GENERAL ASSEMBLY

INTRODUCTION TO PYTHON PROGRAMMING

Joseph Mosby, Web Developer, APCO Worldwide

AGENDA

- Meta-Introduction
- Strings and Variables
- Conditionals
- Lists
- Loops
- Functions
- Dictionaries
- Importing and Modules
- More Modules
- Q & A

STRINGS

A string is a sequence of characters.

- “hello” - string
- “how are you” - string
- 123 - NOT a string
- “123” - string

STRINGS EXERCISE

- print “hello”
- print “we are happy to see you”
- print “123”
- print len(“hello”)
- print “we are happy to see you”[:12]
- print “we are happy to see you”[13:]

STRINGS EXERCISE

- print “hello” → hello
- print “we are happy to see you” → we are happy to see you
- print “123” → 123
- print len(“hello”) → 5
- print “we are happy to see you”[:12] → we are happy
- print “we are happy to see you”[13:] → to see you

VARIABLES

A variable is a named container for other data, like strings.

- hello = “hello”
- my_variable = “this is a string”

VARIABLES EXERCISE

- print “hello
- my_variable = “hello”
- print my_variable
- print “my_variable”
- my variable = “hello”

VARIABLES EXERCISE

- print “hello” → “hello”
- my_variable = “hello” →
- print my_variable → “hello”
- print “my_variable” → “my_variable”
- my variable = “hello” → SyntaxError

CONDITIONALS

A conditional allows you to change the flow of your program based on the state of certain conditions.

```
if x == 10:
    print “x was equal to 10!”
else:
    print “x was not equal to 10!”
```

CONDITIONALS EXERCISE

-
  ```
  my_variable = “hello”
  if my_variable == “hello”
          print “my variable was equal to hello!”
  else:
          print “my variable was not equal to hello!”
  ```

PAUSE

LISTS

A list is a compound data type that contains a list of items.

- [1, 2, 3, 4, 5]
- [“hello”, “this”, “is”, “a”, “list”]
- [“hello”, 1, 2, “list”]

LIST EXERCISE

- my_list = [1, 2, 3, “hello”, “list”]
- print my_list[0]
- print my_list[1]
- print my_list[3]
- my_list[3] = “this”
- print my_list
- print my_list[2:]

LOOPS

Loops allow you to perform tasks repeatedly until some condition is fulfilled.

```
y = [‘h’, ‘e’, ‘l’, ‘l’, ‘o’]
for x in y:
    print x
```

```
y = 5
while y > 0:
    print “hello”
    y = y - 1
```

LOOP EXERCISE

```
my_list = [‘h’, ‘e’, ‘llo’, ‘w’, ‘o’, ‘rld’]
count = 0
for letter in my_list:
        print count + “:” + letter
        count += 1
```

FUNCTIONS

Functions are blocks of reusable code that perform some repeatable action.

```
def foo(x_list):
    y = 0
    for item in x_list:
        y = y + item
    return y
```

FUNCTIONS EXERCISE

```
def foo(x_list):
    y = 0
    for item in x_list:
        y = y + item
    return y

list_one = [1, 2, 3, 4, 5]

print foo(list_one)
```

PAUSE

DICTIONARIES

Dictionaries are complex data structure with “keys” mapped to “values”

{‘animals’: [‘dog’, ‘cat’, ‘bee’], ‘trees’: [‘oak’, ‘pine’, ‘elm’], ‘instructor’: ‘Joe’}

DICTIONARIES EXERCISE

```
my_dict = {‘first_name’: ‘Joe’, ‘last_name’: ‘Mosby’, ‘city’: ‘Washington’}

print “Hello.”
print “My name is “ + my_dict[‘first_name’] + “ “ + my_dict[‘last_name’]
print “I live in “ + my_dict[‘city’]
```

MODULES

Modules are separate code files that you can import to use in your own programs.

```
import math
print math.sqrt(65536)
```

THE PYTHON STANDARD LIBRARY

import…

math

datetime

calendar

random

pickle

sqlite3

zipfile

csv

os

email

json

HTMLParser

xml.dom

webbrowser

and more!

IMPORTING EXERCISE

```
import math, random
my_numbers = {}
my_numbers[‘random’] = random.randint(5,55)
my_numbers[‘random_plus’] = my_numbers[‘random’] + 42
my_numbers[‘with_pi’] = math.pi

print my_numbers
```

MORE MODULES!

pip is a tool that installs additional modules for you from the web.

```
pip install requests
import requests
requests.get(‘http://docs.python.org’)
```

https://pip.pypa.io/en/latest/installing.html

Q&A

THANKS!

JOSEPH MOSBY

- josephmosby@gmail.com
- josephmosby.com
- twitter.com/josephmosby