

# HackerFrogs Afterschool

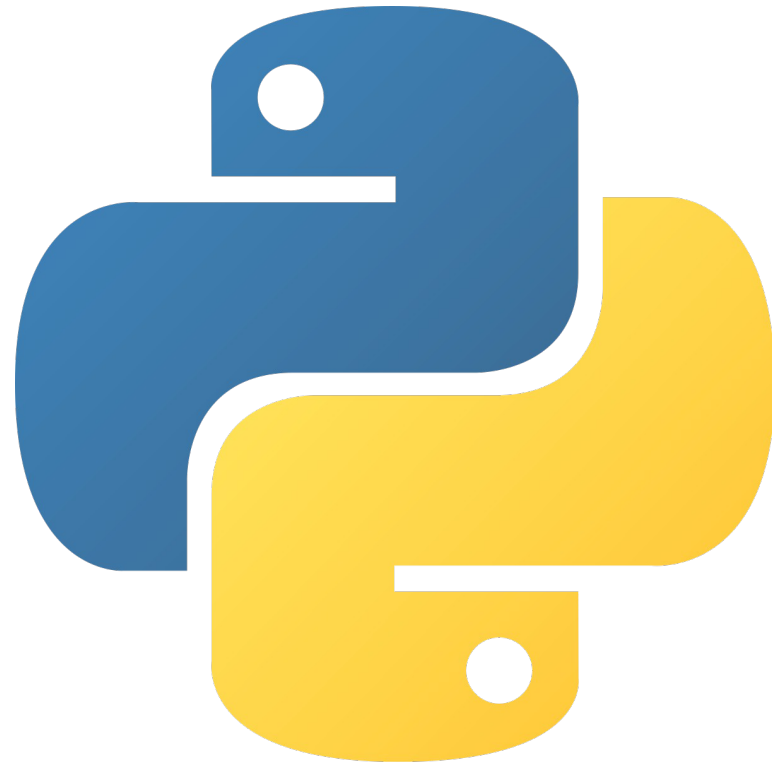
## Python Programming Basics: Part 1

Class:  
Programming (Python)

Workshop Number:  
AS-PRO-PY-01

Document Version:  
1.0

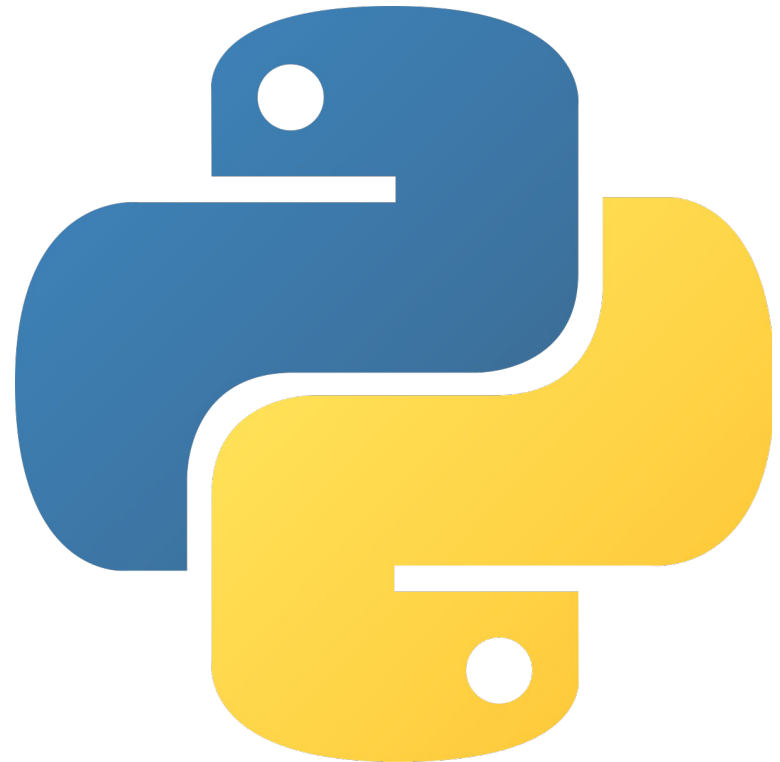
Special Requirements:  
None



# Python Programming Intro

This is the first  
workshop to intro  
Python programming.

Let us begin!



# Part 1: Hello, World!



Programmers learning a language usually write a “Hello, World!” program as their first exercise.

# Hello, World!



Hello, World\_

To complete our **Hello, World!** exercise, we'll go over the Python print function first.

# The Print Function

```
>>> print("This is how you print text in Python!")  
This is how you print text in Python!
```

Virtually all programming languages have some method of printing text to the screen (often referred to as the console). In Python, the **print** function is the primary method of doing so.

# The Print Function

```
>>> print("This is how you print text in Python!")  
This is how you print text in Python!
```

To use the **print** function, we use the function name, **print**, then a pair of parentheses, then encase what we want printed, in either single or double quotes, between the parentheses.

# The Print Function

```
>>> variable = "A variable could be a lot of different things!"  
>>> print(variable)  
A variable could be a lot of different things!
```

Another common use of the print function is to print the values of variables to the console. In this case, we simply input the name of the variable we want printed between the parentheses.

# Let's Learn with Learnpython.org

For a more hand-on and interactive session, we'll be using the Learnpython.org website. Please navigate to the following URL:

[https://learnpython.org/en/Hello,\\_World!](https://learnpython.org/en/Hello,_World!)



# Part 1 – Hello World Quiz

If we write a print function without parentheses, what will happen?

- A) You should only use double quotes ( " )
- B) You should only use single quotes ( ' )
- C) You can use either single or double quotes ( " , ' )
- D) Neither, you should use backticks ( ` )

# Part 1 – Hello World Quiz

If we write a print function without parentheses, what will happen?

- A) You should only use double quotes ( " )
- B) You should only use single quotes ( ' )
- C) You can use either single or double quotes ( " , ' )
- D) Neither, you should use backticks ( ` )

# Part 2: Variables

Virtually all programming languages use variables, which are storage locations associated with a specific name, as well as a value. Take the following as an example:

```
my_name = 'shyhat'
```

# Variables

Wherever we refer to **my\_name** in the code, **shyhat** will be inserted there. So, if we have the following code:

```
my_name = 'shyhat'  
print(my_name)
```

The output should be **shyhat**.

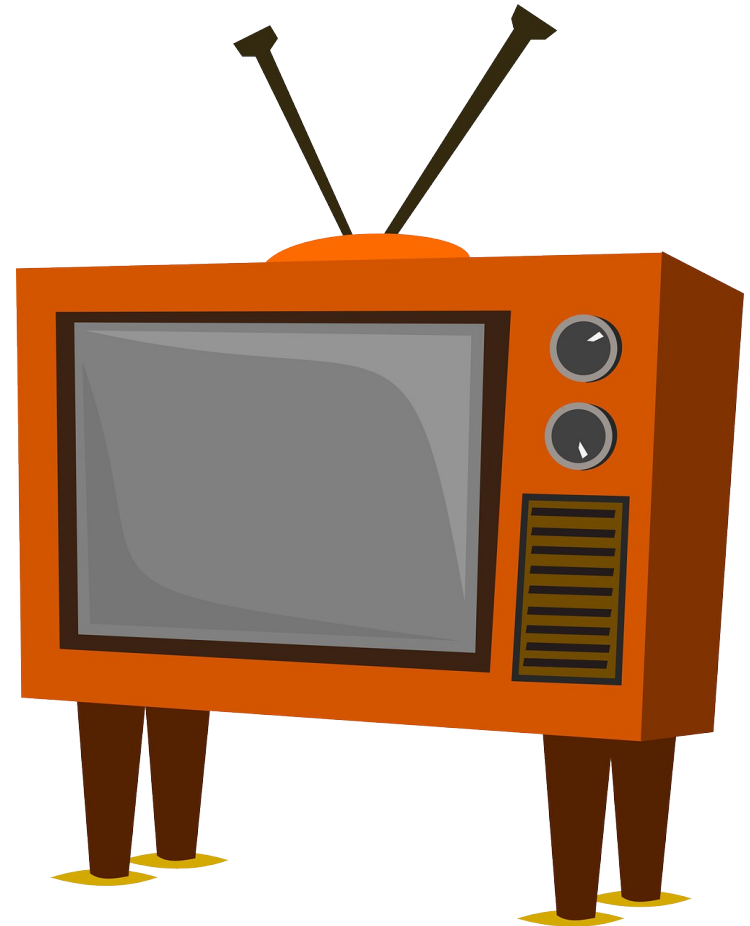
# Variables

The real benefit of using variables in programming is that a variables can be referenced multiple times in the same code, saving a lot of time and accounting while writing code.

In addition, variables can change value after they've been defined.

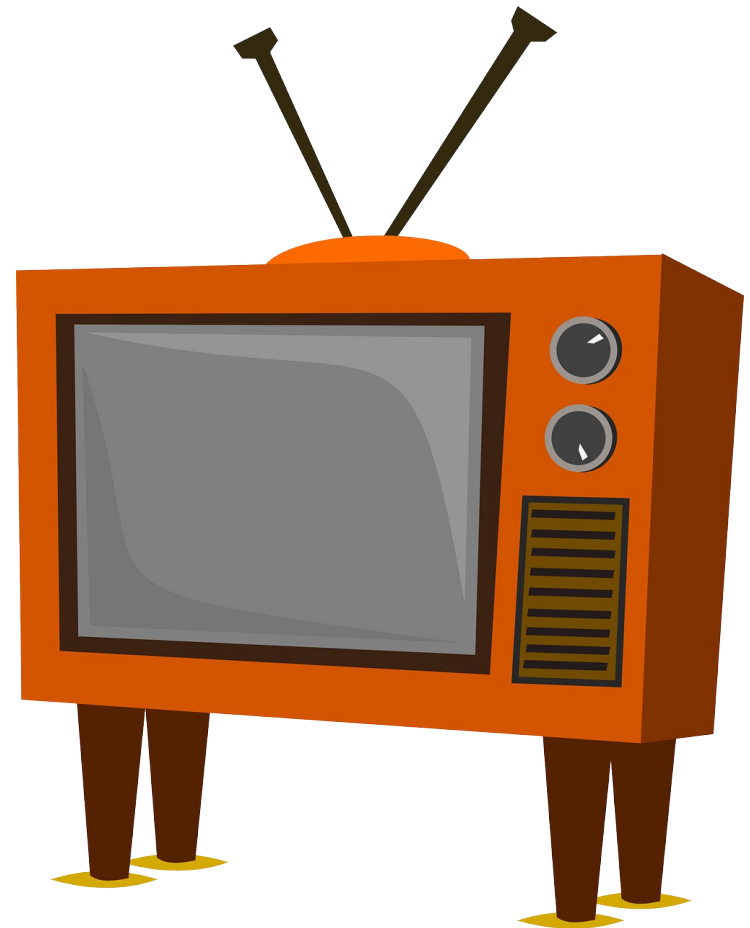
# Variables

For example, say we have an online store program where we sell televisions.



# Variables

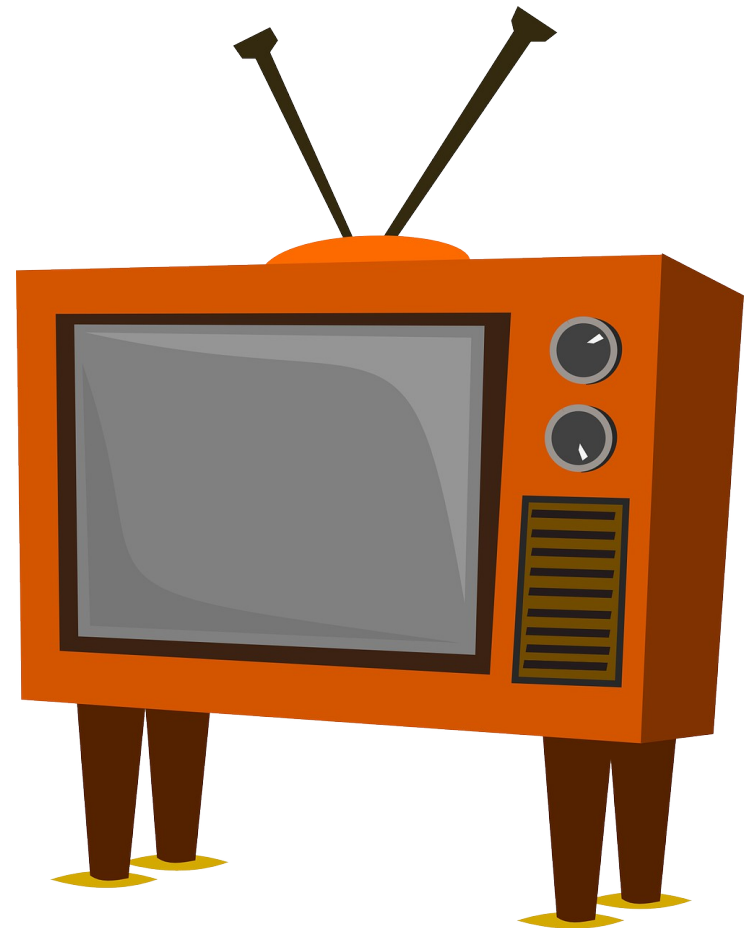
The program has two variables, **tv\_price**, which is the price of a television, and **current\_total**, which keeps track of the total price of the user's shopping cart.



# Variables

The two variables are set to 199.99 and 0, respectively. However, if a TV is purchased, then the **total\_price** variable increases by a value equal to the **tv\_price** variable.

See the following:





# Variables

```
tv_price = 199.99  
current_total = 0
```

```
current_total = current_total + tv_price
```

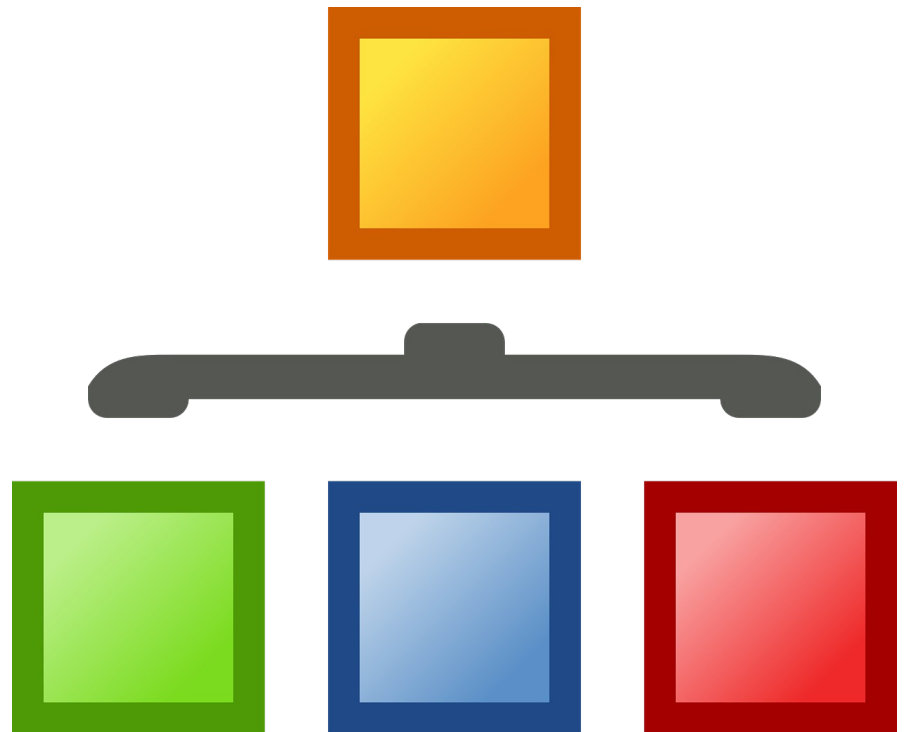
```
print(current_total)
```

The output should be **199.99**

# Data Types

Data types are attributes of data which tell the compiler or interpreter how to use the data.

The data types covered in this workshop are as follows:



# Data Types

## **Strings (str)**

Non-numerical characters, or a combination of numerical and non-numerical characters. E.g, “Mark Twain”, “P@ss\Word123”, “carrot”

## **Integers (int)**

Whole numbers. E.g., 1, 255, 1337

## **Floating-Point Numbers (float)**

Numbers that are accurate to several decimal places. E.g., 0.5 or 12.758

# Data Types



The reason why data types are important is because computers do not inherently know how to interpret data.

# Data Types



For example, if we have a program which calculates the average of school grades (adds all grades together, then divides the sum by the number of assignments) --

# Data Types



the program wouldn't work if it was given words instead of numbers for the grades.

# Data Types

Error: Grades cannot contain non-numeric input.

Because there are data types, we can have the program check that all of the grades are numbers, and if there is a non-number in the input, we can inform the user that the program only accepts numbers for input, and prompt the user to input another value.

# Variables and Types Exercise

Let's cover the next few exercises on the  
Learnpython.org website:

[https://www.learnpython.org/en/Variables\\_and\\_Types](https://www.learnpython.org/en/Variables_and_Types)



# Part 2 – Variables and Types Quiz

Which of the following about naming variables is true?

- A) Python keywords (e.g., class, def, True, False, None) should never be used as variable names
- B) Use single-letter variable names (x, y, a, b) as much as possible
- C) Variables that contain number should be named in UPPERCASE letters, and variables with text should be named in lowercase letters.
- D) None of the above

# Part 2 – Variables and Types Quiz

Which of the following about naming variables is true?

- A) Python keywords (e.g., class, def, True, False, None) should never be used as variable names
- B) Use single-letter variable names (x, y, a, b) as much as possible
- C) Variables that contain number should be named in UPPERCASE letters, and variables with text should be named in lowercase letters.
- D) None of the above

# Part 2 – Variables and Types Quiz

Which of the following is not a Python data type?

- A) Strings – letters or combinations of letters and number that are encased in single or double quotes
- B) Integers – whole numbers, such as 1, 1337, and 256. If put between quotes, they become strings.
- C) Floats (floating point numbers) – numbers with decimal points, such as 0.0, 5.285, and 3.16.
- D) HackerFrogs – folks who like hacking and live streams

# Part 2 – Variables and Types Quiz

Which of the following is not a Python data type?

- A) Strings – letters or combinations of letters and number that are encased in single or double quotes
- B) Integers – whole numbers, such as 1, 1337, and 256. If put between quotes, they become strings.
- C) Floats (floating point numbers) – numbers with decimal points, such as 0.0, 5.285, and 3.16.
- D) HackerFrogs – folks who like hacking and live streams

# Part 3: Lists

Lists are part of what Python categorizes as collections, variables that hold one or more values, which can be strings, numbers, or a mixture of different data types.



# Lists

An example list in Python could look like the following:



# Lists

```
vegetables = ["carrots", "lettuce", "beets"]
```

or

```
related_to_pie = ["round", 3.14, "delicious"]
```

As we can see by the example, a list is defined as a number of objects (separated by commas ', ') between a pair of square brackets '[' ].

# List Methods

```
>>> vegetables = ["carrots", "lettuce", "beets"]  
>>> vegetables.append("radishes")  
>>> print(vegetables)  
['carrots', 'lettuce', 'beets', 'radishes']
```

Methods are actions that can be performed with objects.



# List Methods

```
>>> vegetables = ["carrots", "lettuce", "beets"]  
>>> vegetables.append("radishes")  
>>> print(vegetables)  
['carrots', 'lettuce', 'beets', 'radishes']
```

Methods are performed by indicating the name of the object (the list), then a dot ( . ), then the name of the method, followed by a pair of parentheses, which contain any arguments required for that method.

# List Methods

```
>>> vegetables = ["carrots", "lettuce", "beets"]  
>>> vegetables.append("radishes")  
>>> print(vegetables)  
['carrots', 'lettuce', 'beets', 'radishes']
```

The list method we will learn in this workshop is the **append** method. The append method will add an item to the end of the specified list.

Take the following code as an example:

# List Methods

```
friends = ['Alex', 'Barb']
```

```
friends.append('Carl')
```

```
print(friends)
```

The output should be:

```
['Alex', 'Barb', 'Carl']
```

# List Indexing

List indexing allows for the selection of specific items in a list by referring to them by the position in which they appear in the list.



# List Indexing

To use list indexing, use the name of the list, directly followed by a pair of square brackets ' [ ] ', with the position of the desired list item inside the brackets.

For example:



# List Indexing

```
trees = ['pine', 'elm', 'cedar']  
  
print(trees[0])
```

The output should be **pine**, since pine is the first item on the list. Note that counting positions in indexing start from zero, not one. In this example, `trees[2]` refers to **cedar**, since cedar is the number 2 item on the list if we count from zero.

# Lists Exercise

Let's cover the next few exercises on the  
Learnpython.org website:

<https://learnpython.org/en/Lists>

# Part 3 – Lists Quiz

Which of the following statements about Python lists is false?

- A) When printing a list, all items in the list must be printed
- B) It's not possible to add items to a list after list creation
- C) Counting items in a list starts from zero, not one
- D) Items in a list cannot be changed after they've been added



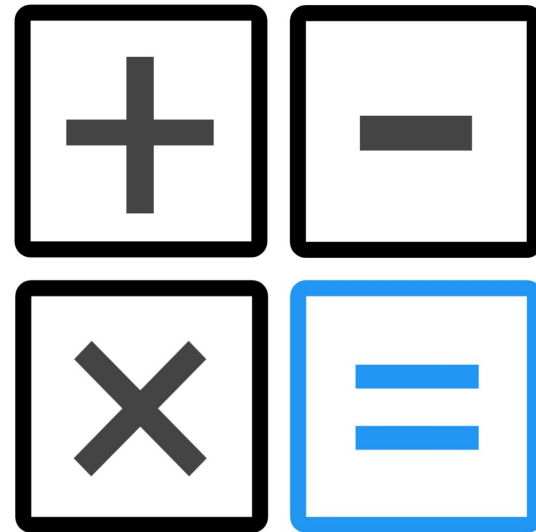
# Part 3 – Lists Quiz

Which of the following statements about Python lists is false?

- A) When printing a list, all items in the list must be printed
- B) It's not possible to add items to a list after list creation
- C) Counting items in a list starts from zero, not one
- D) Items in a list cannot be changed after they've been added

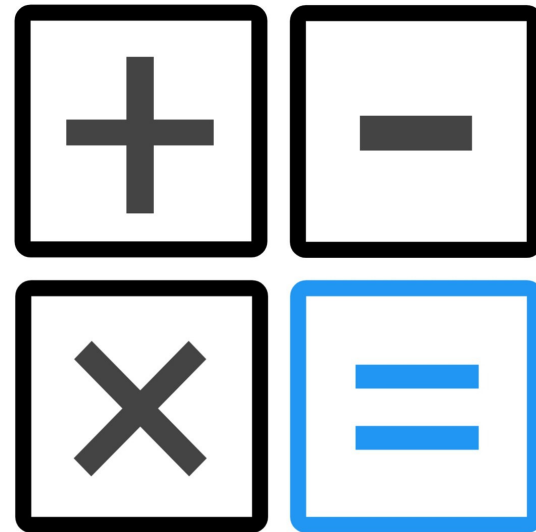
# Part 4: Basic Operators

In programming terms, operators are used to perform an operation on a piece of data.



# Basic Operators

In this lesson, we will learn about basic arithmetic operators that are used for mathematical addition, subtraction, division, multiplication, and a couple of others.



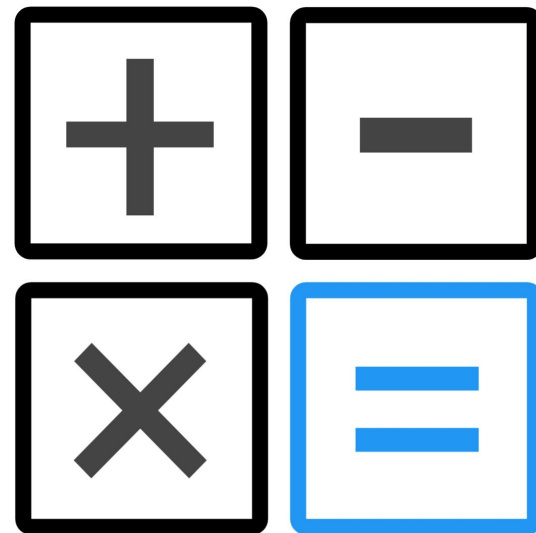
# Basic Operators

Python uses the following operators (symbols) for basic arithmetic:

Operator Name	Symbol	Example
Addition	+	$x + y$
Subtraction	-	$x - y$
Multiplication	*	$x * y$
Division	/	$x / y$
Exponentiation	**	$x ** y$

# Basic Operators

The arithmetic operations that occur in Python follow the PEMDAS order of operations.



# Basic Operators

(P)arentheses  
(E)xponents  
(M)ultiplication  
(D)ivision  
(A)ddition  
(S)ubtraction

# Basic Operators

(P)arentheses

(E)xponents

(M)ultiplication

(D)ivision

(A)ddition

(S)ubtraction

# Basic Operators

(P)arentheses

(E)xponents

(M)ultiplication

(D)ivision

(A)ddition

(S)ubtraction



# Basic Operators

(P)arentheses

(E)xponents

(M)ultiplication

(D)ivision

(A)ddition

(S)ubtraction

# Basic Operators

(P)arentheses  
(E)xponents  
(M)ultiplication  
(D)ivision  
(A)ddition  
(S)ubtraction

# Basic Operators

(P)arentheses  
(E)xponents  
(M)ultiplication  
(D)ivision  
(A)ddition  
(S)ubtraction

# Basic Operators

(P)arentheses  
(E)xponents  
(M)ultiplication  
(D)ivision  
(A)ddition  
(S)ubtraction

# Basic Operators

In summary, all operations enclosed in (p)arentheses are performed first, then all (e)xponent operations are resolved, followed by (m)ultiplication, (d)ivision, (a)ddition, then (s)ubtraction operations at the end.

# Basic Operators

Given the following equation in Python syntax:

`( 4 / 2 ) * 2 ** 2 / 4 + 2 - 4`

**First** the division operation inside of the parentheses (4 divided by 2) is performed.

**Second**, the exponent operation (`2 ** 2`, which is Python's syntax for “two to the second power”, is resolved. **Third**, the multiplication (2 times 4) is resolved.

# Basic Operators

Given the following equation in Python syntax:

$$8 / 4 + 2 - 4$$

**Forth**, the division (8 divided by 4) is resolved.

**Fifth**, the addition (2 + 2) is resolved. **Finally**, the subtraction (4 - 4) is resolved, which sets our final result to 0 (zero).

# Basic Operators

A common operator used in programming is the **modulo** operator ( % )

$$5 \% 2 = 1$$

$$8 \% 5 = 3$$

A modulo operation returns the remainder after one number is divided by another. The resulting number after a modulo operation is called the **modulus**



# Basic Operators

$$n \% 2$$

A typical use for the modulo operation is to determine whether a number is even or odd by performing a modulo operation on it with the number 2. If the result is 1, the number is odd, and if the result is 0, the number is even

# Basic Operators Exercise

Let's cover the next few exercises on the  
Learnpython.org website:

[https://learnpython.org/en/Basic\\_Operators](https://learnpython.org/en/Basic_Operators)

# Part 4 – Basic Operators Quiz

Which of the following methods will allow us to print a string and an integer with the same print function?

- A) No special method need be taken to mix a string and an integer with the print function
- B) The integer must be converted to a string to mix integers and strings in a print function
- C) The string must be converted to an integer to mix integers and strings in a print function
- D) You cannot print a string and integer with print

# Part 4 – Basic Operators Quiz

Which of the following methods will allow us to print a string and an integer with the same print function?

A) No special method need be taken to mix a string and an integer with the print function

B) The integer must be converted to a string to mix integers and strings in a print function

C) The string must be converted to an integer to mix integers and strings in a print function

D) You cannot print a string and integer with print

# Workshop Review Exercise

Before finishing today's workshop, let's write a program that uses some of the concepts we learned:

We'll use one of the Python windows on the [learnpython.org](http://learnpython.org) website to write the following program

# Workshop Review Exercise

A sandwich restaurant owner wants a program that calculates and prints out the total price of customer orders.

The menu only consists of two items: sandwiches that cost 3.50 and soft drink cans, which cost 0.99.

# Workshop Review Exercise

The program should print out two totals, corresponding with the following customer orders:

Order One: 3 sandwiches and 3 soft drinks

Order Two: 7 sandwiches and 6 soft drinks

The program should contain at least two variables, with values equal to the prices of the two menu items.

# Workshop Review Exercise

We'll go over a possible version of the program after this, but please write your own attempt at the program now.

Order One: 3 sandwiches and 3 soft drinks

Order Two: 7 sandwiches and 6 soft drinks

sandwiches: 3.50

soft drinks: 0.99.



# Workshop Review Exercise

The following program fulfills all of the requirements stated previously:

```
sandwich_price = 3.50
soft_drink_price = 0.99

order_one = (sandwich_price * 3) + (soft_drink_price * 3)
order_two = (sandwich_price * 7) + (soft_drink_price * 6)


print(order_one)
print(order_two)
```

# Summary



Let's review the programming concepts we learned in this workshop:

# Hello, World!



Hello, World\_

We wrote the **Hello, World!** program in Python, which is the first program students write for any computer language.

# The Print Function

```
>>> print("This is how you print text in Python!")  
This is how you print text in Python!
```

We learned how to use the Python print function, which prints information to the console and is common to almost all programs written in Python.

# Variables

```
>>> name = "HackerFrogs"  
>>> greetings = "Hello, "  
>>> print(greetings + name)  
Hello, HackerFrogs
```

We learned about programming variables, named storage locations which hold specific values, which can be referenced multiple times in a program.

# Data Types

```
>>> pi = 3.14
>>> number = 1337
>>> name = "shyhat"
>>> print(type(pi),type(number),type(name))
(<class 'float'>, <class 'int'>, <class 'str'>)
```

We learned about Data Types, including **strings** (non-numeric text), **integers** (whole numbers), and **floats** (numbers which include decimal places).

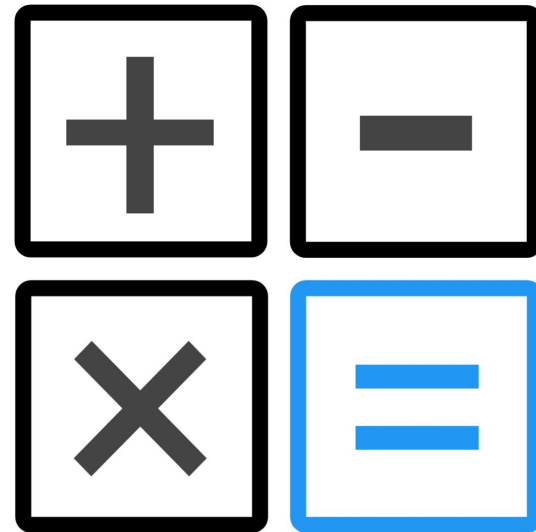
# Lists

```
>>> vegetables = ["carrots", "lettuce", "beets"]  
>>> print(vegetables[0])  
carrots
```

We learned about lists, which are variables with one or more items associated with them. Items in the list can be referenced by a process called **list indexing**.

# Basic Operators

The last thing we learned in this workshop is Python basic operators, which are used to perform arithmetic operations.





# What's Next?

In the next HackerFrogs Afterschool programming workshop, we'll continue learning Python with the [learnpython.org](https://learnpython.org) website.



# Next Workshop's Topics

- String Formatting
- Basic String Operations
- Conditions

# Next Workshop's Topics

- String Formatting
- Basic String Operations
  - Conditions

# Next Workshop's Topics

- String Formatting
- Basic String Operations
- Conditions

# Next Workshop's Topics

- String Formatting
- Basic String Operations
- Conditions

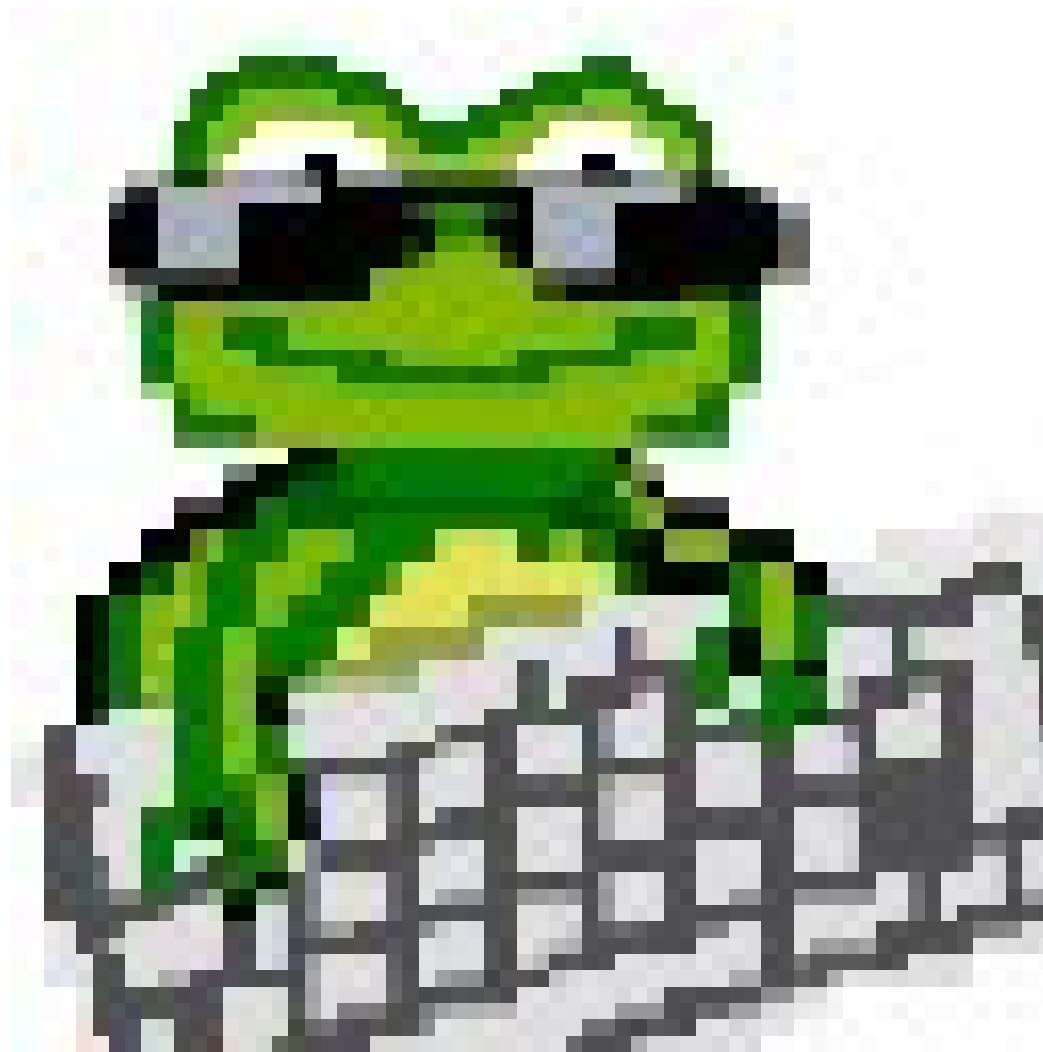
# Extra Credit

Looking for more study material on this workshop's topics?

See this video's description for links to supplemental documents and exercises!



# Until Next Time, HackerFrogs!



# Intermission





# Intermission



# Intermission



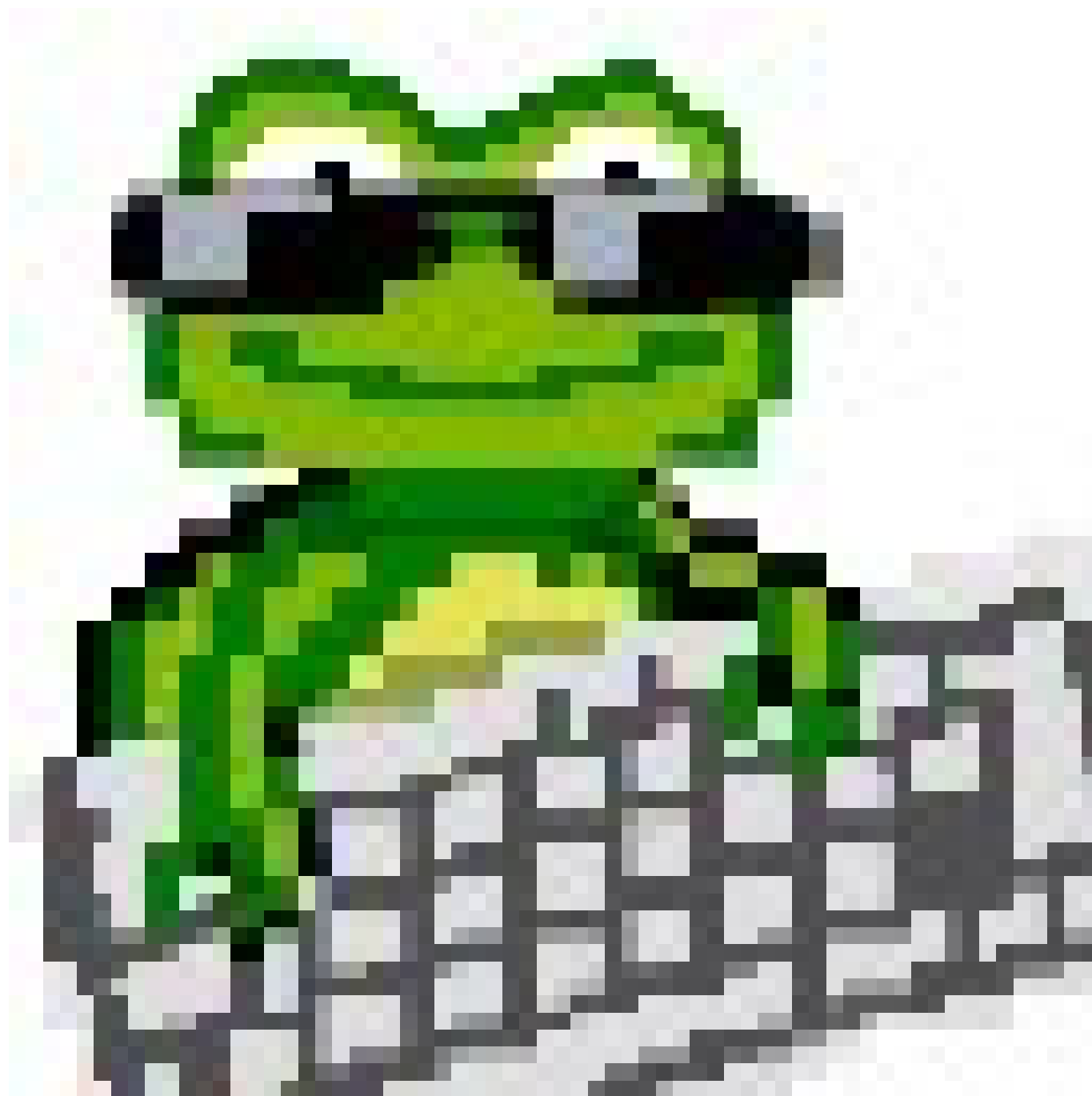
# Intermission

SUBSCRIBE



# Intermission





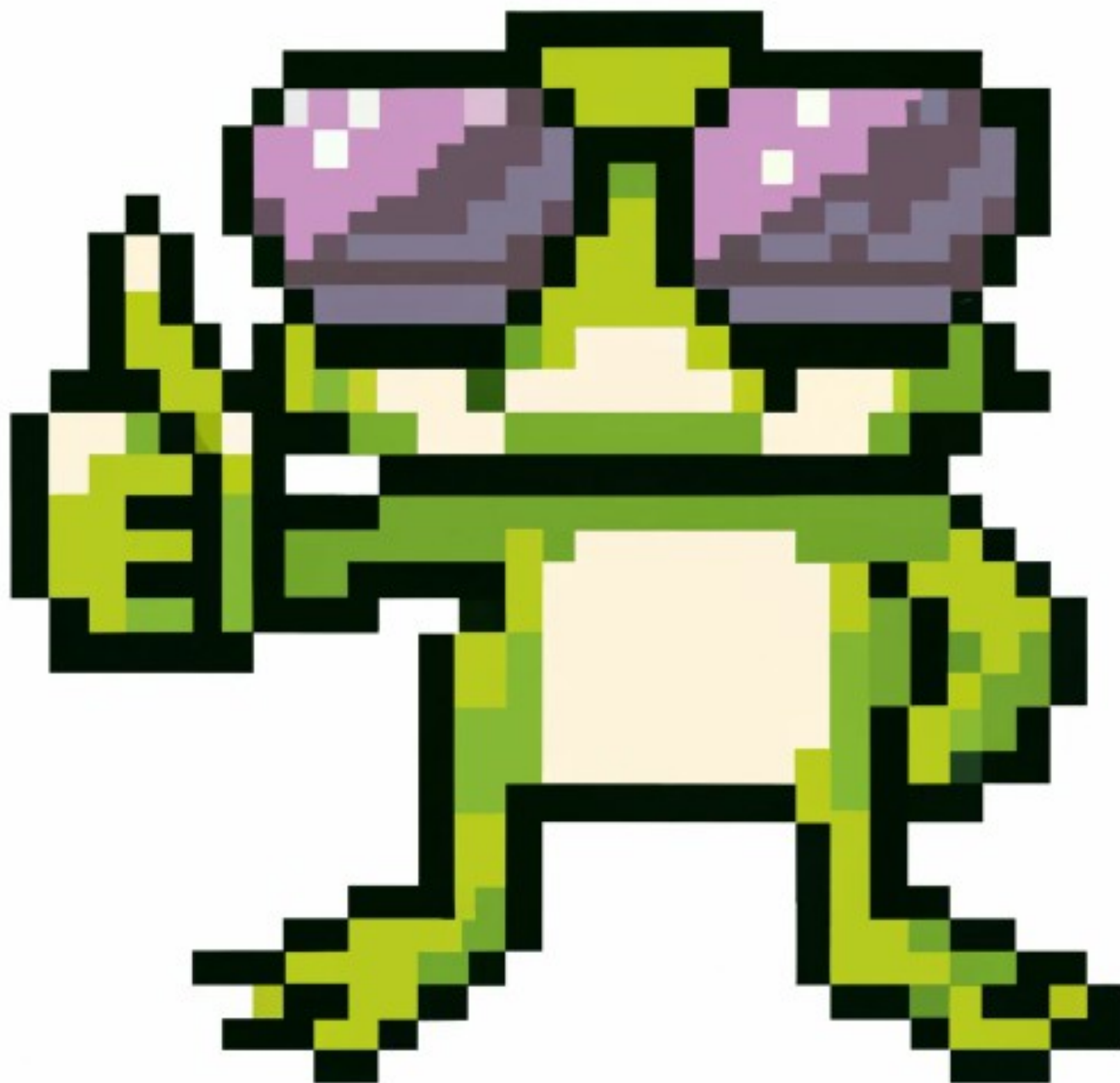






***HACK***







Add a comment...

---