Sup

Time for some Python

Hackathon Club 2020

Python

The thing

What does it do?

- Interpreted language
  - Stores program as byte code, NOT machine code
  - All you need is an “interpreter” to run it on any computer

What can it do?

- Run scripts on your computer
  - Automatically open programs, write text, “scrape” things off the Internet, etc
- Run simple servers
- Make games
- Almost anything
  - However, it has strengths and weaknesses

Strengths vs Weaknesses chart

Strengths

- “High level” language
  - Very simple, don’t need to worry about a lot of syntax issues like in low level languages
- Lots of libraries
  - Very popular language, lots of tools, resources, libraries (tools that you use in your code), interpreters
  - Able to use C and C++ libraries
- Can run on any computer
  - As long as it has an interpreter for it, which most do

Weaknesses

- Very slow
  - Lower level languages can be optimized to a much higher degree than Python can
  - Can be improved by conversion tools to lower level languages (underdeveloped)
- Weak for mobile development
  - Not very integrated with mobile devices
- Run time errors
  - Being dynamically typed means it isn’t able to catch many errors that are easily found with static code analysis (automatic error checking in your IDE)

Why should we use this for Hackathon Club?

- High level, easy to program
- Lots of libraries to use
- Run on any computer
  - Windows, Mac, Linux
  - ...except TempleOS

Syntax

Comments

#wassup

#hello, this next line of code does something

“””Just kidding

Lol”””” # There’s a few more use cases for triple quotes which we will go over

Comment = False # this is a comment, the line before the # is valid code

Variables

Examples of invalid variable names

1Number
Hello*2
What-is-up
Hello Apple

String = ‘a’

Also_string = “Hello!”

Also_also_string = “””Hello \””” # Allows multi line strings and “escape characters”

Number = -4 # This is an integer

Number = 4.2222 # This is a floating point number

Boolean = True

“””Remember, no spaces or numbers in front in your variable names! (Or any escape character like \) Certain variable names like “if”, “else”, “def” are also not allowed”””

Operators

```
Counter = 0

Counter += 10 # Counter = 10 This is the same as Counter = Counter + 10

Counter %= 2 # Counter = 0

Name = “Nick”

Name += “ Walls” # Name = “Nick Walls”
```

Comparisons

Number = 1 # This is ASSIGNING the value “1” to the variable “Number”

Number == 2 “””This is COMPARING the value OF the variable “Number” to 2. This returns False”””

Number != 2 # This returns True because the value of “Number” is not 2

Number >= 0 “””This returns True because the value of “Number” is greater than or equal to 0”””

Print

```
print("Hello") # Prints "Hello" to a console window

"""This is the most basic function of Python and the most useful. It can be used to debug (find problems in) code to see what code runs or returns, or provide information."""

Number = 3

print(Number) # Prints the number "3" to a console window
```

Errors

```
if False ISNOTEQUAL True:
                   ^
SyntaxError: invalid syntax
```

```
misspelled_variable_name

NameError: name 'misspelled_variable_name' is not
defined
```

```
numerator = 100

denominator = 0

bad_results = numerator / denominator

ZeroDivisionError: division by zero
```

“if” statement

```
watch_out = “corona”

if watch_out == “shaurya tornado”:

    print(“Oh no!”)

elif watch_out == “steve in smash”:

    print(“I’m OK with that”)

else:

    print(“I don’t know what to watch out for. This is bad.”) # This will run
```

Activity

Make a variable that has an integer.

If the integer is greater than 10, square the variable.

If the integer is less than or equal to 10, print “Oh no”

To do for rest of club time

- Play with things
  - Try to make something using what you know now
- You can start learning more Python using online resources, try to do some of this before next meet
- Ask for help if you need it

Day 2

What are functions?

- Code sections which store code that, when called, will run the code inside.
- This is useful for running specific operations that would otherwise be repeated.

```
def hello():

    print("Hello")

# When the indent ends, the function ends

hello() # Will print "Hello" to a console window

hello() # Will do it again

hellogamers() # Will give an error. Functions need to be defined first
```

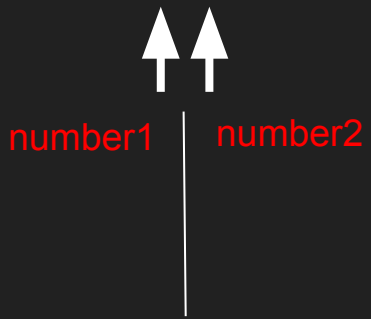
Functions with returns

- Returns allow you to perform a function and then output a value afterward.
- You can assign this output to a variable or perform further operations (for example print it)

```
def hello():

    return “Hello”

hello() # Does nothing

print(hello()) # Prints “Hello”

welcome = hello() “””Assigns “Hello” to the variable ‘welcome’ “””

print(welcome) # Also prints “Hello”
```

Functions with parameters

```
def add(number1, number2):

    return number1 + number2 # Returns the sum of the first and second number

sum = add(4, 6) # Variable “sum” now equals 10
```



This is essentially your “input” for a function. You can both provide input, and return an output as seen above.

Modules

Modules are interfaces which allow you to use other people's code. They can give you functions to use, data, etc, and make coding much easier.

You need to add modules to your “environment” which allows your interpreter to use the modules when running your code.

Depending on your specific environment (IDE, portable environments like WinPython, etc) you will need to follow different instructions. Look up the instructions specific to your setup.

The module we will be using today is “requests”. This comes from the pip package manager (a tool used to install Python modules). If you need help, feel free to ask for it during our meeting.

Modules

```
import requests # This adds the module to your code

X = requests.get("https://google.com")

"""This calls the "get" function from the requests module. This is a GET request to the website "google.com" and assigns the text to X"""

print(X.text) # Prints the text received from the above function

# Don't worry about the .text part for right now

# You are now all caught up and ready for our activity today!
```

Activity

- We will be scraping data from data.gov using the requests module
- More information will be available at meeting time