commands on paper and give those to the teachers, it will be easier for the teachers to look at the commands in order to type them into the console.

The commands:

1. print(2+3*4)
2. print((2+3)*4)
3. print(2**10)
4. print(6/3)
5. print(7/3)
6. print(7//3)
7. print(7%3)
8. print(3/6)
9. print(3//6)
10. print(3%6)

6. Atomic (Basic) Data Types

This section uses slides 13-14 of the values and variables powerpoint.

Python has dynamic and weak typing (this type is inferred based on assignment. You can change the type of variable by reassigning the variable).

- **Integer:** 'number line numbers'
- **Floating point:** decimals
- **Boolean:** true or false (Python uses the keywords "True" and "False", but anything that isn't 0/False/None is True. Different languages may have different rules).
- **String:** The list and strings lesson will cover these in much more detail. For now, teachers should just know that a string is a set of characters.

7. Relational vs. Logical operators

This section uses slides 15-17 of the values and variables powerpoint.

Give a brief review of the different relational and logical operators in Python. Then have teachers read through the *Runestone Interactive* section and try to solve the boolean logic examples in the presentation.

Additional exercise: Number 7 has parentheses to avoid confusion. What happens when you take those parentheses out? How might this change the value of the statement?

8. More Practice with Logic!

This section uses slides 18-19 of the values and variables powerpoint.

These exercises are a bit trickier than the previous session's. Group teachers to solve them.

Note: The difference between "and" and "or" is very important! For an "and" to be true, both sides must evaluate to true. For an "or" to be true, at least one statement must be true. This is called an "inclusive or" which means it is still ok if both sides are true.

Here is a truth table (read as "if A is true and B is False then (A and B) evaluates to False")

A	B	A and B	A or B
True	True	True	True
True	False	False	True
False	True	False	True
False	False	False	False

9. Declaring and Creating Variables

This section uses slides 20-24 of the values and variables powerpoint.

Variables are essentially named boxes that store values. In Python, variables are not inherently typed (i.e. the box for "dog" could hold a string "Boxer" and then be changed to hold the number 5 without problems)

Using variables is one of the fundamental tasks of programming (for most languages). Make sure teachers understand how variables work before continuing!

10. Exam Reference Sheet

This section uses slide 23 of the values and variables powerpoint

Give teachers a few minutes to look at the sheet. They will have more opportunities to look at the AP exam in future sessions, so this should merely be a brief look at the coding style. The AP exam is designed to be language-independent, as CSP is being taught with many programming languages.

Session 2

TOTAL: 90 min

Materials: The instructor should follow the Loops (https://drive.google.com/open?id=1yrpfEeqGUqdS1N30zhIlPqvbl8vHio8_QEC5HLXcjDM) presentation. The teachers should also have access to Runestone.

Python Console: PyCharm

Pair teachers based on programming experience with Python.

1. Introduction to loops

This section uses slides 2-7 of the Loops Powerpoint

This section goes over a basic definition of *for* and *while* loops. It may help for the instructor to do some demonstrations here, so teachers can see the loops in action.

Note: in Python, *for* loops are used for two different purposes. The "for each item in list" loop will be covered with the lists and strings lesson. For now, we are learning for loops in the context of repeating something a specified number of times.

Remember, with *while* loops: if the condition in the *while* loop never becomes false, the *while* loop will never end!

2. Range Practice

This section uses slides 8-11 of the Loops Powerpoint

Teachers should pair up and trace through each question.

3. More Loops Practice

This section uses slides 12-13 of the Loops Powerpoint

1. Explain and model pair programming for the teachers.
    - The rules of pair programming
        - There are two roles: the **navigator** and the **pilot**.
        - The navigator gives instructions to the pilot on what to type. The navigator may not use the computer.
        - The pilot follows the navigator's directions and types whatever the navigator says to. The pilot may not take over and type whatever they want.
        - If the pilot and navigator disagree on what to type, they should discuss it until they come to an agreement. They are to work as a team.
2. When teaching students how to do pair programming, selecting a volunteer student to act out the roles of navigator and pilot with the teacher for can be especially helpful. Instruct teachers to use pair programming with their partners as they work through the next set of practice exercises.
3. After enough time has passed that teachers have read through the introduction of the Python for Beginners lesson (https://opentechschool.github.io/python-beginners/en/loops.html#introduction), the instructor should demonstrate **live coding** with the teachers participating as students in order to produce a solution for the *Drawing a Dashed Line* exercise in *Python for Beginners.*
    - For **live coding**, the teacher projects their IDE (PyCharm, etc.) to the class and goes through the process of writing a piece of code with the class. The teacher should engage the students, ask them questions, and ask them for suggestions so that the coding is a class-wide effort. Meanwhile, the students also type the code. Students should not try to work ahead while live coding is happening, but teachers are encouraged to differentiate if they think the given exercise is too easy for a given student or if they notice that a student is having trouble keeping up.
    - Through live coding, students learn
        - how to set up a Python file for making their own programs
        - the teacher's preference for the header on each file (typically name, date, class, and a brief summary of the program contained in the file)
        - thought processes for typing out code (after planning has happened on paper)
        - whatever coding practices the teacher models (writing comments, conventions for naming variables, use of white space, etc.)
    - Most importantly teachers should model the process of making a typo or some other common mistake (forgetting the colon at the end of a for/while statement, etc.),

running the code with that error, reading the error message, and going through the code to fix the mistake. In this way, students learn from live coding that it is okay to make mistakes.

4. After the live coding demonstration, teachers should continue to work on the remaining practice exercises using pair programming model. Recommend to teachers that they save their files for future reference.

Homework

Teachers were instructed to complete the first ten chapters of *How to Think Like a Computer Scientist* before the workshop, but if they need a refresher on variables, they can go through Chapter 2 of the book. If they would like to review the content about loops (iteration), they can go through Chapter 4 and Chapter 8.

Options for Differentiated Instruction

Group can be split into a "beginner" and an "advanced" group based on the existing skill levels established in the earlier lesson. The advanced group could move at a faster pace and begin examining control structures such as for loops and while loops, using the resources linked below.

Evidence of Learning

Formative Assessment

Teachers act as students for part of the introductory sessions, but they will of course be absorbing information as teachers in order to properly learn the materials needed to teach.

Summative Assessment

Teachers will create small programs to demonstrate knowledge of the concepts covered in the lesson.

(http://www.umbc.edu/) (http://www.umd.edu/)

(http://www.nsf.gov/)

Authored by: CS Matters in Maryland
Website: csmatters.org (http://csmatters.org)

Email: csmattersinmaryland@gmail.com (mailto:csmattersinmaryland@gmail.com)