# Domain 1
# Lesson 1

Python v2

# Python Introduction

Python is a versatile programming language with a wide variety of uses, including basic apps, web development, data analysis, artificial intelligence, the Internet of Things, automation, and game development. This workbook helps users practice using Python and reinforces the concepts covered in the video portion of this course.

## Purpose

Upon completing this project, you will better understand general facts about Python.

## Steps for Completion

1. Label the following statements as true or false.
    a. ______________ Python code uses curly brackets.
    b. ______________ If Python code is not indented correctly, it will not work.
    c. ______________ Python is an interpreted language, meaning one can write and run code in the Python engine.
    d. ______________Classes cannot be created in Python, but objects can be.
    e. ______________ Python is a dynamically typed language, meaning that data types do not need to be declared on variables before the variables themselves are declared.

### Project Details

**Project file**
N/A

**Estimated completion time**
5 minutes

**Video reference**
**Domain 1**
**Topic**: Identify Data Types
**Subtopic**: How to Study for This Exam; What Is Python?

**Objectives covered**
**1** Operations Using Data Types and Operators
**1.1** Evaluate expressions to identify the data types Python assigns to variables

# Installing Python

Before one can practice using Python, they must have the proper tools installed on their device. This project is essential, as many other projects cannot be completed without Python and Visual Studio Code. The content displayed on the following websites may differ from this course because these tools are updated frequently.

## Purpose

Upon completing this project, you will have installed Python and Visual Studio Code and ensured they work properly.

## Steps for Completion

1. Download the latest version of Python at **python.org**
2. Download Visual Studio Code as your Integrated Development Environment app at **code.visualstudio.com**
    a. You can take the default installation once you have installed the Python engine.
3. Verify that Python is installed by opening a command prompt and running **py**
    a. If there are three arrows and Python information, Python is installed and ready for use.
4. Close the Python session within the command prompt.
5. Open Visual Studio Code.
6. Install the Python extension.
7. Restart Visual Studio Code if necessary.
8. Open the **Domain 1 Student** folder within Visual Studio to easily access files.
    a. You may open all six domain Student folders now, or when you begin projects for each domain.

### Project Details

**Project file**

Domain 1 Student folder

**Estimated completion time**

10-15 minutes

**Video reference**

**Domain 1**
  **Topic**: Identify Data Types
    **Subtopic**: What Needs to Be Installed

**Objectives covered**

**1** Operations Using Data Types and Operators
  **1.1** Evaluate expressions to identify the data types Python assigns to variables

# Domain 1
# Lesson 2

Python v2

# Strings and Integers

Strings and integers are data types that can be added to variables within Python code. A variable is a container that stores a value, like a string of text, a number, or other data types. Strings of text go inside quotation marks and can be used in various ways. An integer is a numeric data type that represents a whole number. Variable names can contain letters, numbers, and underscores, but not symbols or spaces. Python's standard naming convention is all lowercase characters with underscores replacing spaces.

## Purpose

Upon completing this project, you will better understand how to use strings and integers in Visual Studio. The files needed for this project should already be in the program.

## Steps for Completion

1. Open the **111-str.py** file from your Domain 1 Student folder within Visual Studio.
2. Note the two variables with strings containing a first and last name.
3. Add a print statement on line 3 to make the code output a full name.
    a. The order of the names can be last, then first, or first, then last.
        i. If the names are last, then first, they must be separated by a comma.
4. Run the code to ensure it outputs a first and last name in your chosen order.
5. In Python, users can add single or double quotes around text strings but not ________________ quotes.
6. Why will the following print statement cause an error message to appear when it is run?

    print ('Jason's turn')

    a. ________________________________________________________________
    ________________________________________________________________
7. Save the file as **111-str-completed**
8. Open the **112-numbers.py** file.
9. Numbers represented as ________________ cannot be used in calculations.
10. Add a print statement on line 4 that prints the data type of the number_of_tries variable to verify that it is an integer.
11. Run the code to verify that the data type on line 1 is an integer.
12. Save the file as **112-numbers-completed**

### Project Details

**Project file**

111-str.py<br>
112-numbers.py

**Estimated completion time**

10-15 minutes

**Video reference**

**Domain 1**
- **Topic**: Identify Data Types
    - **Subtopic**: str; int

**Objectives covered**

**1** Operations Using Data Types and Operators
- **1.1** Evaluate expressions to identify the data types Python assigns to variables
    - **1.1.1** str
    - **1.1.2** int

# Floats and Bools

Floats and bools are other data types used in Python code. While integers are whole numbers, floating numbers, or shorts, are numbers with decimals. The Boolean data type is either true or false or, in binary terms, 1 or 0.

## Purpose

Upon completing this project, you will better understand how to use floats and bools in Python coding.

## Steps for Completion

1. Open the **113-numbers.py** file from your Domain 1 Student folder.
2. Note that the multiplier variable on line 2 is a float data type.
3. Add a print statement on line 4 that calculates the sum of the number_of_tries and multiplier variables. Run the code.
4. Which data type did the calculation return?
   a. ______________________________
5. Change the operator on line 4 from addition to division. Run the code.
6. Which data type did the calculation return?
   a. ______________________________
7. Save the file as **113-numbers-completed**
8. Python code written for games with scores will likely use ______________, while games involving money will probably need ____________.
9. Open the **114-boolean.py** file.
10. Note the Boolean variable on line 1, char_life, which is set to True. Run the code.
11. Save the file as **114-boolean-completed**
12. Python is case-sensitive, and Boolean variables must be in ______________ casing.

### Project Details

**Project file**

113-numbers.py
114-boolean.py

**Estimated completion time**

10-15 minutes

**Video reference**

**Domain 1**
**Topic**: Identify Data Types
**Subtopic**: float; bool

**Objectives covered**

**1** Operations Using Data Types and Operators
**1.1** Evaluate expressions to identify the data types Python assigns to variables
**1.1.3** float
**1.1.4** bool

# Review 1.1

## Purpose

Upon completing this project, you will better understand data types and be able to identify data types for data better.

## Steps for Completion

1. Open the **114-analyze.py** file from your Domain 1 Student folder.
2. Identify the data types for each of the five print statements:

   a. ______________________________

   b. ______________________________

   c. ______________________________

   d. ______________________________

   e. ______________________________

3. Close the file and return to the videos.

### Project Details

**Project file**

114-analyze.py

**Estimated completion time**

5 minutes

**Video reference**

**Domain 1**

**Topic**: Identify Data Types

**Subtopic**: bool; Review on 1.1

**Objectives covered**

**1** Operations Using Data Types and Operators

**1.1** Evaluate expressions to identify the data types Python assigns to variables

# Domain 1
# Lesson 3

## Python v2

# Data Type Conversion

Programmers must understand how to convert data from one type to another because sometimes data can be input or imported as an incorrect data type. Converting data types is necessary to make data usable in given situations.

Coding information to be aware of includes functions and methods. A function is a procedure that performs an action. Python has a variety of built-in functions, such as int for converting to integers and str for strings. Users can also define custom functions. A method is a specific type of function that belongs to an instance of a class or an object and can access or modify the instance or object to which it belongs.

## Purpose

Upon completing this project, you will have experience converting data types within Python code.

## Steps for Completion

1. Open the **121-conversion.py** file from your Domain 1 Student folder.
2. Note that the variable on line 2 stores a rating someone enters (which should be a whole number). Run the code.
3. Enter a float value between 1 and 5 within the terminal, and rerun the code.
4. What data type is stored in the input variable by default?
    a. ____________________
5. Convert the rating variable on line 3 to an integer. Run the code.
6. Enter a float value between 1 and 5 within the terminal, and rerun the code.
7. Convert the points value on line 4 to a string. Run the code.
8. Enter an integer value between 1 and 5 within the terminal, and rerun the code.
9. Enter a float value between 1 and 5 within the terminal, and rerun the code.
10. Convert the input on line 2 to a float. Run the code.
11. Enter an integer value between 1 and 5 within the terminal, and rerun the code.
12. Save the file as **121-conversion-completed**

### Project Details

**Project file**

121-conversion.py

**Estimated completion time**

15 minutes

**Video reference**

**Domain 1**
- **Topic**: Analyze Data Types and Operators
    - **Subtopic**: Data Type Conversion

**Objectives covered**

**1** Operations Using Data Types and Operators
- **1.2** Perform and analyze data and data type operations
    - **1.2.1** Data type conversion

# Indexing

To best understand indexing, programmers need to understand lists and list management. A list is a set of multiple values assigned to a single variable. Lists have square brackets surrounding the values within the list. Indexing is the positioning of each element within a list. Most programming languages, including Python, are zero-based with lists and their values, meaning that the first value in the list is at index zero, the next is index one, and so forth.

## Purpose

Upon completing this project, you will better understand how to navigate indexed data.

## Steps for Completion

1. Open the **122-indexing.py** file from your Domain 1 Student folder.
2. Note that line 1 lists categories for users to choose and line 2 has a print statement to print the categories variable. Run the code.
3. Copy the code on line 2 and paste it on line 4, editing it so that only the first category, Early Bronze Age, prints. Run the code.
4. Change the category value on line 4 to 2. Run the code.
5. Which item in the list was returned?
   a. ______________________________________________
6. Without counting all the list items, change the category value on line 4 so that the last category in the list returns. Run the code.
7. Save the file as **122-indexing-completed**

### Project Details

**Project file**

122-indexing.py

**Estimated completion time**

10 minutes

**Video reference**

**Domain 1**
**Topic**: Analyze Data Types and Operators
**Subtopic**: Indexing

**Objectives covered**

**1** Operations Using Data Types and Operators
**1.2** Perform and analyze data and data type operations
**1.2.2** Indexing

# Domain 1
# Lesson 4

## Python v2

# Slicing

Whereas indexing extracts a value from a list, slicing extracts characters from a list, word, or phrase. Knowing how to slice a value within Python code can save one from having to extract text elsewhere, saving time and potential errors within an app. Users can extract characters from a variable's beginning, middle, and end.

## Purpose

Upon completing this project, you will better understand how to extract data from a variable.

## Steps for Completion

1. Open the **123-slicing.py** file from your Domain 1 Student folder.
2. Note that line 1 consists of a variable called serial_number, a series of sixteen numbers.
3. Add a print function on line 2 to output the first four characters from the serial_number variable. Run the code.
4. Copy the code on line 2 and paste it on line 4, editing it so that only the last four characters of the variable print. Run the code.
5. Copy the code on line 4 and paste it on line 5, editing it so that the eight middle characters of the variable print. Run the code.
6. Save the file as **123-slicing-completed**

### Project Details

**Project file**

123-slicing.py

**Estimated completion time**

10 minutes

**Video reference**

**Domain 1**

**Topic**: Analyze Data Types and Operators

**Subtopic**: Slicing

**Objectives covered**

**1** Operations Using Data Types and Operators

**1.2** Perform and analyze data and data type operations

**1.2.3** Slicing

# Data Structures

Data structures, or collection data types, consist of lists stored inside variables. Additional data types to understand when creating data structures include dictionaries, sets, and tuples. A dictionary stores data in key-value pairs. A set is a type of collection that stores unique values of data. A tuple is a list with immutable values. Programmers should know what each structure type does and the syntax used to build it. This knowledge can help make apps perform more smoothly.

## Purpose

Upon completing this project, you will better understand data structures.

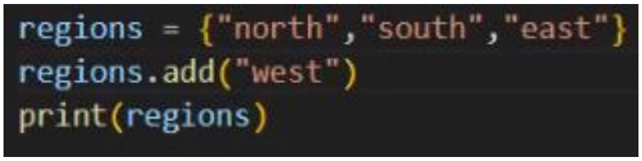
## Steps for Completion

1. Strings are data structures as they are collections of ____________________.

2. Match each code example to its correct data type.

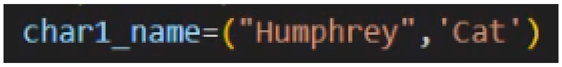
A. Dictionary B. Set C. Tuple

a. ________



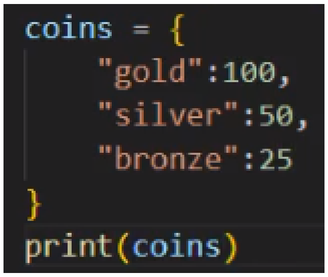
```
regions = {"north","south","east"}
regions.add("west")
print(regions)
```

b. ________



```
char1_name=("Humphrey",'Cat')
```

c. ________



```
coins = {
    "gold":100,
    "silver":50,
    "bronze":25
}
print(coins)
```

### Project Details

**Project file**

N/A

**Estimated completion time**

5 minutes

**Video reference**

**Domain 1**

**Topic**: Analyze Data Types and Operators

**Subtopic**: Construct Data Structures

**Objectives covered**

**1** Operations Using Data Types and Operators

**1.2** Perform and analyze data and data type operations

**1.2.4** Construct data structures

# Domain 1
# Lesson 5

## Python v2

# Lists and Their Operations

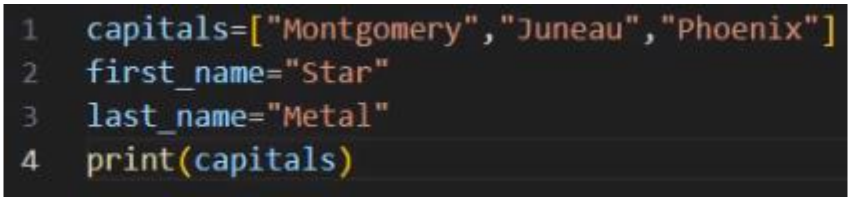
A list is a collection of items in a defined order. A list's primary purpose is to store a list of values in a single variable. Some list operations one can apply to a list include built-in methods. Methods are functions tied to a specific object and invoked using the period (.) notation. Particular methods include append, which adds an item to the end of a list, and insert, which inserts an item at a specified position.

## Purpose

Upon completing this project, you will better understand lists and their operations.

## Steps for Completion

1. When a list of values needs to be adjusted, one does not need to declare a ____________________ for every value.
2. In the following code example, lines 2 and 3 store information in separate variables. Why would one do this instead of using a list?



```
1  capitals=["Montgomery","Juneau","Phoenix"]
2  first_name="Star"
3  last_name="Metal"
4  print(capitals)
```

   a. ____________________________________________________________
   ____________________________________________________________
   ____________________________________________________________

3. Open the **126-list_operations.py** file from your Domain 1 Student folder.
4. Use the append method on line 2 to add Sacramento to the capitals list. Run the code.
5. In programming, a period (.) is the accessor that allows one to use a __________________________ within an object.
6. Answer the questions about the following code:

   capitals.append()

   a. What is the object?
      i. ______________________________________
   b. What is the method?
      i. ______________________________________
7. Use the insert method on line 3 to add Little Rock to the capitals list before Sacramento and rerun the code.
8. Save the file as **126-list_operations-completed**

### Project Details

**Project file**
126-list_operations.py

**Estimated completion time**
10 minutes

**Video reference**
**Domain 1**
**Topic**: Analyze Data Types and Operators
**Subtopic**: Lists; List Operations

**Objectives covered**
**1** Operations Using Data Types and Operators
**1.2** Perform and analyze data and data type operations
**1.2.5** Lists
**1.2.6** List operations

LearnKey

# Review 1.2

## Purpose

Upon completing this project, you will better understand lists as a whole and how changing one bit of code can affect multiple areas of code.

## Steps for Completion

1. Open the **126-list_operations.py** file from your Domain 1 Student folder.
    a. You can use this as a live file to compliment what is in the video reference.
2. Determine why, when the code is run in the video, Little Rock shows twice.
    a. ______________________________
    ______________________________
    ______________________________
3. What are two possible remedies to the problem described in the video?
    a. ______________________________
    ______________________________
    b. ______________________________
    ______________________________
    ______________________________
4. Return to the videos for the answer.
5. Open the **126-analyze.py** file from your Domain 1 Student folder.
6. What will the print statement on line 2 print?
    a. ______________________________
7. What type of data structure is on line 4?
    a. ______________________________
8. How can the coins variable be adjusted to have a bronze value instead of a platinum value?
    a. ______________________________
    ______________________________

### Project Details

**Project file**

126-list_operations.py
126-analyze.py

**Estimated completion time**

10 minutes

**Video reference**

**Domain 1**
**Topic**: Analyze Data Types and Operators
**Subtopic**: List Operations; Review on 1.2 Part 1; Review on 1.2 Part 2

**Objectives covered**

**1** Operations Using Data Types and Operators
**1.2** Perform and analyze data and data type operations

# Domain 1

# Lesson 6

## Python v2

# Assignment Operators

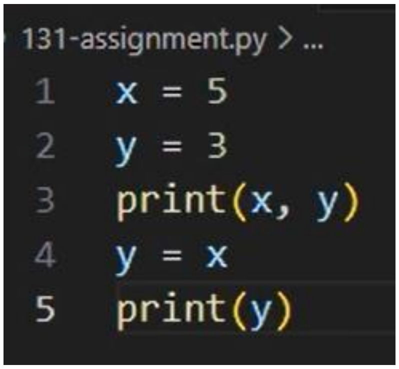
Operators help perform calculations and check for conditions within a block of code. From first to last, the six types of operators that can be applied are arithmetic, containment, comparison, identity, logical, and assignment. Assignments set values to variables.

## Purpose

Upon completing this project, you will better understand assignment operators.

## Steps for Completion

1. ______________________ supersede the six operators in the order of operations.
2. Review the code, then answer the question regarding variable assignments.
    a. What will happen when this code is run?



131-assignment.py > ...

```python
x = 5
y = 3
print(x, y)
y = x
print(y)
```

i. ____________________________________________________________

## Project Details

### Project file

N/A

### Estimated completion time

5 minutes

### Video reference

**Domain 1**
**Topic**: Sequence of Execution
**Subtopic**: Assignment Order

### Objectives covered

**1** Operations Using Data Types and Operators
**1.3** Determine the sequence of execution based on operator precedence
**1.3.1** Assignment

LearnKey

# Comparison Operators

A comparison operator compares two values. The comparisons include equal to, not equal to, less than, greater than, less than or equal to, and greater than or equal to. These comparisons can help users identify whether calculations in code are returning expected results.

## Purpose

Upon completing this project, you will better understand comparison operators.

## Steps for Completion

1. Comparisons return a(n) ____________________________ value.
2. Comparisons happen ___________________________ assignments in a sequence.
3. What is true with an expression if part of a comparison is false?
   a. ________________________________________________________________

      ________________________________________________________________

### Project Details

**Project file**

N/A

**Estimated completion time**

5 minutes

**Video reference**

**Domain 1**
  **Topic**: Sequence of Execution
    **Subtopic**: Comparison Order

**Objectives covered**

**1** Operations Using Data Types and Operators
  **1.3** Determine the sequence of execution based on operator precedence
    **1.3.2** Comparison

# Logical Operators

Logical operators use three keywords in Python: not, and, and or. These keywords have an order of operations that users must understand to implement logical operators and get the intended results.

## Purpose

Upon completing this project, you will better understand logical operators.

## Steps for Completion

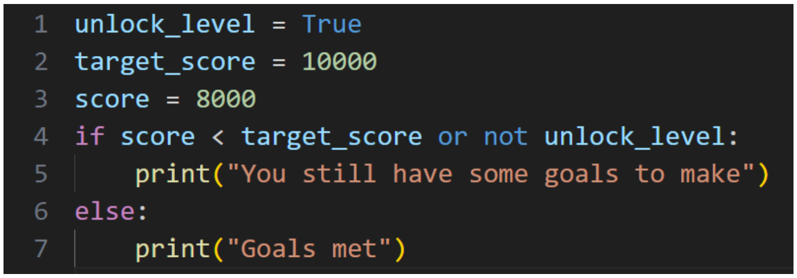
1. List the order of the logical keywords in the order of operations from first to last.

    a. ______________________________

    b. ______________________________

    c. ______________________________

2. Review the code, then answer the question regarding the logical keywords' order of operations.



```
unlock_level = True
target_score = 10000
score = 8000
if score < target_score or not unlock_level:
    print("You still have some goals to make")
else:
    print("Goals met")
```

    a. What will happen when this code is run?

        i. ______________________________

### Project Details

**Project file**

N/A

**Estimated completion time**

5 minutes

**Video reference**

**Domain 1**
**Topic**: Sequence of Execution
**Subtopic**: Logical Order

**Objectives covered**

**1** Operations Using Data Types and Operators
**1.3** Determine the sequence of execution based on operator precedence
**1.3.3** Logical

# Domain 1
# Lesson 7

## Python v2

# Arithmetic

Arithmetic has an order of operations when performing calculations. This order ensures that calculations are performed the same way every time to provide consistent results.

## Purpose

Upon completing this project, you will better understand the order of operations.

## Steps for Completion

1. List the order of operations from first to last.
    a. ______________________________
    b. ______________________________
    c. ______________________________
    d. ______________________________
    e. ______________________________
    f. ______________________________
2. Open the **134-arithmetic.py** file from your Domain 1 Student folder.
3. Run the code. What is the result?
    a. ______________________________
4. Edit the code so that the base and bonus variables are added together before being multiplied by the rank variable.
5. Run the code. What is the result?
    a. ______________________________
6. Save the file as **134-arithmetic-completed**

### Project Details

**Project file**

134-arithmetic.py

**Estimated completion time**

10 minutes

**Video reference**

**Domain 1**
**Topic**: Sequence of Execution
**Subtopic**: Arithmetic Order

**Objectives covered**

**1** Operations Using Data Types and Operators
**1.3** Determine the sequence of execution based on operator precedence
**1.3.4** Arithmetic

# Identity Operator

Two variables can appear equal, but one can use the identity operator to determine if they are truly the same. The identity operator determines if two variables share the same memory location.

## Purpose

Upon completing this project, you will better understand the identity operator.

## Steps for Completion

1. The __________ keyword is used to identify whether two variables share the same memory space.
2. In the order of operations, the identity operation happens before logical and assignment operations but after arithmetic and ______________________operations.
3. If two variables share the same memory space, they will have ________________________ values.

### Project Details

**Project file**
N/A

**Estimated completion time**
5 minutes

**Video reference**
**Domain 1**
**Topic**: Sequence of Execution
**Subtopic**: Identity Order

**Objectives covered**
**1** Operations Using Data Types and Operators
**1.3** Determine the sequence of execution based on operator precedence
**1.3.5** Identity (is)

# Containment Operator

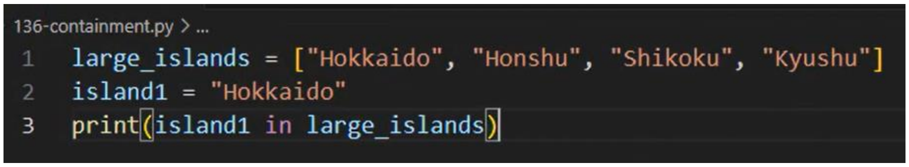
Containment is often used for lists, sets, tuples, and dictionaries. The containment operator checks whether a value is contained within a list of values.

## Purpose

Upon completing this project, you will better understand the containment operator.

## Steps for Completion

1. Containment operations happen ________________ comparison, logical, and identity operations.
2. Review the code, then answer the question about the containment operator.



```
136-containment.py > ...
1  large_islands = ["Hokkaido", "Honshu", "Shikoku", "Kyushu"]
2  island1 = "Hokkaido"
3  print(island1 in large_islands)
```

   a. What result will the code on line 3 output?

      i. ______________________________________________

### Project Details

**Project file**
N/A

**Estimated completion time**
5 minutes

**Video reference**
**Domain 1**
**Topic**: Sequence of Execution
**Subtopic**: Containment Order

**Objectives covered**
**1** Operations Using Data Types and Operators
**1.3** Determine the sequence of execution based on operator precedence
**1.3.6** Containment (in)

# Review 1.3

## Purpose

Upon completing this project, you will better understand the order of operations across categories of operators and within a category of operators.

## Steps for Completion

1. What is the order for the six types of operators?
    a. ______
    b. ______
    c. ______
    d. ______
    e. ______
    f. ______
2. What supersedes all operators in the order of operations?
    a. ______
3. What is the order of preference for logical operators?
    a. ______
    b. ______
    c. ______
4. What is the order of operations for arithmetic operators?
    a. ______
    b. ______
    c. ______
    d. ______
    e. ______
    f. ______

### Project Details

**Project file**
N/A

**Estimated completion time**
15 minutes

**Video reference**
**Domain 1**
**Topic**: Sequence of Execution
**Subtopic**: Containment Order; Review on 1.3

**Objectives covered**
**1** Operations Using Data Types and Operators
**1.3** Determine the sequence of execution based on operator precedence

# Domain 1
# Lesson 8

Python v2

# Using Assignment Operators

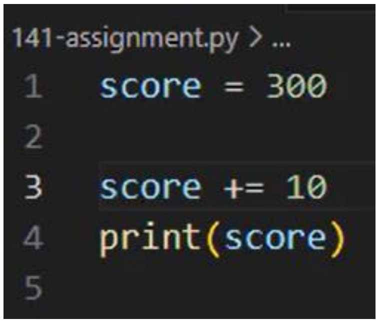
Assignments, which give a value to a variable, can be simple or compound. Many operations can be performed using assignments, including addition, subtraction, multiplication, exponents, and division. The modulus and floor operators are related to the division operator.

## Purpose

Upon completing this project, you will better understand how to use assignments.

## Steps for Completion

1. A(n) ____________________ assignment attaches a value to a variable using an equal sign.
2. The modulus is the ______________________ of two numbers divided by each other.
3. A(n) ____________________ assignment changes the value of a variable using an arithmetic operator.
4. Floor division returns the ____________________ of two numbers divided by each other.
5. Review the code, then answer the question about the assignment operator in the code.
   a. What result will the code on lines 3 and 4 produce?



141-assignment.py > ...

```
1   score = 300
2
3   score += 10
4   print(score)
5
```

      i. ____________________________________

6. What symbols are used to determine the modulus of two numbers?
   a. ____________________________________________
7. What symbols are used for floor division?
   a. ____________________________________________
8. What symbols are used to calculate an exponent to a number?
   a. ____________________________________________

### Project Details

**Project file**

N/A

**Estimated completion time**

10 minutes

**Video reference**

**Domain 1**
**Topic**: Select Operators
**Subtopic**: Assignment

**Objectives covered**

**1** Operations Using Data Types and Operators
**1.4** Select operators to achieve the intended results
**1.4.1** Assignment

# Using Comparison Operators

Comparison operators are used to evaluate the similarities between two variables. Understanding how to use the different comparison operators available helps ensure that one can understand how two variables relate to one another.

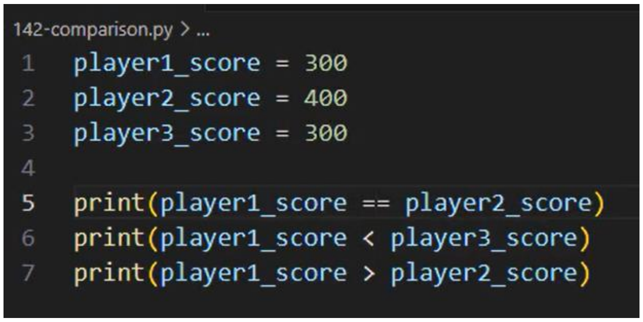
## Purpose

Upon completing this project, you will better understand how to use comparison operators.

## Steps for Completion

1. Review the code, then answer the questions regarding the comparison operators in the code.

   a. What result will the code on line 5 output?



```
142-comparison.py > ...
1  player1_score = 300
2  player2_score = 400
3  player3_score = 300
4
5  print(player1_score == player2_score)
6  print(player1_score < player3_score)
7  print(player1_score > player2_score)
```

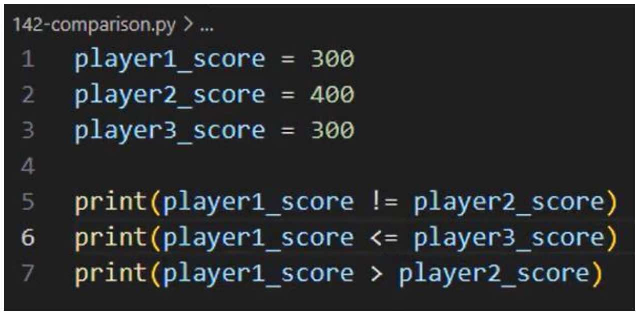
      i. ______________________________

   b. What result will the code on line 6 output?

      i. ______________________________

   c. What result will the code on line 7 output?

      i. ______________________________



```
142-comparison.py > ...
1  player1_score = 300
2  player2_score = 400
3  player3_score = 300
4
5  print(player1_score != player2_score)
6  print(player1_score <= player3_score)
7  print(player1_score > player2_score)
```

   d. What result will the code on line 5 output?

      i. ______________________________

   e. What result will the code on line 6 output?

      i. ______________________________

### Project Details

**Project file**

N/A

**Estimated completion time**

5 minutes

**Video reference**

**Domain 1**
**Topic**: Select Operators
**Subtopic**: Comparison

**Objectives covered**

**1** Operations Using Data Types and Operators
**1.4** Select operators to achieve the intended results
**1.4.2** Comparison

LearnKey

# Using Logical Operators

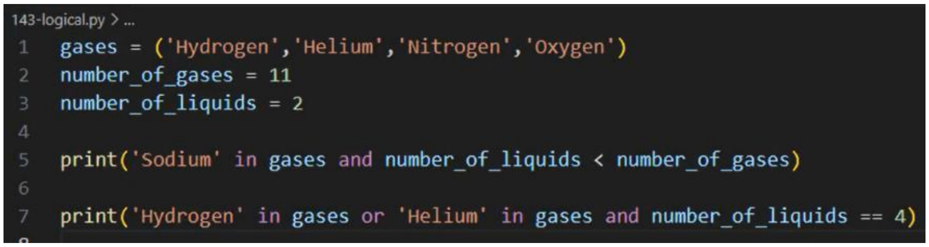
Logical operators involve using the keywords and, or, and not to compare two or more conditions. These keywords have their own order of operations. Developers must understand the order of operations for logical operators to know why statements containing logical operators produce the results they output.

## Purpose

Upon completing this project, you will better understand how to use logical operators.

## Steps for Completion

1. Review the code, then answer the questions regarding the logical operators in the code.



143-logical.py > ...

```
1  gases = ('Hydrogen','Helium','Nitrogen','Oxygen')
2  number_of_gases = 11
3  number_of_liquids = 2
4
5  print('Sodium' in gases and number_of_liquids < number_of_gases)
6
7  print('Hydrogen' in gases or 'Helium' in gases and number_of_liquids == 4)
```

   a. What result will the code on line 5 output? Why?

      i. ______________________________________________

      ______________________________________________

      ______________________________________________

   b. What result will the code on line 7 output? Why?

      i. ______________________________________________

      ______________________________________________

      ______________________________________________

### Project Details

**Project file**

N/A

**Estimated completion time**

5 minutes

**Video reference**

**Domain 1**

**Topic**: Select Operators

**Subtopic**: Logical

**Objectives covered**

**1** Operations Using Data Types and Operators

**1.4** Select operators to achieve the intended results

**1.4.3** Logical

# Domain 1

# Lesson 9

Python v2

# Using Arithmetic Operators

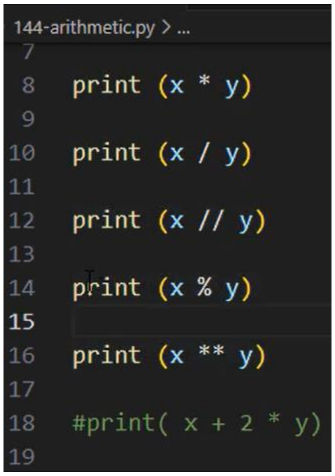
Arithmetic operators are an important part of performing calculations. Knowing what operators are available and the order in which they are performed can help developers write code efficiently and produce intended results, especially when testing code.

## Purpose

Upon completing this project, you will better understand how to use arithmetic operators.

## Steps for Completion

1. The quotient of two integers can be an integer or a(n) ______________.
2. A(n) ______________ operator makes a positive value negative and a negative value positive.
3. Review the code, then answer the questions regarding the arithmetic operators in the code.



```
144-arithmetic.py > ...
 7
 8   print (x * y)
 9
10   print (x / y)
11
12   print (x // y)
13
14   print (x % y)
15
16   print (x ** y)
17
18   #print( x + 2 * y)
19
```

   a. What result will the code on line 12 output?

      i. ______________________________

   b. What result will the code on line 14 output?

      i. ______________________________

   c. What result will the code on line 16 output?

      i. ______________________________

   d. What result will the code on line 18 output?

      i. ______________________________

### Project Details

**Project file**
N/A

**Estimated completion time**
10 minutes

**Video reference**
**Domain 1**
**Topic**: Select Operators
**Subtopic**: Arithmetic

**Objectives covered**
**1** Operations Using Data Types and Operators
**1.4** Select operators to achieve the intended results
**1.4.4** Arithmetic

# Using the Identity Operator

The identity operator, is, is used to see if two variables are taking up the same memory space. This operator is especially useful when speed is an important factor.

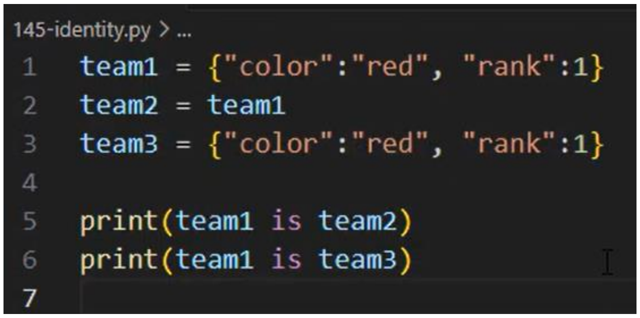
## Purpose

Upon completing this project, you will better understand how to use the identity operator.

## Steps for Completion

1. Review the code, then answer the questions about the identity operators in the code.

    a. Why will the code on line 5 return True?



```
1  team1 = {"color":"red", "rank":1}
2  team2 = team1
3  team3 = {"color":"red", "rank":1}
4
5  print(team1 is team2)
6  print(team1 is team3)
7
```

i. ______________________________
______________________________
______________________________

    b. Why will the code on line 6 return False?

i. ______________________________
______________________________
______________________________

### Project Details

**Project file**
N/A

**Estimated completion time**
5 minutes

**Video reference**
**Domain 1**
**Topic**: Select Operators
**Subtopic**: Identity

**Objectives covered**
**1** Operations Using Data Types and Operators
**1.4** Select operators to achieve the intended results
**1.4.5** Identity (is)

# Using the Containment Operator

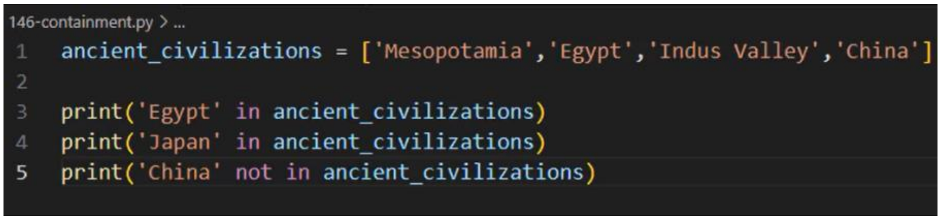
Containment uses the in keyword to determine whether a value is in a list of values. The containment operator can be especially useful when a variable in code contains a long list of values.

## Purpose

Upon completing this project, you will better understand how to use the containment operator.

## Steps for Completion

1. Review the code, then answer the questions regarding the containment operators in the code.



```
146-containment.py > ...
1  ancient_civilizations = ['Mesopotamia','Egypt','Indus Valley','China']
2
3  print('Egypt' in ancient_civilizations)
4  print('Japan' in ancient_civilizations)
5  print('China' not in ancient_civilizations)
```

   a. What result will the code on line 3 output?

      i. ______________________________

   b. What result will the code on line 4 output?

      i. ______________________________

   c. What result will the code on line 5 output?

      i. ______________________________

### Project Details

**Project file**

N/A

**Estimated completion time**

5 minutes

**Video reference**

**Domain 1**

- **Topic**: Select Operators
  - **Subtopic**: Containment

**Objectives covered**

**1** Operations Using Data Types and Operators

- **1.4** Select operators to achieve the intended results
  - **1.4.6** Containment (in)

# Review 1.4

## Purpose

Upon completing this project, you will better understand how assignment operators work in Python.

## Steps for Completion

1. Open the **146-analyze.py** file from your Domain 1 Student folder.
2. Answer the questions the code has on the following lines:
    a. Line 4: ________________________________
    b. Line 6: ________________________________
    c. Line 9: ________________________________
    d. Line 11: ________________________________

### Project Details

**Project file**
146-analyze.py

**Estimated completion time**
5 minutes

**Video reference**
**Domain 1**
**Topic**: Select Operators
**Subtopic**: Containment; Review on 1.4

**Objectives covered**
**1** Operations Using Data Types and Operators
**1.4** Select operators to achieve the intended results