

CS 171 Computer Programming I
Lab 1
Content by Professor Lisa Olivieri of Chestnut Hill College
Modified for Drexel by Professors Mark Boady and Adelaida Medlock

Detailed instructions to the lab assignment are found in the following pages.

- Complete all the exercises and type your answers in the space provided.
- For the last question of this lab you must submit a screenshot your source code (.py file) running in Thonny

What to submit:

- Lab sheet in PDF to Gradescope
- Questions must be tagged in Gradescope
- Your lab partner must be added to the submission
- Only one submission per lab group

Submission must be done via Gradescope

Student Name(s): Tony Kabilan Okeke

User ID(s) (abc123): tko35

Possible Points: 80

Your score out of 80:

Lab Grade on 100% scale:

Python Activity 1: Introduction to Python

Learning Objectives

Students will be able to:

Content:

- Explain how to display data in Python
- Explain how to create a comment in Python
- Determine the difference between a *string literal* and a *number*
- Explain how to input data using Python
- Explain the meaning and purpose of a variable
- Determine if a variable name is valid
- Explain concatenation and the use of “+”

Process:

- Create **print** statements in Python
- Create *Python* code that displays results to calculated addition facts
- Discuss problems and programs with all group members
- Create **input** statements in Python
- Create *Python* code that prompts the user for data and stores it in a variable
- Create valid and good variable names

Prior Knowledge

- Be able to input and execute Python code using the Python IDE

Critical Thinking Questions (30 points):

```
1 print("Go!")  
2
```

1. (2pts) Enter the Python program shown above. What does it do?

The program prints the string literal “Go!” in the console.

FYI: A “string literal” is a sequence of characters surrounded by double quotation marks (“ ”).

2. (6pts) Type and execute following code. What output is produced? Indicate if there is a problem.

- a. `print(“Hello, my name is Pat!”)`

Prints the string literal “Hello, my name is Pat!” in the console.

- b. `print(Hello, my name is Pat)`

Returns a syntax error.

c. `print("Hello.\nMy name is Pat")`

Prints string in the console, with "Hello." on one line and "My name is Pat" on the next.

3. (2pts) What caused the different output format for samples "a" and "c" in question 2?

In "c", the line break character "\n" is included in the string literal. It is not included in "a".

4. (8pts) What do you think the following Python statements output? Enter the statements in the interactive mode of the Python interpreter to verify your answers.

a. `print(2+5)`

Returns the integer 7

b. `print(2*5)`

Returns the integer 10

c. `print("2+5")`

Prints the string "2+5" in the console.

d. `print("Age:",20)`

Prints the string "Age: 20" in the console.

5. (8pts) Examine the output for each statement in question 4.

a. What is the difference in the output for the statements in "a" and "c" of question 4?

The statement in "a" returns the integer 7, while the statement in "c" prints the string "2+5" in the console.

b. What caused the difference?

In "a", the mathematical expression 2+5 was the argument passed to the print function. In "c" the string literal "2+5" was the argument passed to the print function.

c. Which statements contain a *string literal*?

The statement in "c" contains the string literal.

d. What does the comma (,) do in the print statement in part "d" of question 4? How does it affect the spacing of the output?

The comma is used to concatenate the arguments to the print function. It causes the arguments to be printed together with a single space between them.


6. (2pts) Examine the following code and its output. What do the first three lines of the program do?

The first three lines are comments. They provide information about the program, but are not executed.

```

1 # Programmer: Pat Smith
2 # Date: 4/3/2019
3 # Description: This program prints a welcome statement
4
5 print("Hello, Pat!")
6 print("Welcome to Programming in Python!")
7

```



```
>>> %Run Lab1_Key.py
```

Output

```

Hello, Pat!
Welcome to Programming in Python!

```

7. (2pts) In the program from question 6, what would happen if you placed a “#” in front of

```
print("Hello, Pat!")
```

If a “#” were placed in front of the statement, it would no longer be executed. The output of the program would only be “Welcome to Programming in Python”.

Application Questions: Use the Python program mode to design and check your work (4 points)

1. (2 points) Create a Python program containing three statements to print the following output. Copy your program statements in the space below.

```

print("Congratulations!")
print("You just created")
print("Your first Python program")

```

```

Congratulations!
You just created
your first Python program

```

2. (2pts) Create a Python program containing two statements that prints the output to the right. *Have the program calculate the answers to the two arithmetic problems.* Copy your program statements in the space below.

```

print("34 + 123 =", 34+123)
print("56 * 97 = ", 56*97)

```

```

>>> =====
>>>
34 + 123 = 157
56 * 97 = 5432
>>>

```

Critical Thinking Questions (36 points):

```

name = input("What is your name? ")
print("Your name is", name)

```

1. (2pts) Enter and execute the Python program. What is printed on the screen when the Python program is executed?

The prompt “What is your name?” is printed in the console. Once a response to the prompt is provided, the statement “Your name is ...” is printed in the console with the response in place of the ellipsis.

FYI: `input()` and `print()` are *functions* in Python.

2. (4pts) Examine the first line of Python program: `name = input("What is your name? ")`

- a. What appears on the screen when this line of code is executed?
The prompt “What is your name?” appears on screen.

FYI: The words that appear on the screen and tell the user what to enter are known as a *prompt*.

FYI: `name = input("What is your name? ")`

The word **name** in the Python code is a *variable* – a name given to a memory location used to store data.

- b. What happens to the data the user entered?
It is concatenated to the string “What is your name” and printed in the console.
3. (4pts) Explain the errors that occur when you execute each of the following lines of Python code.
- a. `name? = input("What is your name?")`
A syntax error is produced due to the question mark after `name` (variable names cannot contain math symbols).
- b. `your name = input("What is your name?")`
A syntax error is produced because variable names cannot contain spaces.
- c. `1st_name = input("What is your name?")`
A syntax error is produced because variable names cannot begin with numbers
- d. `from = input("Where were you born?")`
A syntax error is produced because reserved words (like `from`) cannot be used as variable names.
4. (2pts) Examine the errors that occurred when executing the lines of code in question 3. Then examine the following lines of valid code.

```
name2 = input("What is your name?")
your_name = input("What is your name?")
yourName = input("What is your name?")
```

List the rules that you need to follow to create a valid *variable* name.

- Only letters, digits and underscore characters can be used.
- The first character must be a letter or underscore.
- Reserved words cannot be used as variable names.
- Variable names cannot contain spaces or other special characters.
- Variable names are case sensitive and thus must be consistent throughout a program.

5. (8pts) Are the following variable names **valid**? Are they **good** names? Why or why not?

| Variable name | Comments about variable name |
|-----------------|----------------------------------------------------------------------------------------------|
| Price | This is a good, valid variable name since it adheres to all rules stated above. |
| costoffirstitem | This is a valid variable name, but it is not good because it is too long. |
| Ic | This is a valid variable name, but it is not good because it is not particularly meaningful. |
| firstName | This is a good, valid variable name since it adheres to all rules stated above. |

6. (2pts) Execute the following lines of code. Is the output what you would expect? Why or why not?

```
name = input("What is your name? ")
print("Your name is", Name)
```

As I expected, the code returns an error because the `Name` variable is not defined. The first statement defines the `name` variable. `Name` and `name` are two distinct variables.

7. (10pts) Use the following set of Python statements to answer the questions below.

```
print("Your name is", "Pat.")
print('Your name is', "Pat.")
print("Your name is" + "Pat.")
print("Your age is", 20)
print("Your age is" + 20)
```

- a. State the output for each of line of code.

Your name is Pat.

Your name is Pat.

Your name isPat.

Your age is 20

TypeError: can only concatenate str (not “int”) to str

- b. How are the first two print statements different? Does the difference affect the output?

In the first print statement, both string literals are enclosed in double quotes. In the second print statement the first string literal is enclosed in single quotes. This difference does not affect the output

- c. Notice that some statements include a comma (,) between the two literals being printed and some statements use a “+”. Do they produce the same output?

No, they produce different outputs. In statements with the comma, the two arguments are concatenated with a space between them. In statements with the plus sign, the two arguments are concatenated without a space between them.

- e. Explain the purpose of the comma.

The comma separates arguments to the print function.

- f. Why does the last print statement crash the program? What would you do to correct it?

Integers cannot be concatenated to strings. To fix it, I would place the integer 20 in double quotes.

FYI: “+” concatenates two strings. The strings can be string literals or a variable containing string literals.

8. (2pts) State what is displayed on the screen when the following program is executed:

```
name = input("Enter your name: ")
ID = input("Enter your student ID number: ")
course = input("Enter your course number: ")

print(name + "'s ID is " + ID + "\nand is enrolled in " + course)
```

Once the program is executed, the following prompts are displayed in the console: “Enter your name:”, “Enter your student ID number:”, “Enter your course number:”.

Once the prompts have been responded to, the provided inputs are concatenated with some other text and printed in the console. The ellipsis below are replaced with the user responses.

“...’s ID is ...”

and is enrolled in ...”

9. (2pts) What caused the output in the print statement in question 8 to be printed on more than one line?

The presence of the line break character “\n” in the final print statement.

Application Questions:

Use the Python Interpreter to enter your code and check your work (10 points)

1. (2pts) State a good variable name for an employee’s ID number: empID
2. (2pts) Write a line of Python code that prompts the user for the name of their favorite ice cream and stores it in a valid variable name.

```
favIC = input("What's your favorite ice cream? ")
```

3. (6pts) **Crazy Sentence Program.** Create a program that prompts the user for the name of an animal, a color, the name of a vehicle, and the name of a city. Then print a sentence that contains the user input in the following order: color, animal, vehicle, city. Include the additional words in the sample output as part of your output. Example: Assume the user enters the words: tiger, green, motorcycle, and Wildwood. The output would be: *The green tiger drove the motorcycle to Wildwood.*

Once your program is complete, take a screenshot of the code and output of the program. You may need multiple screenshots to capture all your code. Paste the screenshot below.

Your code **must** include comments with the names of all group members and the date the code was written.

```
crazy_sentence.py X
crazy_sentence.py > ...
1  # Author: Tony Kabilan Okeke (tko35)
2  # Last Modified: 01/06/22
3  # Description: This program takes user input and concatenates
4  #              it into a sentence.
5
6  # Collect user input
7  print("Please provide the following:")
8  color = input("A color: ")
9  animal = input("An animal: ")
10 car = input("A vehicle: ")
11 city = input("A city: ")
12
13 # Concatenate input
14 print("The {} {} drove the {} to {}".format(color, animal, car, city))
```

PROBLEMS OUTPUT **TERMINAL** DEBUG CONSOLE Python - Week 1 + -

```
[kabil] >> ~/../cs171/Week 1 /bin/python3.9 "/home/kabil/tko35/cs171/Week 1/crazy_sentence.py"
Please provide the following:
A color: red
An animal: gorilla
A vehicle: cyber truck
A city: Philly
The red gorilla drove the cyber truck to Philly.
```