Lesson 1: Introduction To Commands And Variables

Some basic information

1. Code runs line-by-line; what appears on line 1 will happen before what is on line 2. Line 2 will happen before what is on line 3. This continues until the end of the lines is reached.
2. Blank or empty lines are ignored by the computer.

What is a command?

A command is a tool used to tell a computer what to do. A computer will do EXACTLY what you tell it to. Nothing more, nothing less. A command is an instruction that performs a task of some sort.

The first command we will look at is the `print()` function.

The `print()` function is a command we use to tell the computer to output a series of letters.

In [1]:
```
# Example:
print("Hello World!")
```

```
Hello World!
```

As you can see above, the `print("Hello World!")` command outputs the letters given to it. The line above the command starts with a `#` charachter. This tells the computer that we are writing a comment. The computer will ignore any lines starting with the `#` character.

There are also things known as data types. For now, all you need to know about them is that the `print()` function requires a data type known as a `string` . What is a `string` ? It is a collection of one or more letters, numbers, or punctuation surrounded by double quotes; `"` .

In [14]:
```
# Example 1:
# Good command input
print("This is a string!")

# Example 2:
# Good command input
print("This is a 2nd string that contains a number!")
```

```
This is a string!
This is a 2nd string that contains a number!
```

In [15]:
```
# Example 3:
# Bad command input
print(this is not a string and will cause an error)
```

```
  File "<ipython-input-15-c93f56b510e5>", line 3
    print(this is not a string and will cause an error)
                               ^
SyntaxError: invalid syntax
```

What Is A Variable?

A `variable` is a way to store data for use later. `Variables` are similar to a freezer. Pretend you decide to buy a tub of ice cream. You need somewhere to keep it safe until you want to eat it. So you put the ice cream in the freezer so you can have dessert after dinner. A `variable` is similar in that you can store data in it for later use. As mentioned previously, there is a concept known as data typing. For now all you need to know is that `variables` can store the `string` data type we learned about previously.

You can give a `variable` almost any name you want, with some rules. A variable cannot start with a number. **1wholeTubOfIceCream** is not an allowed name because it starts with the number '1'. **OneWholeTubOfIceCream** or **one_whole_tub_of_ice_cream** is allowed as either of these start with a letter. `Variable` names also cannot have spaces in them. **one whole tub of ice cream** is not allowed as a `variable` name because it contains spaces. There are other rules, but for now just know you need to start a `variable` name with a letter and you should name it something helpful.

Example: **TubOfIceCream** is much more helpful than a name such as **T**.

You put data into a `variable` with the `=` symbol. This may be best explained with an example.

In [21]:
```
# Example 4:
# Note that the variable name cannot have spaces,
# but the string data type can.
OneTubOfIceCream = "Chocolate Chip Cookie dough"

# Now we can use the print command we learned about
# on the variable we created.
print(OneTubOfIceCream)

# Example 5:
# We can also call a variable one letter.
T = "Vanilla"
# Now call the print commandon the variable.
print(T)
```

```
Chocolate Chip Cookie dough
Vanilla
```

In [9]:
```
1 # Example 6:
2 # An example of an error
3 One Tub Of Ice Cream = "Mint Chocolate Chip"
4 # Now call the print command on this new variable we created.
5 print(One Tub Of Ice Cream)
```

```
  File "<ipython-input-9-7df9e893dfff>", line 2
    One Tub Of Ice Cream = "Mint Chocolate Chip"
            ^
SyntaxError: invalid syntax
```

As you can see above, the `variable` we tried to name **One Tub Of Ice Cream** gave an error at the output. We never even reached the `print()` command because of this error!

`Variables` can be changed as you please. Below is an example of updating a `variable` before passing it to a command.

In [20]:
```
1  # Example 7: (A little more complex)
2  # Set a string variable to be the main sentence
3  BowlOf = "I want a bowl of {} flavored icecream."
4  # Set a string variable to be a flavor
5  flavor = "vanilla"
6  # Call the print command.
7  print(BowlOf.format(flavor))
8
9  # Set a string variable to be a flavor
10 flavor = "chocolate"
11 # Call the print command.
12 print(BowlOf.format(flavor))
13
14 # Set a string variable to be a flavor
15 flavor = "mint chocolate chip"
16 # Call the print command.
17 print(BowlOf.format(flavor))
18
19
```

```
I want a bowl of vanilla flavored icecream.
I want a bowl of chocolate flavored icecream.
I want a bowl of mint chocolate chip flavored icecream.
```

So what is going on here? Let's break it down line by line.

Line 3: `BowlOf = "I want a bowl of {} flavored icecream"`

Here we are setting the `variable` **BowlOf** to hold the `string` "I want a bowl of {} flavored icecream". But what is going on with the `{}` part? Well these are a special part of `strings`. They let you call the `format()` command from the `string` and insert another `variable`!

Note: The scope operator `.` is a lesson for another time and will not be covered here, for now just know that you use it to access the `format()` command with a `string`.

Line 5: `flavor = "vanilla"`

Here we are doing as we've done before, and setting a `variable` to the `string` vanilla.

line 7: `print(BowlOf.format(flavor))`

On this line, we are getting into the meat and potatoes of the program. Notice that we are calling the `print()` command on the `variable` **BowlOf**. From there we are using the `string` method (command) `.format()` on our other `variable` **flavor**. The `.format()` method (command) takes whatever is given to it and replaces the `{}` symbols in the **BowlOf** `string`.

We then repeat two more times, changing the **flavor** variable to different types of ice cream.

Summary:

We have learned about commands and variables. Commands are instructions we provide to the computer to perform tasks. Variables are containers that hold data for us.

Commands:

- `print()` Outputs letters and numbers that are between `"` characters or data stored in `variables`.
- `.format()` Replaces a set of `{}` (curly brackets) in a `string` with the data stored in a `variable`. Can only be accessed through a `string`.

Variables:

- Can have any name that does not start with numbers and does not have spaces.
- Can hold the `string` data type we learned about.

Assignment:

Using the `print()` command, `variables`, and anything else you learned, tell me each of your pets names.

Example output:

```
My pet's name is AppleJack.
My pet's name is Precious.
```

In [ ]:

```
1
```