(http://csmatters.org/pd-new) P - 05

0bP - 0b101

Python: Strings and Lists

Unit Programming

Revision Date: Jan 09, 2018
Duration: 75 50-minute sessions

Lesson Summary

Summary: Teachers will learn the basics of strings and how they are implemented in Python. They will be introduced to lists.

Outcomes:

- Teachers will understand the concepts of strings in Python.
- Teachers will have knowledge of some basic built-in Python functions for strings.
- Teachers will have experiance using repl.it and CodingBat as practice resources.
- Teachers will gain a basic introduction to lists.

Overview:

- Introduction to Strings (30 min)
- CodingBat practice (30 min)
- Brief Introduction to Lists (15 min)

Learning Objectives

CSP Objective

Big Idea - Professional development provides opportunities for teachers to examine, observe, practice, and receive feedback on their use of research-based instructional strategies to help all of their students master Maryland content standards.

-
    - LO 1b - Professional development provides opportunities for teachers to examine, observe, practice, and receive feedback on their use of research-based instructional strategies to help all of their students master Maryland content standards.

Key Concepts

Teachers will understand:

- Strings
- Indexing
- Slicing
- Other built in functions for strings

Teacher Resources

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

PROG05_Python Strings and Lists Folder (https://drive.google.com/open?id=0B5vAY-fhOT-ibjEwTHFXaWhwenM)

Lesson Plan

TOTAL: 75 min

Materials: The instructor should follow the "Strings and Lists" presentation. The teachers should have access to this presentation and the Internet.

Python Console: Teachers will be using repl.it and codingbat.com for this lesson, but any of the activities in the lesson can be modified so that PyCharm or any other Python console can be used instead.

Note: Pair teachers based on programming experience with Python.

Introduction to Strings (30 min)

1. **Slide 2:** Give teachers a couple of minutes to write down their own definition of what a string is in their journals.
    - As they finish, have them set up a repl.it account. Repl.it is a free online resource that allows one to write, run, and save code in many coding languages.
    - For this session, teachers should select Python 3. (Later on, if teachers are using repl.it and are making use of the turtle library, they should select Python (with Turtle).)
    - Teachers can also share code they have written and saved in repl.it with students by providing a link.
2. **Slide 3:** Take this time to briefly introduce *Python for Everybody (https://www.py4e.com/),* a book for teaching Python 3 that is available from the website as a PDF or online. The CS Matters Curriculum uses *Python for Everybody* for some of the lessons in Unit 2.
3. **Slide 4:** A **string** is a sequence of characters. Strings are indicated by single or double quotes. One can reference each character in a given string by that character's **index**. The indeces of a string begin numbering at 0, not at 1.
4. **Slide 5:** Introduce the **length function** for strings. The length function has the following syntax: len(string), which returns the number of characters in the given string. The command len('computer') would return the integer 8.

5. **Slide 6:** Have teachers respond to the following prompt in their journals: *Explain why the length of a string is one digit higher than the highest index value of the same string.*
    - The answer is that indexing begins at 0, not 1, so the value of the last index is one less than the number of characters in the string. This concept is important for learners to internalize because it allows one to code successful loops and to make less mistakes when working with strings.
6. **Slides 7-9: Traversing a string** is when one accesses the characters in a string one at a time. One can do so with a while loop or with a for loop. Links to examples of each are provided on the slides.
    - **Note:** Teachers who are logged into their repl.it account can "fork" the sample code, which allows them to edit the code and save the change to their account for future access.
    - Encourage teachers to name the code something meaningful that will allow them to recognize what it is later on.
    - For both examples, practice some live coding with the teachers by asking them how to modify the code to do something slightly different or additional. Some example tasks are altering the code so that it
        - Gets the name of the fruit from the user (with the input function).
        - Prints the letters of the fruit all on the same line.
        - Prints the letters of the fruit in reverse order.
    - The more experienced the teachers are at programming, the more of these tasks you might have time to work through.
7. **Slides 10-11: Slicing a string** allows one to retrieve a part of a string. The default syntax is stringName[start:stop:step], for which *start* gives the index at which the slice will begin, *stop* gives the index before which the slice will end, and *step* gives the jump size by which the string will be traversed. The default values for the arguments are as follows: start = 0, end = len(stringName), step = 1; any of these parameters can be left out of a command.
    - **Example:** if string = 'golden retriever', string[2,11,2] would return the string 'le er' because the slice begins at the character whose index is 2, jumps 2 characters forward appending the characters whose indeces are 4,6,8, and 10, stopping before the character whose index is 11.
8. **Slides 12-13:** Several slicing examples are given for teachers to try to predict the output before actually executing the commands in the repl.it terminal.
    - **Activity:** It would be worthwhile to pass out these commands to the teachers on slips of paper instead of making them available on the presentation that is shared with them. This will require the teachers to type the commands instead of copying and pasting the text. Sometimes errors arrise from executing commands that have been copied and pasted from a word-processing program due to imbedded information that comes with that copied text. Also of importance is that teachers (like their students) will learn more from tying the commands themselves.
    - Emphasize to teachers the gains made in learning when students make predictions about what each command will return before executing the command. This makes the students more active in the learning, as opposed to passively observing the results of each command.
    - Teachers should check their answers on Slide 13 once they are finished.
9. **Slides 14-16:** One can search for a string within a string by using stringName.find(subString). This returns the beginning index of the first occurance of subString within string. For example, if stringName = 'grapefruit', stringName.find('fr') returns the integer 5.

- **Activity:** Slide 15 contains some practice with searching strings. As with the previous exercise, ask teachers to predict what each command will return before executing the command. When teachers are finished, they should check their answers on Slide 16.

10. **Slide 17: Parsing a string** is when one splits a string into substrings. This is useful when given a body of text for which one is interested in the words contained in the text but not the spaces inbetween the words. Parsing is also useful in breaking apart an email address to find a username or a domain name.
    - **Activity:** Live code with the teachers to come up with a program that prompts the user to enter their first and last name, separated by a single space, and returns the first and last name as separate strings. See here (https://repl.it/KIu4/2) for a sample solution.
11. **Slide 18:** Python has several built-in functions to be used with strings. Give teachers a few minutes to look through the documentation on string functions. Ask them to pick out a function and describe a task for which that function would be useful. (For example, the string.find() function is useful for parsing strings.)
    - Coding languages often provide many extra functions you might rarely use, and being able to look up available functions using the documentation is often an important skill in programming.

CodingBat practice (30 min)

1. **Slide 19:** Introduce CodingBat (http://codingbat.com/python) to the teachers as another resource for their programming classrooms. CodingBat provides many programming exercises that students can complete. If students are logged in and have indicated a teacher to be associated with their account, then that teacher can see their progress on each exercise the students attempt. A few things to note about CodingBat:
    - All of the exercises require students to create a function, so students will need to learn about functions before using CodingBat.
    - CodingBat determines whether or not a student-provided solution is correct by checking it with sets of arguments and anticipated return values. This is a great opportunity to discuss with students how one can test a program to be sure it is functioning as intended.
    - Students can request a hint and then copy and paste it as their solution, so teachers should look out for this and discourage it however they can. There is a tutorial on the CodingBat site on how teachers can make their own coding exercises, so that might be a way around this.
2. **Slide 20:** Reiterate the importance of planning a program before typing it up. Requiring students to write out their plan for a program on paper forces them to have a solid algorithm before going to the computer to type the code. Remind teachers that an algorithm can be written out in natural language (sentences, bulleted statements, etc.) or pseudocode.
3. **Slide 21:**
    - **Activity:** Have teachers try some of the CodingBat exercises given on the slide. Emphasize the expectation that they are to plan out their algorithms on paper before typing out the solution code. Also remind teachers that going through this process will help them to understand what their students are going through as they learn to code and how best to support them in that endeavor.

Brief Intro to Lists (15 min)

1. **Slides 22-24:** A list is a sequence of values indicated by square brackets [ ]. The items or elements in a list can be of any type (integer, floating point, Boolean, string, a list can even contain a list as an element!). Elements in a list are indexed starting with 0 instead of 1, just like characters in a string. It is helpful to point out that one can think of a string as a list of characters.
2. **Slides 25-26:** The biggest difference between strings and lists is that **lists are mutable** whereas strings are not. One is able to overwrite the elements of an established list, but one cannot edit the contents of a string on the character level.
3. **Slide 27:** Give teachers are few minutes to work through each of the commands. As always, teachers should make a prediction about the effect of a given command before executing it.
4. **Slide 28:** If time permits, teachers should take a few minutes to write down their own definitions of the given set of words. They can refer to the glossary at the end of the Lists chapter of the Runestone Interactive book *How to Think Like a Computer Scientist* for assistance if they are struggling with a given term. Teachers should then compare their definitions with their partner's.
    - If time does not permit, teachers should complete this activity outside of the session (homework).

Options for Differentiated Instruction

It may be advantageous to set more time aside for practice with strings and lists after the session for further practice.

Evidence of Learning

Formative Assessment

Teachers will gain an understanding of their mastery through completing the various labeled tasks in the PowerPoint.

Summative Assessment

Teachers can gain an understanding of their mastery through the completion of excercises on CodingBat.

(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)