3.5. CODESTERS LESSON 5

Vocabulary

variable: _____

_____

list: _____

_____

loop: _____

_____

Check for Understanding (Activity 12)

1. How are items separated in a list?

2. What keyboard characters surround a list?

Exit Ticket

1. How many spaces must you indent the commands inside of a loop block?

4. STEM 2 ASSESSMENTS

The following formative assessments may be used to determine progress of students and to plan instruction with the STEM 2 syllabus. These documents are a work in progress and will be supplemented as the course is taught during the 2017-2018 school year.

4.1. PYTHON FORMATIVE ASSESSMENT 1

Name _____

1. Write a line of code that will print, "*Go Bulldogs!*".

2. How do you enter a comment in Python?

3. A) What do the following lines of code output?

   print(2 * 3)

   print(2 ** 3)

   B) Why do they give a different answer?

4. Why doesn't this code work?

   A = 22

   print(a)

5. Correct the following code. Write the correct code underneath the given code. (There may be more than one mistake.)

   Print("Hello. What is your name?)"

4.2. PYTHON FORMATIVE ASSESSMENT 2

Name _____

1. Why would this code not work?

   print(a)

   a = 45

2. What do the following lines of code output? Explain why the outputs are different.

   print(19 – 12)

   print(“19 – 12”)

3. What would the following line of code output?

   print(“This \nis \nmy \nsample.”)

4. Write a Python program to create a variable, assign a value to that variable, and print the value stored in that variable.

5. Circle the legal variable names in Python.

my_name	super bad	144xy
color	first number	variable_fun

4.3. PYTHON FORMATIVE ASSESSMENT 3

Name _____

1. Try this example Python code first:

   print("Don't is short for do not")

   Then, try this code:
   print('Isn't this going to work?')

   Which code did not work, the first or second? Why not? What error message did it give?

   Can you fix it? Compare it with the example that worked. Write your code below.

2. Try this example Python program:

   ```
   zombies = 5
   zombies = 10
   ghosts = 3
   ghosts = ghosts + 2
   bad_guys = zombies + ghosts
   print("There are", bad_guys, "bad guys.")
   ```

   When you ran this program, what result did it print? Explain the result.

3. Examine and try the following program:

   ```
   children = 30
   candy = 5
   total = children*candy
   print(total)
   ```

   The above program calculates the total number of pieces of candy a class of 30 children would need for each child to receive 5 pieces. How many total pieces of candy are required? _____

   Write a program to calculate the total number of brownies required for a class party if we know the number of children attending the party and the number of brownies each child should receive. Choose your own number of children and how many brownies each should get!

4.4. PYTHON SUMMATIVE ASSESSMENT

Name _____

1. Which of these is not a Python data type?

    a. int
    b. float
    c. iter
    d. str

2. How can we create a new line (move down) in a print function?

3. Which operator is used to compare two numbers for equality?

4. What is the difference between 12 + 13 and “12” + “13”?

5. Explain the difference between an integer, a float, and a string.

6. How do you enter a comment in a program?

7. What is the difference between == and =

8. Correct the following lines of code:

    a. (print”How are you?”)
    b. print(‘Go Team”)
    c. print(‘Kelly’s team is winning’)

5. PYTHON QUICK REFERENCE GUIDE

Adapted from iNTERFACEWARE

This Quick Reference Guide was created with the intent that students would have it readily available to refer to when programming. It is also a work in progress and could be supplemented as the course is taught during the 2017-2018 school year. Instructors may alter the document as necessary for their classes.

5.1. PRINT COMMAND

print displays values on the screen

Table 1 print Function Examples

Input	Output	Input	output
print("Hello")	Hello	print("2 + 3")	2 + 3
print(2 + 3)	5	print(Hello)	SyntaxError

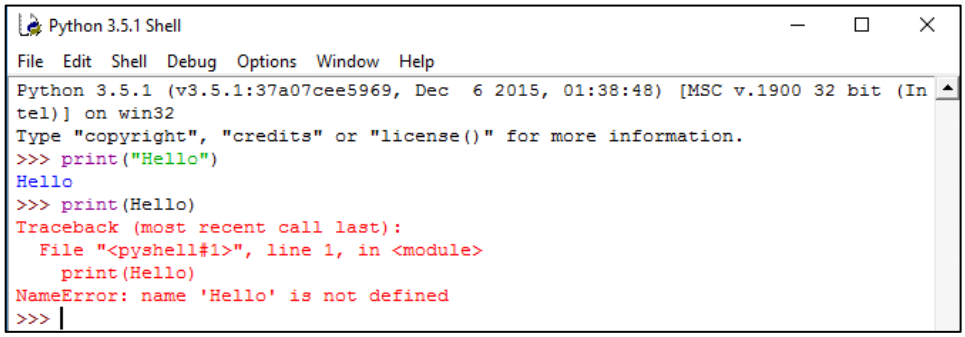

Figure 5 print Function Examples ©2001-2018 Python Software Foundation. Screenshot by author.

5.2. ESCAPE CODES

Table 2 Escape Codes

Escape Code	Meaning	Escape Code	Meaning
\\	Backslash	\'	Single quote
\"	Double quote (useful in strings enclosed in double quotes)	\n	Line feed (move down)
\r	Carriage return (move to the left)	\t	Tab

5.3. CALCULATIONS AND VARIABLES

A **variable** can be any combination of letters, digits, and underscore characters. Variables in Python are ***case sensitive*** (variable and VARIABLE are not the same.) A variable cannot be a digit. Multi-word variable names should be separated by underscores (see more examples in Table 3).

Table 3 Variable Name Examples

Legal Variable Names	Illegal Variable Names	Legal, but not Proper Variable Names
first_name	first name	FirstName
distance	9ds	firstName
ds9	%correct	X

Python supports the standard arithmetic operations on integers and floating point numbers. Calculations follow the order of operations.

Table 4 Arithmetic Operators

Symbol	Meaning	Symbol	Meaning
+	Addition	*	Multiplication
-	Subtraction	/	Division
%	Remainder from division	**	Power

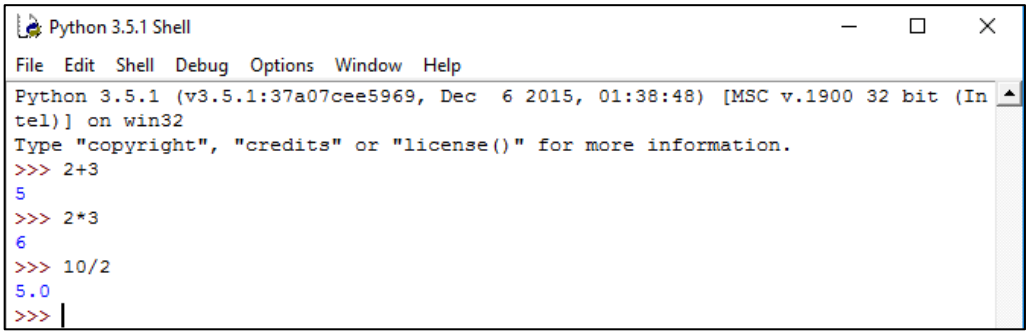

Figure 6 Arithmetic Operation Examples ©2001-2018 Python Software Foundation. Screenshot by author.

Python also uses comparison operators.

Table 5 Comparison Operators

Symbol	Meaning	Symbol	Meaning
==	Equal to	<=	Less than or equal to
!=	Not equal to	>	Greater than
<	Less than	>=	Greater than or equal to

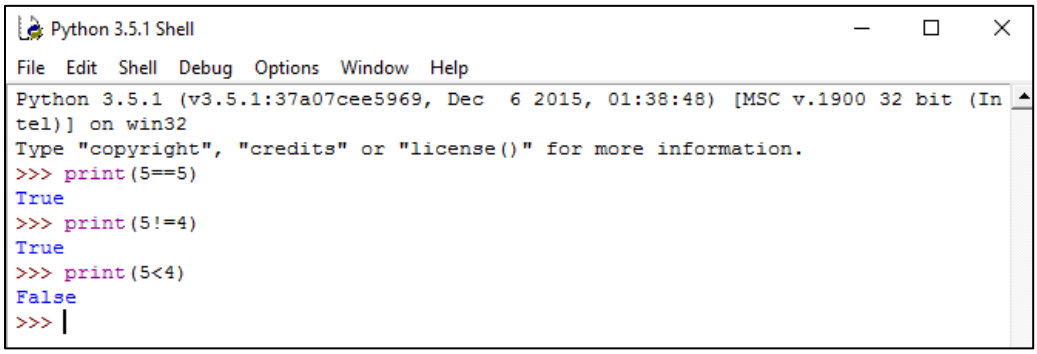

Figure 7 Comparison Operator Examples ©2001-2018 Python Software Foundation. Screenshot by author.

5.4. DATA TYPES

Numbers can be *integers* (int) or *floating* point values (float).

42 3.14159

Strings (str) are enclosed in quotes, and can contain any printable character:

“test” “Hello, world!”

5.5. COMMENTS

Comments in Python are for the programmer or other humans to read. Everything after the # character is ignored.

```
#This is a comment.
print("Hello") #This is also a comment
```

Multi-line comments use three single quotes in a row.

```
'''
This is a
multi-line comment.
'''
```

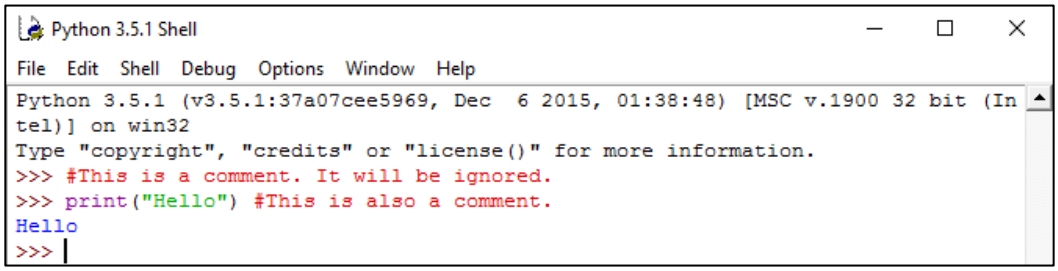

Figure 8 Comment Example ©2001-2018 Python Software Foundation. Screenshot by author.

5.6. COLOR CODES

Table 6 Color Codes

Code	Color	Code	Color
Strings	Green	Comments	Red
Output	Blue	Reserved words	purple

6. PROJECTS

Adapted from ***Five Mini Programming Projects for the Python Beginner*** by Shelly Tan. The original article is available at https://knightlab.northwestern.edu/2014/06/05/five-mini-programming-projects-for-the-python-beginner/ and solution videos are available at https://www.youtube.com/playlist?list=PLhP5GzqIk6qsYjU_3tod0nqoWGXlq9RvF.

6.1. PROJECT 1: DICE ROLLING SIMULATOR

Goal: Write a program that simulates rolling dice.

Concepts:

- Random
- Integer
- Print
- While loops

Brief Description:

Students will write a program that randomly chooses a number between 1 and 6. The program will print that number. The program will then ask the user if they would like to roll again.

The student needs to set a minimum and maximum number that the die can produce (minimum of 1 and maximum of 6 typically). Also, a function that randomly chooses a number within that range and prints it is required.

6.2. PROJECT 2: GUESS THE NUMBER

Goal: Write a program that randomly generates a number and allows the user to try to guess the number.

Concepts:

- Random function
- Variables
- Integers
- Input/Output
- Print
- While loops
- If/Else statements

Brief Description:

Students will write a program that uses the random module to randomly generate a number unknown to the user. The user can then guess a number (input). The program must indicate whether the user's guess is correct or incorrect with appropriate output. In addition, the program must tell the user if their guess is too high or too low. The program should also provide an appropriate positive indication when the guess is correct.

6.3. PROJECT 3: MAD LIBS GENERATOR

Goal: Write a program that applies user provided input to create a story template.

Concepts:

- Strings
- Variables
- Concatenation
- Print

Brief Description:

Students will write a program that prompts the user to provide input that is part of speech appropriate. The program must then use the data in a premade story template and print it.

6.4. PROJECT 4: TEXT BASED ADVENTURE GAME

Goal: Write a program that allows users to move through rooms.

Concepts:

- Strings
- Variables
- Input/Output
- If/Else statements
- Print
- List
- Integers

Brief Description:

Students will write a program that will let users move through rooms based on their input and get descriptions of each room. Students will establish directions in which user can move and a way to track the distance the user has moved. The program will then print room descriptions. Students will include limits for how far the user can move, essentially creating "walls" around the rooms. The program will include messages to the user when limits are reached.

This project can be as basic or as complex as the programmer chooses.

6.5. PROJECT 5: HANGMAN

Goal: Write a program that allows a user to guess a word.

Concepts:

- Random
- Variables
- Boolean
- Input/Output
- Integers
- Char
- Strings
- Length
- Print

Brief Description:

Students will write a program to create a word guess game. The program must allow users to input letter guesses. The program must have a limit on how many guesses the user is allowed. Students will include functions to check to see if the input letters are in the hidden word, to print those letters, and to count input variables for guess limits.