

Description:

Your role as an assistant computer programmer means that you must learn to be proficient in many types of programming and coding. For this assignment, you will have to learn how to use a high-level programming tool known as Python. Because you are in training, you will be working on several modules and continue to build upon your skills. You will work on various skills in Python and submit them to the head computer programmer to make sure your codes are both sufficient and efficient. For this task, you will learn and utilize data types, variables, and expressions.

Students will be able to:

- Define key terminology associated with programming and Python
- Understand Python syntax, keywords, and special symbols.
- Understand variable assignments and usage
- Understand how variable assignments affect expressions
- Distinguish between the variable types

Students will understand:

Python is a widely known, high-level coding language. Prior to this lesson, students were exposed to understanding how computer programming languages are logical, methodical and meticulous. They were also exposed to the concept of debugging when given erroneous or repetitive code. In this lesson, students will learn how to define and utilize variables by understanding the concepts of data types and expressions.

Key Definitions & Concepts: [1]

- **Value:** one of the basic units of data (i.e. number, string) that a program manipulates.
- **Type:** a category of values
- **Integer:** a type that represents whole numbers (type int)
- **Floating-point:** a type that represents numbers with fractional parts (type float)
- **String:** a type that represents a sequence of characters (type str)
- **Variable:** a name that refers to a value
- **Statement:** a section of code that represents a command or action
- **Assignment:** a statement that assigns a value to a variable
- **Keyword:** a reserved word that is used by the compiler to parse a program (ex: if, def...)
- **Operator:** a special symbol that represents a simple computation like addition, multiplication, or string concatenation
- **Expression:** a combination of variables, operators, and values that represents a single result value
- **Operand:** one of the values on which an operator operates
- **Concatenate:** to join two operands end-to-end
- **Comment:** information in a program that is meant for other programmers to read and has no effect on the execution of the program

Standards: [Copied from: 2]

2.AP.11: Create clearly named variables that represent different data types and perform operation on their values.

2.AP.15: Seek and incorporate feedback from team members and users to refine a solution that meets user needs.

Background Information

Prior Knowledge:

- Formal versus natural language
- Logical, step-by-step thinking
- Introductory debugging techniques

Science Practices: [Copied from: 3]

- Planning and carrying out investigations
- Constructing explanations and designing solutions

Core Ideas: [Copied from: 4]

- Developing possible solutions
- Optimizing the design solution

Cross Cutting Concepts: [Copied from: 5]

- Systems and system models
- Structure and function

Possible Preconceptions/Misconceptions:

Students should be able to complete this lesson successfully, save for the “Equations” section in the second exploration worksheet. Since the next lesson covers functions and modules in Python, expect the students to have partially correct answers for converting the algebraic equations into Python equations.

Lesson Plan - 5E(+) Model

Engage: [6]

Engage the students with an anchoring event: a video clip from Khan Academy. This clip talks about how computer programming is used everywhere in the world around them. The purpose of this clip is to get students thinking about the importance of computer programming and to engage them in a class discussion of “what is computer programming?” and “how is programming useful to us?.” The following is a link to the website: [What is programming?](#) The students should complete the *Computer Programming* half sheet while watching the video. Then, they will be engaged in a whole class discussion facilitated by the instructor about how that video clip connects with the lesson plan’s goal and objectives. Teachers will use students’ responses to address any misconceptions as necessary. This section should take up to 10 minutes to complete.

Explore:

Part I: Introduction [7]

Students will work in pairs to complete the “your definition” half of the *Concept Check* worksheet to have a base understanding of the following: variables, variable types, and Python keywords. The

students are provided with typical variable naming conventions and helpful tips when choosing a variable name. This information should help the students make enough connections to be able to create definitions in a group of 2 to 3 students. After 5 minutes working in their pairs, the teacher should lead a class discussion to correct or improve the students' responses. The instructor should prompt the students to write these improved or updated definitions into the "class definition" half of the worksheet. This section should take less than 10 minutes to complete.

Part II: *Benchmark Lesson: Declaring Variables and Differentiating Variable Types*

Without preface, the teacher will allot 10 minutes for the students to complete the *Python: Valid and Invalid Variable Names* section of the following worksheet. The students will have the prior worksheets and the *Tables of Python Information* handout to use as resources. These worksheets provide the students with both the explicit rules and the common methods of declaring variables in Python. Also, the handout provides the students with tables that include Python keywords, operators, and escape sequences. The goal of this section of the lesson is to have students work individually with a set of resources to learn proper variable declaration in Python. Completion of the variable declaration table leads into a 5 minute class discussion to correct the students' tables and to ensure they understand why certain conventions are better than others. This enables students to begin understanding the different variable types. This portion of the lesson should take less than 15 minutes.

The instructor will lead the class in an open discussion to complete the *Variable Types* table. By asking students what values (i.e. data) can be represented by the given variables, they will be able to determine the corresponding variable type(s). The purpose of this worksheet is to provide students with an understanding of how variables in Python have a particular type. Depending the variable type, programmers are limited in terms of functionality. For example, an integer needs to be converted into a string before a Python program can utilize the integer value within a print statement. This section of the lesson should take about 5 minutes to complete.

Part III: *Investigation Lesson: Evaluating Expressions and Variable Assignments*

The instructor will give each pair of students access to a computer/tablet that runs IDLE and has the latest update of Python installed. The students will be engaged in completing the *Python: Expressions and Assignments* worksheet which is designed to have students grapple with given codes and understand the associated outputs. Instruct students to type each code and each expression into the interactive mode of IDLE and allot 10 minutes for the students to complete this section of the worksheet with a partner. Upon completion, the teacher will engage the class in an open discussion about why the computer outputs make sense. (It is especially necessary to discuss the double versus single quote usage in Python). The goal here is to ensure that students understand how a variable is declared, assigned and utilized. This section should take less than 15 minutes to complete.

Explain:

Throughout the exploration, the students will engage in discussions that inquire their understanding and knowledge of the information at-hand. Teachers will be informally asking students to explain all topics and relevant connections throughout the entirety of this lesson. While the worksheets do not include many explicit "why" or "how" questions, the open class discussions and corrections serve that purpose. Further, there are multiple sections throughout the exploration that prompt the students to explain and to detail their reasoning.

Elaborate:

The elaboration of this lesson is the “Equations” section on the final class worksheet. Since students would be primed to understanding variable declaration and assignment, then they can be challenged to transfer their new programming knowledge into prior known mathematical concepts. By asking the students to convert algebraic equations into Python programming language, they are forced to bridge the lesson goals with the logic goals of the prior Python lesson. This transfer of understanding is utilized in all aspects of life, especially within careers. Hence, this section of the lesson provides a crucial and applicable technique utilized in the real world.

Evaluate:

This lesson is designed with having both informal and formal evaluations throughout its entirety. The informal evaluations occur throughout the exploration because of the leading and open-ended questions and class discussions. This allows teachers to gauge surface-level student understanding. By surveying the students during completion of the worksheets, teachers will be able to hear and address any misconceptions or misunderstandings as necessary. The formal evaluation of this lesson is the exit ticket. This 5 minute, individual worksheet prompts the students to observe a Python code and its output. The analysis questions align with the lesson goals of understanding variable assignments and their functionality within code.

Enrich:

This lesson could be extended by introducing how to define and utilize functions in Python. Since this lesson focuses on variable declaration and assignment, then the students are successfully primed to interacting with Python code that includes functions. Also, students can be introduced to conditional logic and loops. Priming students to understanding functions is beneficial and logical because it provides availability for students to learn about conditional statements and loops by interacting with and exploring Python code that includes functions with the conditionals and the loops.

****All associated documents are attached below****

****Reference *Annotated Bibliography* on the very last page of this packet****

Name: _____ Date: _____

Computer Programming [6]

- 1.) Define programming.

- 2.) List at least three examples of how programs are used:

- 3.) How do programs affect your daily life?

Name: _____ Date: _____

Computer Programming [6]

- 1.) Define programming.

- 2.) List at least three examples of how programs are used:

- 3.) How do programs affect your daily life?

Name: _____ Date: _____

Python: Variable Naming Conventions

1. Variable names can only contain numbers, letters and underscores.
2. Variable names cannot start with a number.
3. The case of a variable name matters.
4. Variables names cannot be a Python keyword.
5. When you reference the *value* of a string variable in Python interactive mode (or the shell), then you must enclose it in quotation marks.

Helpful Tips for Naming Variables

1. Choose descriptive names.
2. Choose a naming method and stay consistent throughout the program.
3. It is common to use all lowercase for variable names, but it is not mandatory.
4. Avoid starting a variable name with an underscore. (Note: doing otherwise increases the likelihood of your program interfering with Python internal processes).
5. Avoid long variable names.

Concept Check

Directions: Define the following list of words with a partner in the left side of the table. Use the right side of the table to include the presented definition after class discussion.

Variable (Your definition)

Variable (Class definition)

Variable Type (Your definition)

Variable Type (Class definition)

String Variable (Your definition)

String Variable (Class definition)

Keyword (Your Definition)

Keyword (Class Definition)

Python Shell (Your definition)

Python Shell (Class definition)

Tables of Python Information

Python: Keywords [7]

False	class	finally	is	return	None
continue	for	lambda	try	True	def
from	nonlocal	while	and	del	global
not	with	as	elif	if	or
yield	assert	else	import	pass	break
	except	in	raise		

Python: Operators

Name	Meaning	Name	Meaning
+	Addition	+=	Addition Assignment
-	Subtraction	-=	Subtraction Assignment
*	Multiplication	*=	Multiplication Assignment
/	Float Division	/=	Float Division Assignment
//	Integer Division	//=	Integer Division Assignment
**	Exponentiation	**=	Exponentiation Assignment
%	Remainder	%=	Remainder Assignment

Python: Escape Sequences

Escape Sequence	Function
\'	Print the next character as a single quote, not as a string closer.
\"	Print the next character as a double quote, not as a string closer.
\n	Print a new line character.
\t	Print a tab character
\\	Print the next character as a backslash, not as an escape character.

Name: _____ Date: _____

Python: Valid and Invalid Variable Names

Directions: For each variable name listed, decide whether it is valid or invalid, and whether or not it follows the list of helpful conventions. If you answer “No” for either or both of these columns, then declare a suggested correction. Finally, explain your reasoning.

Variable Name	Valid? (Yes or No)	Best Convention? (Yes or No)	Suggested Correction	Reasoning
r	Yes	No	radius	More descriptive
1st_Score				
import				
LastName				
_piSquared				
No_1093827365780				
German_Shepherd				
My Variable				
nAmInG				

Python: Variable Types

Directions: Determine the possible variable type for the following variables. (Choose from string, floating point or integer. Can be more than one).

Variable Name	Variable Type
firstName	
age	
temperature	
x	
shirt_size	

Name: _____ Date: _____

Python: Expressions and Assignments

Directions: Given the following variable assignments, what is the result of the expressions?

Code 1:

```
w = 2
y = 7
z = 12
```

Expression 1: `print(w * z)`

Answer: _____

Expression 2: `print(w * z + y)`

Answer: _____

Expression 3: `print((w + z) * y)`

Answer: _____

Expression 4: `print(w / z)`

Answer: _____

Code 2:

```
a = "a"
b = "b"
s = "This is fun."
```

Expression 1: `print("a = " + a)`

Answer: _____

Expression 2: `print("a = " + 'a')`

Answer: _____

Expression 3: `print(a + b)`

Answer: _____

Expression 4: `print(s)`

Answer: _____

Python: Equations

Directions: Convert the following algebraic equations into Python equations. (Don't forget to declare all of your variables. You will need to use `math.pi` and `math.sqrt`).

$$V = \frac{4}{3}\pi r^3$$

$$D = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

Name: _____ Date: _____

Exit Ticket

Directions: Read the given code and its output carefully. Answer all associated questions below. (Hint: the symbol: # is known in Python as a token that begins a comment).

Code:

```
##Variable Assignment: Exit Ticket
##Python Lesson #2

##Title:      Hello, World!
##Description: Python program that prints a greeting

#Prompt the user for a value and assign the input into the variable First_name
First_name = input("What is your first name? ")
#Print a greeting
print("Hello, " + First_name + "!")
```

Output:

Hello, Prof. Johnson!

Analysis Questions:

1. What does the input() function do?
2. What is considered the assignment statement?
3. What does the assignment statement do?
4. What is the variable type of the assignment statement?
5. Explain in your own words how the print statement works.

Bonus:

What does the # do?

Name: _____ ANSWER KEY _____ Date: _____

Computer Programming [6]

1.) Define programming.

The process of creating a set of instructions that tell a computer how to perform a task
Multiple languages

2.) List at least three examples of how programs are used:

Robots

Self-Driving cars

Doctors and medical science

Games

3.) How do programs affect your daily life?

Variable between each student

Name: _____ ANSWER KEY _____ Date: _____

Python: Variable Naming Conventions

1. Variable names can only contain numbers, letters and underscores.
2. Variable names cannot start with a number.
3. The case of a variable name matters.
4. Variables names cannot be a Python keyword.
5. When you reference the *value* of a string variable in Python interactive mode (or the shell), then you must enclose it in quotation marks.

Helpful Tips for Naming Variables

1. Choose descriptive names.
2. Choose a naming method and stay consistent throughout the program.
3. It is common to use all lowercase for variable names, but it is not mandatory.
4. Avoid starting a variable name with an underscore. (Note: doing otherwise increases the likelihood of your program interfering with Python internal processes).
5. Avoid long variable names.

Concept Check

Directions: Define the following list of words with a partner in the left side of the table. Use the right side of the table to include the presented definition after class discussion.

Variable (Your definition)

Variable (Class definition)

A name that refers to a value

Variable Type (Your definition)

Variable Type (Your definition)

A category of values

String Variable (Your definition)

String Variable (Class definition)

A type that represents a sequence of characters

Keyword (Your Definition)

Keyword (Class Definition)

A reserved word that is used by the compiler to parse a program

Python Shell (Your definition)

Python Shell (Class definition)

Interactive mode for Python programming

Name: _____ ANSWER KEY _____ Date: _____

Python: Valid and Invalid Variable Names

Directions: For each variable name listed, decide whether it is valid or invalid, and whether or not it follows the list of helpful conventions. If you answer “No” for either or both of these columns, then declare a suggested correction. Finally, explain your reasoning.

Variable Name	Valid? (Yes or No)	Best Convention? (Yes or No)	Suggested Correction	Reasoning
r	<u>Yes</u>	<u>No</u>	<u>radius</u>	<u>Descriptive</u>
1st_Score	<u>No</u>	<u>No</u>	<u>First_Score</u>	<u>Initial number</u>
import	<u>No</u>	<u>No</u>	<u>May vary</u>	<u>Keyword</u>
LastName	<u>Yes</u>	<u>Yes</u>	<u>n/a</u>	<u>Good to go!</u>
_piSquared	<u>Yes</u>	<u>No</u>	<u>piSquared</u>	<u>Starts with _</u>
No_1093827365780	<u>Yes</u>	<u>No</u>	<u>account_no</u>	<u>Too long</u>
German_Shepherd	<u>Yes</u>	<u>Yes</u>	<u>n/a</u>	<u>Good to go!</u>
My Variable	<u>No</u>	<u>No</u>	<u>MyVariable</u>	<u>Space inbetween</u>
nAmInG	<u>Yes</u>	<u>No</u>	<u>naming</u>	<u>Hard to read</u>

Python: Variable Types

Directions: Determine the possible variable type for the following variables. (Choose from string, floating point or integer. Can be more than one).

Variable Name	Variable Type
firstName	<u>str</u>
age	<u>int</u>
temperature	<u>float</u>
x	<u>str int float</u>
shirt_size	<u>str</u>

Name: _____ ANSWER KEY _____ Date: _____

Python: Expressions & Assignments

Directions: Given the following variable assignments, what is the result of the expressions? If there is an error (syntax, semantic or runtime), then write the error type on the answer line.

Code 1:

```
w = 2
y = 7
z = 12
```

Expression 1: print(w * z)	Answer: _____ 24 _____
Expression 2: print(w * z + y)	Answer: _____ 31 _____
Expression 3: print((w + z) * y)	Answer: _____ 98 _____
Expression 4: print(w / z)	Answer: _____ 0.16666666666666666 _____

Code 2:

```
a = "a"
b = "b"
s = "This is fun."
```

Expression 1: print("a = " + a)	Answer: _____ a = a _____
Expression 2: print("a = " + 'a')	Answer: _____ a = a _____
Expression 3: print(a + b)	Answer: _____ ab _____
Expression 4: print(s)	Answer: _____ This is fun. _____

Python: Equations

Directions: Convert the following algebraic equations into Python equations. (Don't forget to declare all of your variables. You will need to use math.pi and math.sqrt).

$$V = \frac{4}{3}\pi r^3$$

```
R = input(radius)
V = (4/3)*(math.pi)*(r**3)
```

$$D = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

```
X2 = input(final horizontal distance)
X1 = input(initial horizontal distance)
Y2 = input(final vertical distance)
Y1 = input(initial vertical distance)
diffX = (X2 - X1)**2
diffY = (Y2 - Y1)**2
D = math.sqrt(diffX + diffY)
```

Name: _____ ANSWER KEY _____ Date: _____

Exit Ticket

Directions: Read the given code and its output carefully. Answer all associated questions below.
(Hint: the symbol: # is known in Python as a token that begins a comment).

Code:

```
##Variable Assignment: Exit Ticket
##Python Lesson #2

##Title:      Hello, World!
##Description: Python program that prints a greeting

#Prompt the user for a value and assign the input into the variable First_name
First_name = input("What is your first name? ")
#Print a greeting
print("Hello, " + First_name + "!")
```

Output:

Hello, Prof. Johnson!

Analysis Questions:

1. What does the input() function do?
Takes information in from the keyboard.
2. What is considered the assignment statement?
First_name = input("What is your first name? ")
3. What does the assignment statement do?
Assigns the keyboard input value into the First_name variable.
4. What is the variable type of the assignment statement?
String
5. Explain in your own words how the print statement works.
Moving from left to right, it prints Hello, then it references the First_name variable and prints the input value of Prof. Johnson, then it prints !. This process is called string concatenation.

Bonus:

What does the # do?

means that all information on the corresponding line is disregarded by the compiler.

Annotated Bibliography

[1] Downey, A. (2012). *Think Python: How to Think Like a Computer Scientist* (Vol. 2.0.17). Needham, MA: Green Tea Press. From <http://www.greenteapress.com/thinkpython/thinkpython.pdf>

This online textbook was used for excerption within the Python Data Types, Variables and Expressions lesson plan as part of the Computer Science module. This reference aided in the completion of providing definitions for the key concepts and definitions sections and for the associated worksheets. This book was useful because of its layout and completeness. The lesson expands upon the material used from this book as it uses the material in the creation of worksheets and activities that are not provided in the textbook.

[2] Standards Aligned System. (n.d.). Retrieved from <https://www.pdesas.org/>

This website was used in each lesson in the Computer Science module to select proper Pennsylvania State standards, which are based in Common Core, that each lesson is centered around.

[3] Nsta. (n.d.). Science and Engineering Practices. Retrieved January 18, 2019, from <https://ngss.nsta.org/PracticesFull.aspx>

This website used in every lesson in the Computer Science module to find Standards for Scientific Practices that are applicable in each lesson.

[4] Nsta. (n.d.). Disciplinary Core Ideas. Retrieved from <https://ngss.nsta.org/DisciplinaryCoreIdeasTop.aspx>

This website was used in each lesson in the Computer Science module to select appropriate disciplinary core ideas set forth by the NSTA that are at the center of each lesson.

[5] Nsta. (n.d.). Crosscutting Concepts. Retrieved from <https://ngss.nsta.org/CrosscuttingConceptsFull.aspx>

This website was used in each lesson in the Computer Science module to selecting appropriate crosscutting concepts set forth by the NSTA that apply to each Python lesson.

[6] What is Programming? (n.d.). Retrieved from <https://www.khanacademy.org/computing/computer-programming/programming/intro-to-programming/v/programming-intro>

This video on Khan Academy is used as an engagement in the Python Data Types, Variables and Expressions lesson plan as part of the Computer Science module. Questions were developed based on this video for students to answer.

[7] List of Keywords in Python. (n.d.). Retrieved from <https://www.programiz.com/python-programming/keyword-list>

Python by Programiz is an online tool that is useful for understanding the underlying concepts of computer programming in Python. The table of keywords in Python was excerpted to be utilized as an instructional sheet for students to reference while completing the exploration activities.