Basics of Pythons

Intro to Coding on Python

Bernadette and Astrid

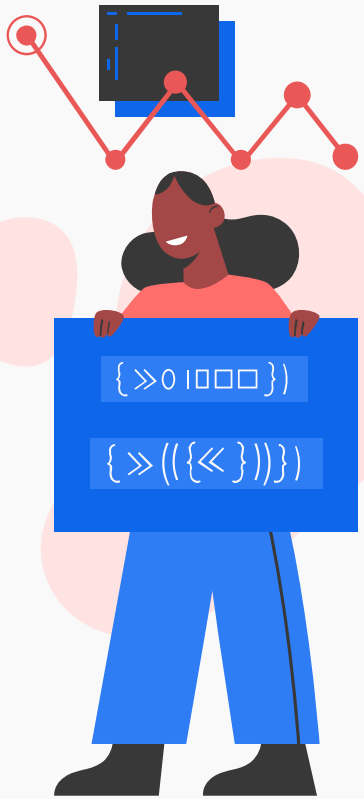
Quick Summaries

We will expand on these later but just to get these fresh in your mind

Imports	Statements that transfer code from other files to your own with pre-made functions
Conditionals	Allow programs to make decisions and display different paths for code to go
Loops	Makes it so a specific part of a program can be repeated a set amount of times
Operators	Help navigate code and are connectors of specific code segments
Basic Format	Code will not work unless colons, indentation, and spacing are right
Lists	Variables grouped together to hold information that can be later accessed

About Python

Python is a high level programming language, meaning that it is designed to be easily understood by humans. Since computers process information using binary code people developed “languages” like Python or Java to allow for more complex and logical algorithms to be translated into binary. To sum it all up, Python is used to express these ideas in a way that both computers and humans can understand.



Terms you need to know

Program
A collection of statements that performs a specific task when run by a computer. A program is often referred to as software.

Parameters
The input variables of a procedure. A set of information included to help a process or function operate as intended.

Method/Procedure
A predefined set of instructions that can be used to accomplish a task. It is available on a specific object. For example, a turtle object has a list of methods that can be used to manipulate the object, such as forward(), backward(), and so on.

Variable
An abstraction inside the program that can hold a value. Each variable has associated data storage that represents one value at a time, but that value can be a list or other collection that in turn contains multiple values.

Imports

Imports are very useful to save time on your coding project and can help expand the code itself. You use imports to insert “premade” functions.

Some commonly imported functions to learn:

- Turtle modules
- Random

```
#Importing turtle module
import turtle as trtl
window = trtl.Screen()
window.mainloop()
```

Random is a collection of functions that can be used to randomized something as you can guess.

Importing random will look like this: More will be added in code but before you do that, you have to import it.

```
#Importing random
import random
```

Simple Commands for Your Imported Function

As we said previously the “trtl” is an abbreviation of our imported turtle function. To command your function to act you need to format it as you see. A period after your function and then the command follows. There is a list of certain commands that are valid, so you can't just make up any command that sounds right, but there are many to choose from! In the parenthesis after the command is where your parameters are. For example, in trtl.backwards(4), the four is your parameter and makes it so that the turtle moves four pixels backwards.

Turtle Commands:
trtl.circle(insertsize)
trtl.backward(insertdistance)
trtl.forward(insertdistance)
trtl.pencolor(“insertpythoncolorinquotes”)
trtl.left(angle)
trtl.right(angle)

Other Turtle Commands can be found here:
https://docs.python.org/3/library/turtle.html

Loops

There are two common loops used in python: while and for loops. They can be used to repeat segments of code until a specific condition is met.

```
count = 0
while count < 5:
    print(count)
    count += 1
```

This is a while loop used to display numbers from 0-4.

```
fruits = ["apple", "banana", "cherry"]
for fruit in fruits:
    print(fruit)
```

Example of a for loop used to iterate through a list of elements to print each element.

See how much more efficient it was compared to printing each variable!

Operators(Used in Parameters)

Operators are symbols or keywords that operate on one or more values(used in conditionals)

Relational

Example showed on next slide

Used to compare two values and return a boolean result(true or false).

- == (Equal to)
- != (Not equal to)
- > (Greater than)
- < (Less than)
- >= (Greater than or equal to)
- <= (Less than or equal to)

Logical

Used to combine conditional statements.

- and (Returns True if both operands are True)
- or (Returns True if at least one operand is True)
- not (Reverses the logical state of its operand)

What Are Conditionals?

Conditionals are the “if… then” of programming. In your code if you wanted a result to rely on an action you would use a conditional. To code a conditional in Python you would use if and else. Formatting a conditional is very specific, you need to start with if and follow it with your variable and an operator as pictured. After those are established you need to put a colon to define the new code block.

```
test = 1


if test == 1:
    wn.bgcolor("pink")
else:
    wn.bgcolor("blue")
```

This conditional turns the background color different depending on if the variable “test” equals 1. Since test == 1, it returns a true result where background color is set to pink.

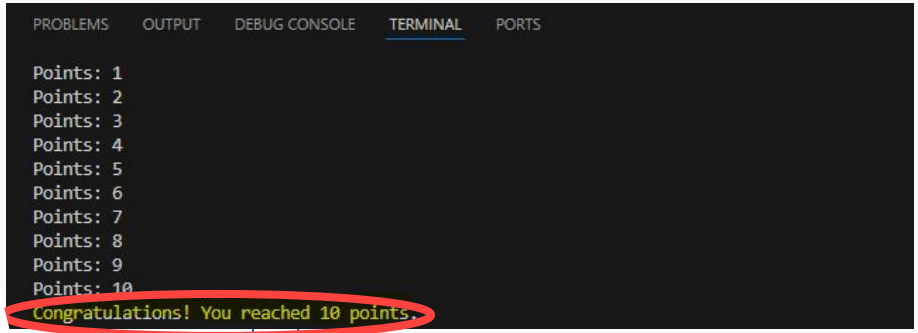
Print Statements

Print statements display words in your code. Formatting them are pretty simple, it's just print("input text here"). The main thing never to forget are the quotation marks. For example, print statements can be used for asking questions for your potential users to answer, displaying instruction for the user to follow in your program, etc.

Example taken from code

```
print("Congratulations! You reached 10 points.")
```

What it looks like when printed



Importance of Code Syntax

Colon

Colons need to be directly placed after conditionals, loops, and when creating a function, or else the program will deem a syntax error and the program will malfunction.

Spacing

Proper spacing between code segments are recommended to increase code's organization and readability.

Indenting

For the next line after a colon is placed, an indent NEEDS to be placed. Sometimes it will be placed automatically, but be sure to check.

Lists

Lists in Python are used to store collections of items, which can be of any type (e.g., numbers, strings, objects). You can add, remove, or modify elements in a list, and you can also access them via indices (starting from 0). You can use the import random and assign it to your lists and the program will choose a random value from your list. The random.choice() method can be used to randomly select an item from a list. If your lists contains words instead of numbers you need to have quotes around each value and comma after.

```
testlist = ["blue", "pink", "yellow"]
```

```
for step in testlist:
    color = rand.choice(testlist)
    wn.bgcolor(color)
```

In this example a list is made with three values. I used a loop to access the list. I initiated the “color” variable by assigning a random value using the random import from my list to it. Using the color variable in place of hardcoding, a color in the program takes a random color from my list and makes that the background color.

Try It Yourself!

For your final, I challenge you to attempt to make a flower utilizing everything we have learned today!

Try to start a turtle module like seen in the example, start a screen, and loop it. Once you have the basics, you can add things to have your trtl do.

List of eligible turtle colors ↓

aquamarine	blue2	chartreuse1	cyan	DarkOliveGreen3	DarkSlateGray
aquamarine1	blue3	chartreuse2	cyan1	DarkOliveGreen4	DarkSlateGray1
aquamarine2	blue4	chartreuse3	cyan2	DarkOrange	DarkSlateGray2
aquamarine3	BlueViolet	chartreuse4	cyan3	DarkOrange1	DarkSlateGray3
aquamarine4	brown	chocolate	cyan4	DarkOrange2	DarkSlateGray4
azure	brown1	chocolate1	DarkBlue	DarkOrange3	DarkSlateGrey
azure1	brown2	chocolate2	DarkCyan	DarkOrange4	DarkTurquoise
azure2	brown3	chocolate3	DarkGoldenrod	DarkOrchid	DarkViolet
azure3	brown4	chocolate4	DarkGoldenrod1	DarkOrchid1	DeepPink
azure4	burlywood	coral	DarkGoldenrod2	DarkOrchid2	DeepPink1
beige	burlywood1	coral1	DarkGoldenrod3	DarkOrchid3	DeepPink2
bisque	burlywood2	coral2	DarkGoldenrod4	DarkOrchid4	DeepPink3
bisque1	burlywood3	coral3	DarkGray	DarkRed	DeepPink4
bisque2	burlywood4	coral4	DarkGreen	DarkSalmon	DeepSkyBlue

Thanks!

Do you have any questions or want to know more complicated code?
Feel free to ask now, or find us later!
We are happy to help you find answers!
Thank you for taking the time to attend our workshop!