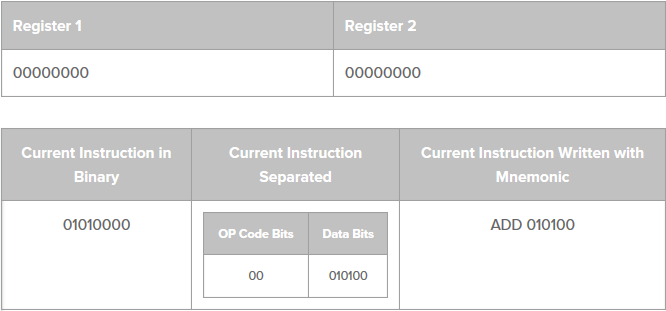
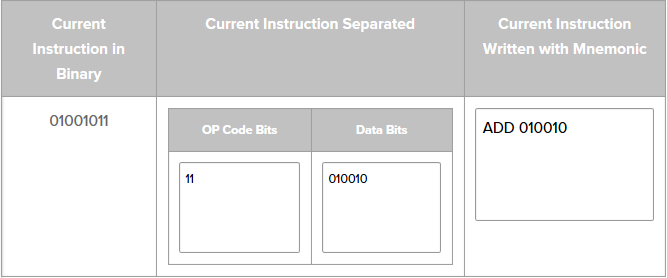
**PLTW Activity 1.3.1**

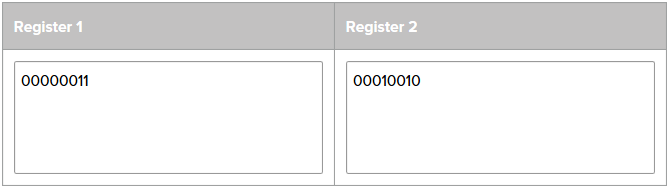
**Notes**

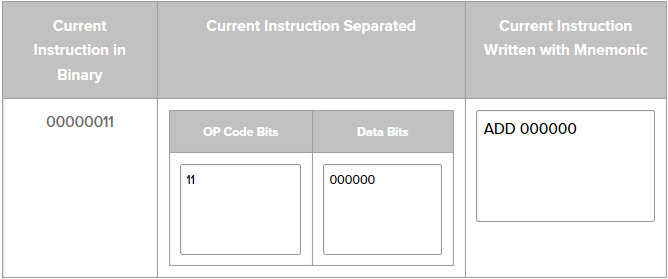
Machine code, or machine language, consists of zeros and ones that represent simple instructions executed by a computer’s processor. For an assembly language to operate, a register—a miniscule part of the processor microchip that stores bits of data—holds typically 32 or 64 bits of data in modern computers. Instructions for assembly languages are constructed from op codes and data, and additional data can be provided as input or written as output from/to the registers.

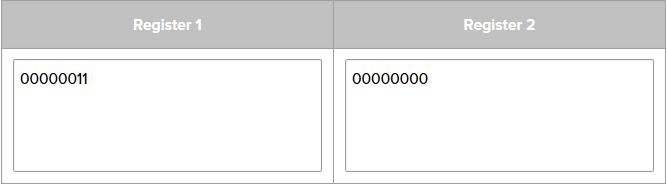
**Journal**

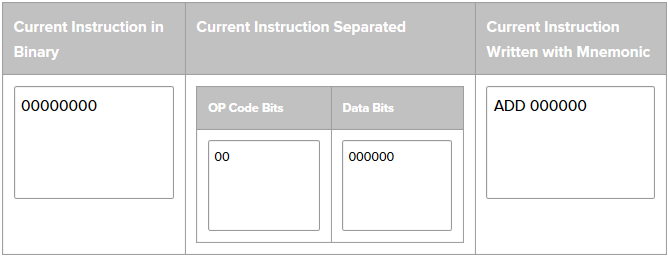


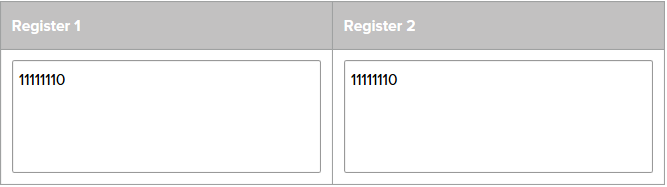


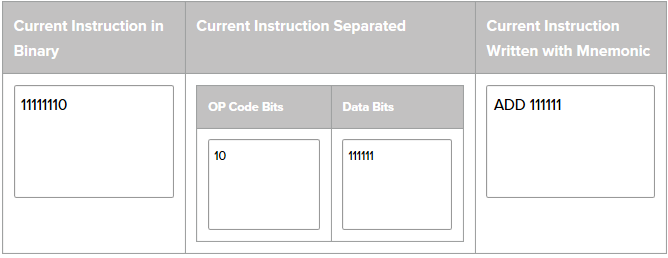


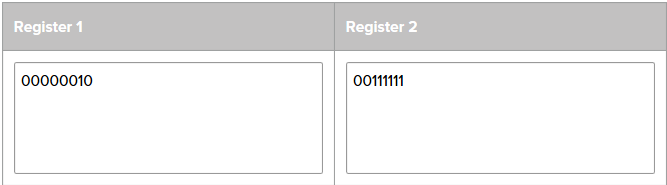


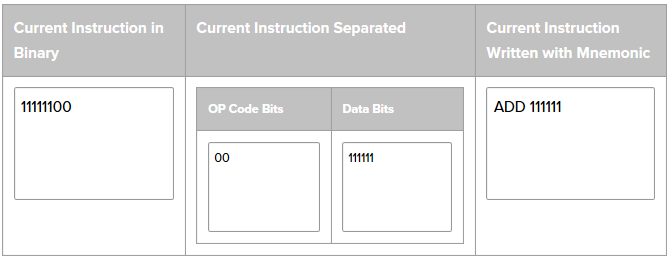


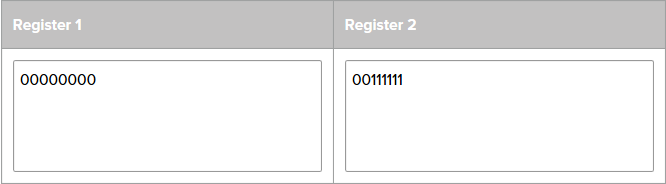












**Conclusion**

1. “The first program in this exercise was actually pulled directly from the bit representation of this Word® document.”
   1. How did this activity change your perception of what programs are? When someone uses a program, they initially detect the application or IDE for their respective coding. However, behind the scenes, the compiler is converting the code within the application to binary code—i.e., text, computer processor instructions, or other data utilizing any two-symbol system, especially the binary number of 0 and 1—which your computer can comprehend and to which it can react accordingly.
   2. How did it change your perception of what data are? Data in its most basic form is 0s and 1s. While I have heard that statement before, following this activity I have a better grasp as to why.
2. Real assembly languages are similar to the simplified one used in this activity, though the way they process instructions and their instruction sets may vary greatly. For example, most assembly languages give programmers the ability to manipulate multiple registers, specifying which registers to work with, or they grant access to data outside of the CPU. What do you think the challenges of programming in an assembly language instead of a high-level language like Scratch would be? An assembly language requires knowledge of its syntax and API, whereas *Scratch* necessitates that you “drag and drop” blocks serving as code.
3. Run a shade of yellow represented by decimal RGB values of 255, 255, and 102. What are the end contents of the registers? 255 has a binary value of 11111111. 102 has a binary value of 1100110. The end contents of the registers are 11 ADD 111111, 11 ADD 111111, and 10 ADD 1100110.

**PLTW Activity 1.3.2**

**Notes**

The four native types (i.e., the types of data built into programming language, often faster and more efficient with resources) of data are: integers, floats/floating points (i.e., a native type representing rational numbers to limited precision), strings, and Booleans.

The stateof a program comprises the data that describes the existence of a program at a given time; the state is similar to a summary of a program. The syntax of a programming language is the grammar of that language, which defines what specific sequences of characters are allowed, what they mean, and how they can be arranged.

Call functions signal the compiler or interpreter to execute a function or method defined elsewhere, replacing the call with a return value, if applicable. A built-in function is a set of functions available to the programmer without having to use “import,” “include,” or “using” statements. A namespace is a set of variables and function names that have been reserved by the compiler/interpreter.

An interpreter converts a program written in a higher-level language into a lower-level language and executes it, commencing execution before converting the entire program. A compiler is a computer program created to read an entire program and convert it into a lower-level language and ultimately, assembly language used by the processor. Functions validate (i.e., ensure that the values stored in variables are of the correct type and/or within appropriate value ranges) input arguments (i.e., the values that the programmer provides in the function call).

A test suite is a software package designed to evaluate the accuracy or effectiveness of another software solution.

**Journal**

Activity 1.3.2 was completed at 4:39 PM on November 11, 2017.

Christian Leon instructed Amber on the basics of manipulating the Unity Engine (Direct Assistance).

**Conclusion**

1. Read the introduction to this activity again. Describe something you would like to have automated by a program. In a perfect world, I would have access to an AI that could complete my homework on the days when I am ill, especially tired, or feeling the effects of my karate training and initiate and implement certain tasks, such as household chores.
2. What are the native data types you learned about in this activity? The native data types I learned about in this activity are: integers, floats, strings, and Booleans.
3. What are some differences between the command line of the interpretive IPython session and the code editor where you edit a file of code? An interpretive IPython session contains the syntax and API of that language, whereas a text editor enables you to edit the raw code without any “autocorrect” functions.

**PLTW Activity 1.3.3**

**Notes**

Computers can “make decisions” using *if* statements” (i.e., Boolean or variable checks). An operator (i.e., ==, >>, <<, !=, &&) is used to confirm Boolean statement compound conditionals. A conditional with one or more logical operators used to incorporate several logical expressions can be produced by connecting Boolean expressions with *and, or,* and *not* phrases. The two Boolean values that can be returned are true and false.

An *if* statement causes a program to execute the subsequent block of code, if the Boolean expression returns a value of true. An *else* or *else if* statement is executed if the Boolean expression is *false.* The compiler progresses in its interpretation of the code regardless of whether the Boolean expression is returned *True* or *False*. The text inside a pair of quotation marks is known as a string of characters and will be printed in the terminal window, without the quotes.

The *in* operator can also be utilized to create a Boolean expression. The operator will return a value of True or False. Iterables, a property of collections that are used to provide elements one at a time, in sequence, are constructed of zero or more elements, a node in an HTML document, and encompass strings as proficient as other variable types, such as tuples and lists. A string is an iterable that is composed of elements that are characters.

**Journal**

The terminology included in this assignment leads me to reminisce about the coding class in which I enrolled at Palomar College in the summer following 8th grade. There, I learned the basic coding syntax (e.g., Booleans, *if* statements, classes, etc.) and received a taste of what I might expect to learn in AP Computer Science Principles in high school.

**Conclusion**

1. Describe the relationship between blocks of code indented after the colon in if, elif, and else blocks. The code located directly after an *if* statement is performed if the Boolean or variable expression is returned *True*. *Elif*, or *else if*, statements are located after *if* statements and perform the code located within it, when the first *if* statement is returned a value of false. *Elif* statements also comprise Boolean or variable checks and function only when they are returned true and the first *if* statement is false. *Else* statements usually follow *if* and *elif* statements and act when the *if* and *elif* statements are returned false.

2. There are many operators that can be used to create Boolean expressions. List the ones you have learned about and name one more that you learned about by searching for Boolean operators on the Internet. The Boolean operators of which I am presently aware are: &&, <<, >>, ==, and !=.

3. Steve and Latisha wrote this code:

if check == 2:

print('Code complete.')

else:

print('Code complete.')

print('Not all systems are ready')

Ira, Jayla, and Kendra are all saying it would be better to move lines 22 and 24 to a single line executing print (‘Code complete.’) just before line 21. These three students have different reasons for their opinions. Their reasons are below. Do you think each of them is right, wrong, or somewhere in between? Explain.

Ira: “It would be better to have a single print statement because that code is going to happen no matter what. The program will run slower by having it there twice.”

Jayla: “It would be better to have a single print statement because that code is going to happen no matter what. Later, if you want to change your program, you're going to have to remember to change it in two places the way the code is now.”

Kendra: “It would be better to have a single print statement because it is going to happen no matter what. That program would take up less memory if you just wrote it once.”

The three students concur that removing line 24 and pasting line 22 before line 21 would be “better” than having multiple lines that elucidate on what “is going to happen no matter what.” The students are correct, but they will need to insert a replacement line of code on line 22. For example, the line could be “All systems ready.”

**PLTW Activity 1.3.4**

**Notes**

Glass box testing is the process for evaluating the accuracy or effectiveness of a piece of software while examining its algorithmic structure. A test suite (i.e., a software package designed to evaluate the accuracy or effectiveness of another software solution) is created to run every line of code.

Some programmers write their test suite first, an approach referred to as test driven design, aka Extreme Programming—a software development process in which developers first create a test suite and then create the code to satisfy the test suite. A catch is an exception (or error) caught by the compiler if it is handled instead of being passed to parent programs. If an exception is not caught, the operating system will detect it.

Typecasting is converting data from one type to another. A null string is a string that contains no characters.

**Journal**

Activity 1.3.4 completed at 5:43 PM on November 11, 2017. I was paired with Christian Leon for all the 1.3 activities.

**Conclusion**

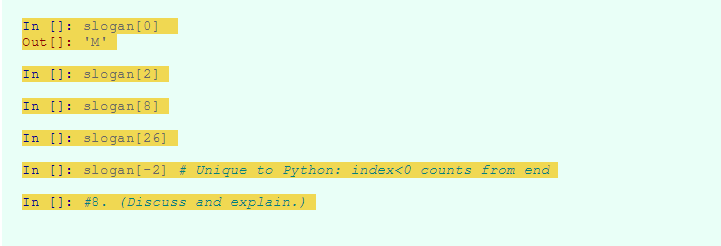
1. What is the relationship between if-structures and glass box testing? *If* statements could be used to follow through with a glass box test (i.e., the process of evaluating the accuracy or effectiveness of a piece of software, while examining its algorithmic structure).
2. Nested if-else structures can contain many blocks of code. How many of those blocks of code might be executed? At least one of those strands of code will be executed.
3. What does a test suite do, and why do you think programmers often write test suites first, before they've even written the functions that will be tested? A test suite (i.e., a software package designed to evaluate the accuracy or effectiveness of another software solution) is created to run every line of code. Extreme programming is intended for more experienced users. By creating the environment in which they will work prior to the actual functions, the users accelerate the entire coding process.

**PLTW Activity 1.3.5**

**Notes**

String literals are enclosed in single or double quotes. Strings are iterables, a property of collections that is used to provide elements one at a time in a sequence and contain a sequence of characters, in sequential order, that begin at 0.

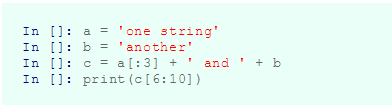
**Journal**



**Conclusion**

1. How many characters are in this sentence? Does it matter whether Python is storing the string as one byte per character or four bytes per character? There are 34 characters in the first sentence. It matters if Python stores the string as one byte per character versus four bytes per character, as one of those options (4 bytes) consumes more space.
2. This question asks you about something you have not learned. In fact, the question is asking about details that go beyond what you will learn in this course. However, wondering what is going on at a lower level of abstraction – and talking about it – can be a useful strategy when learning about computing.

Describe what you think occurs in memory when the following code is executed. The memory usage triples and then multiplies by sixty percent.



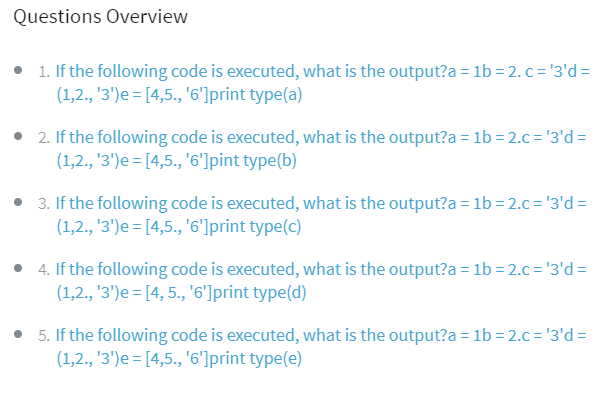
**PLTW Activity 1.3.6**

**Notes**

A string can store letters, numbers, and anything else that can be represented by characters. A tuple is a native data type in Python that can store a collection but cannot assign new values to individual elements. Bound means that the variable name is now listed in the namespace in a table that shows variable names and memory addresses.

Syntax is the precise rules defining how the letters, words, and punctuation of a programming language are required to be together. Tuples are immutable, which means they are not able to be changed after creation. You can only assign a variable that was a tuple to a whole new value, which can be another tuple. Unlike tuples, lists are mutable (i.e., an object that can be changed after its creation; Python includes lists and dictionaries).

**Journal (please see next page)**



**Conclusion**

1. Consider a string, tuple, and list of characters.

In []: a = 'abcde'

In []: b = ('a', 'b', 'c', 'd', 'e')

In []: c = ['a', 'b', 'c', 'd', 'e']

The values of a[3], b[3], and c[3] are all the same. In what ways are a, b, and c different? Aside from the simple naming of the integers, the difference between the three lists is that a has a value of ‘abcde’, b has a value of ('a', 'b', 'c', 'd', 'e'), and c has a value of ['a', 'b', 'c', 'd', 'e'].

1. Why do computer programming languages almost always have a variety of variable types? Why can't everything be represented with an integer? Certain instances (e.g., input from the users, strings, floating points, etc.) necessitate a variable type that is not limited to numbers.

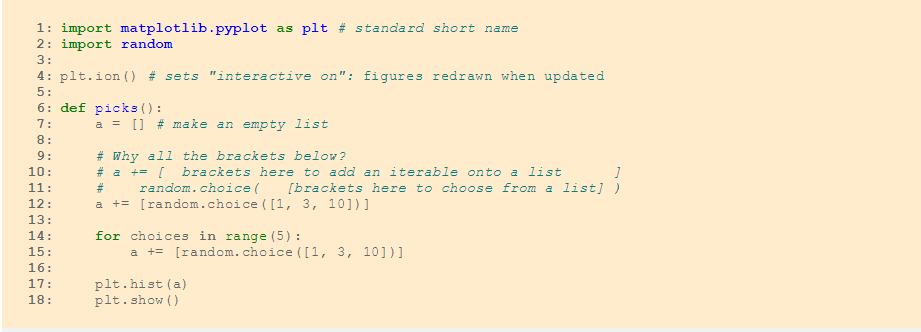
**PLTW Activity 1.3.7**

**Notes**

A loop is an algorithmic structure used in performing instructions multiple times, with each pass through the loop known as an iteration, and is usually controlled by a condition, evaluated with each iteration. You can loop over a block of code once for each item in a list, a tuple, or an iterable (i.e., a property of collections that are used to provide elements one at a time and in sequence) data structure. Python uses indentation to organize blocks of code—a group of program statements that are grouped together.

A function name is the unique identifier given to the function when it is defined and used again whenever the function is called. Inside a range is an argument (i.e., the values that the programmer provides in the function call) separated by commas. The default value of an argument is the value of that argument if the function is referenced without an argument, which is usually 0. A namespace is the set of variable and function names that have been reserved by the compiler/interpreter.

**Journal**



**Conclusion**

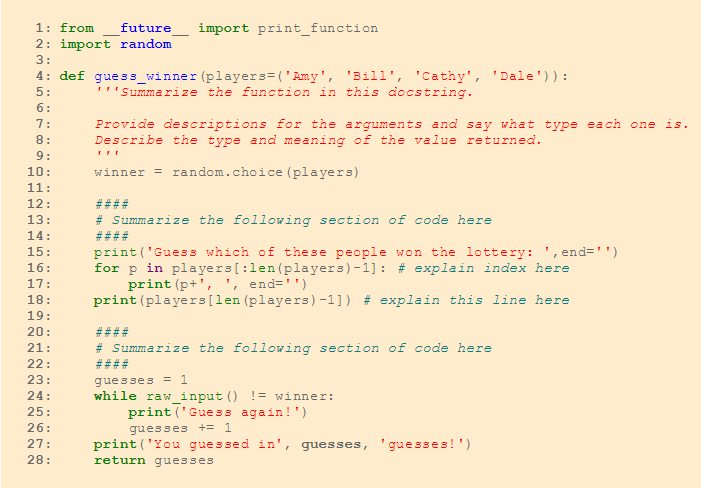
1. Sometimes code using an iterative loop can be written without a loop, simply by repeating the iterated code over and over as separate lines in the program. Explain the disadvantages of developing a program this way. The disadvantages of developing a program with unnecessary liens of code are: more lines and memory usage, less conciseness and preciseness, and additional time consumption.
2. Name a large collection across which you might iterate. An actor or a class with many functions comprise large collections that are iterated.
3. What is the relationship between iteration and the analysis of a large set of data? Iteration is the act of repeating a process to generate an infinite sequence of outcomes or to approach a desired target. Iteration can be used to analyze an extensive set of data.

**PLTW Activity 1.3.8**

**Notes**

A *while* loop is another method to iterate code, as we discovered while using the Scratch™ programming language. Unlike *for* loops, which are used to iterate across a collection of a determined length, a *while* loop is controlled by a conditional expression that might be less predictable. The conditional expression is evaluated once before an iteration through the *while* block and again, before each additional iteration. If the conditional expression is false when evaluated (always at the beginning/end of an iteration), execution jumps to the code after the *while* block. Any Boolean operator, like ==, !=, or <=, can be used in the conditional expression for a *while* loop. One useful Boolean expression for validating input is the *is in* operator.

**Journal**



Mykyta’s favorite loop is a *while* loop.

I designed a second game at Code Day; the source code for both games are embedded within the submission.

**Conclusion**

1. Describe the difference between a while loop and a for loop. While “while” loops and *for* loops both iterate code, a *for* loop requires a certain event to occur and the use of “i” and “j” variables, whereas *while* loops occur when a certain event is occurring. In other words, there does not appear to be much difference other than notation.
2. Reflect on how effectively you and your partner worked together. Individually, describe the strengths of your process and style, and describe what could be improved and how.

Rafe: Christian is amicable, easy to collaborate with, and his interests regarding projects align with my own. The only aspect than could be improved is our ability and experience with coding, as we would like to be more knowledgeable than we currently are.

**Git Repository:** <https://github.com/Battlehawkz/AP-Comp.-Sci.-Principles-Archives-Rafe-Gerson>

**PLTW Activity 1.3.9**

**Notes (work in progress)**

**Journal**

**Conclusion**

**PLTW Activity 1.3.10**

**Notes (work in progress)**

**Journal**

**Conclusion**