Data Types

Python has a wide variety of built-in types for storing anything from numbers and text (e.g., `int`, `float`, `str`) to common data structures (e.g., `list`, `tuple`).

Manager: Recorder:

Presenter: Reflector:

Content Learning Objectives

After completing this activity, students should be able to:

- Explain differences between integer and floating-point.
- Reference a specific element of a sequence by an index.
- Compare and contrast sequence types (str, list, tuple).

Process Skill Goals

During the activity, students should make progress toward:

- Providing feedback on how well other team members are working. (Teamwork)


Copyright © 2023 T. Shepherd, C. Mayfield, and H. Hu. This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License.

Model 1 Integers and Floats

Every value in Python has a ***type*** which determines what can be done with the value. Consider the following statements and expressions that were entered into a Python Shell.

Python code	Shell output
`integer = 3`	
`type(integer)`	<class 'int'>
`type("integer")`	<class 'str'>
`pi = 3.1415`	
`type(pi)`	<class 'float'>
`word = str(pi)`	
`word`	'3.1415'
`number = float(word)`	
`print(word * 2)`	3.14153.1415
`print(number * 2)`	6.283
`print(word + 2)`	TypeError
`print(number + 2)`	5.14159
`euler = 2.7182`	
`int(euler)`	2
`round(euler)`	3

Questions (15 min) Start time:

1. What is the value and type (`int`, `float`, or `str`) of the following variables?

Variable	Value of Variable	Type of Value
`integer`		
`word`		
`number`		
`euler`		

2. List the function calls that convert a value to another type.

3. How does the behavior of the operators (+ and *) depend on the data type?

4. What is the difference between the `int` function and the `round` function?

5. What is the value of `3 + 3 + 3`? What is the value of `.3 + .3 + .3`? Enter these expressions into a Python Shell—what do you notice about the results?

6. Based on the previous question:

a) In order to store a number with 100% accuracy, what data type is required?

b) How might you precisely represent a bank account balance of $123.45?

7. Try calculating a very large integer in a Python Shell, for example, 123^{456}. Is there a limit to the integers that Python can handle?

8. Try calculating a very large floating-point number in a Python Shell, for example, 123.0^{465}. Is there a limit to the floating-point numbers that Python can handle?

9. Summarize the difference between the numeric data types (`int` and `float`). What are their pros and cons?

Model 2 Lists

A variable can hold multiple values in the form of a ***list***. The values are separated by commas and wrapped in square brackets. For example:

```
primes = [2, 3, 5, 7, 11, 13, 17, 19, 23, 29]
```

Each ***element*** of the list can be referenced by an ***index,*** which is the value's sequential position starting at 0. For example, `primes[4]` is 11.

index	0	1	2	3	4	5	6	7	8	9
value	2	3	5	7	11	13	17	19	23	29

Questions (15 min) Start time:

10. What is the index of the second element of `primes`? What is the value at that index?

11. How does the index number compare to the position of the element?

12. Type each line of code in a Python Shell and write the corresponding output. If an error occurs, write the type of error.

Python code	Shell output
`odd = [1, 3, 5, 7]`	
`odd`	
`odd[2]`	
`odd[4]`	
`len(odd)`	
`number = odd[1]`	
`number`	
`odd[1] = 2`	
`odd`	
`number`	

13. How did you reference the value of the 3rd element of `odd`?

14. What did the output of the len() function tell you about the list?

15. One of the lines in #12 displayed an error. Explain the reason for the error.

16. Write a statement that assigns a list of three integers to the variable `run`.

17. Write a statement that assigns the value 100 to the last element of `run`.

18. Write a statement that assigns the first value of `run` to a variable named `first`.

Presenter: Write your team's answers for the last three questions, in large print, on a blank sheet of paper. When asked, hold up your answers to the entire class. Be ready to explain your team's answers.

Model 3 Sequences

Lists and strings are examples of ***sequence*** types. Consider the following lines that were entered into a Python Shell. Write an asterisk (*) next to any row your team has questions about.

Python code	Shell output
`seq1 = "one two"`	
`type(seq1)`	<class 'str'>
`len(seq1)`	7
`seq1[1]`	'n'
`seq1[1] = '1'`	TypeError: 'str' object does not support item assignment
`seq2 = ["one", "two"]`	
`type(seq2)`	<class 'list'>
`seq2[1]`	'two'
`seq2[1] = 1`	
`print(seq2)`	['one', 1]
`seq3 = ("one", "two")`	
`type(seq3)`	<class 'tuple'>
`len(seq3)`	2
`seq3[1]`	'two'
`seq3[1] = '1'`	TypeError: 'tuple' object does not support item assignment
`seq4 = "one", 1`	
`type(seq4)`	<class 'tuple'>
`seq4`	('one', 1)

Questions (15 min) Start time:

19. What are the names of the three sequence types introduced in Model 3?

20. How does the syntax of creating a tuple differ from creating a list?

21. Is there more than one way (syntax) to create a tuple? Justify your answer.

22. Which sequence types allow their elements to be changed? Which do not?

23. Is it possible to store values of different types in a sequence? If yes, give an example from the table; if no, explain why not.

24. Summarize the difference between lists and tuples. How do they look differently, and how do they work differently?

25. (Optional) Enter the following lines in a Python shell and write the output. What do you learn about converting strings and lists?

```
letters = list("Hello")
letters
str(letters)
```