Python 101

– a crash course –

Prepared by: Dr. Ali Hadi

Dec, 2014

Covering?

- Intro.
- Data types
- List, Tuple, Dictionaries, Strings
- Input / Output
- Structures: if, for, while
- Working with Files
- Functions
- Common Modules: os, time, sys

Who should attend?

- This is a basic python course
- We are not targeting experienced Python developers, so If you are a python developer already, then you are in the wrong room :)

Prerequisites

- You need a basic understanding of programming
- You need a basic experience writing working code in a modern programming language (C++, PHP, Perl, Java, JavaScript)
- You need a text editor, we'll use ***PyCharm***

Competition !

- There is a number of ***Do It Yourself (DIY)*** labs to complete
- At the end of the workshop there's a programming challenge to solve
- Winner(s) will receive a valuable prize

What?

- Open source scripting language
- Developed by Guido van Rossum in the early 1990s
- Name came from TV series “Monty Python’s Flying Circus”
- Cross platform (Linux, Windows, Mac, etc)
- Ideal language for scripting and rapid app dev
- Uses **.py** extension

http://www.python.org

Get it from

Ok huh … but Why?

- Python is
  - Simple, and easy to learn,
  - Free and Open Source,
  - powerful high-level programming language
  - relatively fast,
  - object-oriented,
  - elegant syntax,
  - widely used, and
  - Portable,
  - Extensive Libraries,
  - Interpreted, Extensible, Embedd

Install Time!

- Linux (out of the box)
  - v2.7.x
  - v3.2.x

- Windows
  - Next, Next, Next, Finish ☺

Running Python

- Interactive Shell
- Command Line Interface (CLI)
- IDLE
- PyCharm (*recommended*)

Data Types …

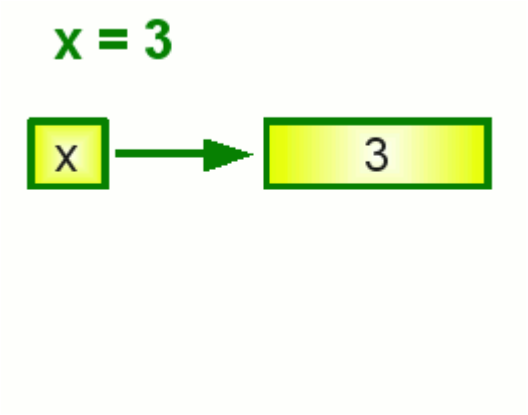
Basics

- a = 10
- b = 3.45
- c = ‘h’
- str = “Hello”
- x = 34 – 11

This is a comment.

Another comment!

See? no need to define the data type, Python figures it out!



a = “Python is awesome“
“Python” in a
“python” in a

Quotes

- Single (')

'This is a string with single quotes‘

- Double (")

“Welcome to PSUT’s infosec club”

- Triple, both single (''') and (""")

'''String in triple quotes can extend
over multiple lines, like this one, and can contain
'single' and "double" quotes.'''

DIY #1

- Not the usual hello world ☺
- Suppose you want to buy a prepaid recharge card. The card cost is 5JD and the card tax is 30%.

- Calc how much will this card cost?

Input / Output ...

Input / Output

- Use "**input**" for input ☺ and "**print**" for output

```
name = input("What's your name? ")
print("Nice to meet you " + name + "!")
age = input("Your age? ")
print("So, you are already " + age + " years old, " + name +
    "!")
```

- What's the data type for "name" and "age"?

type(name)

```
degree = input("PSUT degrees are? ")
>>> PSUT degrees are? BSc, MSc
Type(degree)
>>> str
```

DIY #2

- Looking back into DIY#1, you can only purchase one card! Modify your previous code to take the number of cards you want to purchase.

- Hint: don’t forget casting your input

Lists, tuples, dictionaries …

Lists

- Like arrays but far better!
  - Dynamic, no pre-definitions, no initialization

Examples:

- lst = ["Amman", "Zarqa", "Irbid", "Jarash"]
- lst2 = [2, "two", [1, 2, 3]] ← list with mixed data types

- len(lst2)
- lst[0]
- lst[2]
- Lst[-1] ← even negative indexes!

Lists: Slicing, Modify, etc

- names = [‘ali’, ‘zakaria’, ‘jolani’, ‘anas’]
- names[1:-1]
- names[0:3]
- names[0] = ‘hadi’
- names.**append**(‘mohammad’)

You can even append a whole list

- Other functions to check:
  - extend, remove, pop, reverse, sort, insert, min, max, index,etc

Boolean checks

- 3 in ["one", "two", "three"]

More Lists

- Concat

lst1 = [1,2,3]

lst2 = [4,5,6]

lst1 + lst2

- Multiply

lst1 * 3

- Complex lists:

[[01, ‘Ali’, ‘CS’],
[02, ‘Ahmed’, ‘CE’],
[03, ‘Zakaria’, ‘IS’],
[04, ‘Jolani’, ‘IS’] **]**

Tuples

- Like lists, but can’t be modified!
- All functions that doesn’t modify a list, could be applied to Tuples too!

Nice Trick

- Remember how to switch the contents of two variables?

  Move x to temp

  Move y to x

  Move temp to x

- With Python ☺

  x , y = y , x ← Called Unpacking

Strings

- Anything between two quotes is considered a string
- Strings like tuples are immutable (can't be changed)
  - We can deal with them using slicing, indexing, etc

- msg = 'Welcome PSUT Students\n'
- msg[-1] → '\n'
- msg[8:12] → PSUT
- Len(msg)
- Msg * 2

More Strings

- words = [“Python”, “under”, “Linux”, “is”, “great”]
- “ “.join(words) → “Python under Linux is great”

- s = “Amman is the Capital City in Jordan“
- words2 = s.split()

  → [“Amman”, “is”, “the”, “Capital”, “City”, “in”, “Jordan”]

Check lstrip() and rstrip() too

- Alpha = “a \t\tb c d \n\n efghijklm\n\topqrs\tt\uvwx\tyz”
- Alpha.strip() ← will remove all whitespaces

Even More Strings ☺

- Other string functions to check:
- find(), upper(), lower(), swapcase(), title(), isdigit(), isalpha(), isupper(), islower(), etc

- Formatting:

print ("My name is %s and weight is %d kg!" % (‘Ahmed', 88))

→ My name is Ahmed and weight is 88 kg!

“Our first {0} workshop at {1}".format("Python", “PSUT”)

→ ‘Our first Python workshop at PSUT'

Dictionaries

- major = {“CS“:400, “SE”:300, “CE":350, “IS”:100, “CG”:150}

```
major[‘CS’]
major[“IS”] = 250
major.pop(“CG”)
```

- Other functions to check:
  values(), items(), etc

DIY #3

- Write a program that will reads a students marks and adds them to a list.
- Print the final list of student marks
- Print the max and min mark in the list

Structures: if, for, while …

If/elif/else

```
char = input("please enter a single char: ")

if char.lower() in ['a','e','i','o','u']:
    print(“Char is a vowel")
else:
    print (“Char isn’t a vowel")
```

If you have multi decisions; use:

```
if condition:
    do_something
elif condition2:
    do_something
else:
    do_something
```

While Loops

```
sum = 0
counter = 1
while counter <= 5:
    num = int(input("Please enter a grade: "))
    sum = sum + num
    counter += 1

print("The sum for {} numbers is {}".format(5,
    sum))
```

For Loops

- Iterating through strings:

```
str1="Hello PSUT Students"
for char in str1:
    print(char)
```

- Iterating through a range of integers

```
for i in range(4):
    print(i, end=" ")
```

More For Loops

- Iterating through dict:

```
for key in major:
    print(key)
```

OR

```
for key in major:
    print("The key is %s and its value is %d" % (key, major[key]))
```

Even More For Loops

- Using the random Module

```
import random
for counter in range(5):
    rnumber = random.randint(0,99)
    print(rnumber)
```

What's this?

DIY #4

Answers?

- Write a simple guess the number game.
- The computer chooses a random number between 1 and N.

Game rules:

- You have 20 tries to guess the correct number.
- If your guess < chosen number → print “Number too small”
- If your guess > chosen number → print “Number too large”
- If you enter -1 → print “Sorry, that you're giving up!”

Functions …

Functions

```
def hello(name, course):
    print("Hello {}, welcome to our {}
   workshop".format(name, course))

hello('Ali', 'Python') ← calling our hello function
```

More Functions

```
def findlist(lst, value):
    for mem in lst:
        if mem == value:
            return True
    return False

lst = [2,4,6,8,10]
result = findlist(lst,8)
print (result)
```

Searching for a value in a list, and returns either True/False if found or not

Even More Functions

- Returning more than one value

```
def maxmin(lst):
    return max(lst), min(lst)

max , min = maxmin(lst)
print("The max # in the list is: %d" % max)
print("The min # in the list is: %d" % min)
```

Finding Max and Min

Even Even Even More Functions ☺

```
def linesInFile(filename):
    f = open(filename,'r')
    count = 0
    for lines in f:
        count += 1
    return count

nlines = linesInFile('file1.txt')
print("No of lines in %s are: %d" %
    ('file1.txt',nlines))
```

Count no. of lines in a text file

DIY #5

- Write a function that will take the number of lines required from the user and give the following output:

Line 0: *
Line 1: * *
Line 2: * * *
Line 3: * * * *

OR →

Line 0: 0
Line 1: 0 1
Line 2: 0 1 2
Line 3: 0 1 2 3

Answers?

DIY #6

- Write a function that will calculate the number of words within a text file.

Files (txt, csv) …

Text Files: Reading

```
f = open("file1.txt", "r")
for line in f:
    print(line, end=" ")
f.close()
```

Print each line in a file

Text Files: Writing

```
import random
f = open("file2.txt", "w")
for count in range(100):
    rnumber = random.randint(0,99)
    f.write(str(rnumber) + '\n')
f.close()
```

Generate a random number and store it in a text file

CSV Files

```
import csv
lst1 = []
f = open(‘emails.csv', 'r')
csv_f = csv.reader(f)
for row in csv_f:
    print ('Email: ', row[0])
f.close()
```

Read a CSV File and print its content

DIY #7

- Write a program that will read the file with random numbers we did previously.
- The program must then filter each number.
- If the number is odd, then add it to an odd list.
- If the number is even, then add it to an even list.
- Print both final lists.

Common Modules …

Custom Modules

- Suppose you want to create your own basic math functions. All you need to do is store them in a **.py** file (lets say **mymath.py**) and includes the following:

```
def add(x,y):
    return x + y
def mul(x,y):
    return x * y
```

- Now to use them:

```
from mymath import *
print (add(10, 20))
```

Useful Modules

- Time, math, os, sys,

More later

```
import time, math, os, sys
print ("The current time is: ", time.ctime())
print ("Factorial of 5 is: ", math.factorial(5))
print("The current working directory is: ",
    os.getcwd())
print("Your OS is: ", sys.platform)
```

DIY #8

- Add the work you did in DIY#6 and DIY#7 to a module name it "**psutworkshop.py**", then show how you'll be using it.

Finally, Challenge Time…

The Rules “not hard”

- You can submit your work only once
- Write your own answer, don’t search the Internet
- Student who submits his/her work first (correctly) wins

- The email must have the following:
  - Subject: PSUT Infosec Python Workshop #1

  To: psutinfosecclub@gmail.com
  - Content:
  - Name, Major
  - Your Code

Challenge: Stego

- Since our club is about Information Security, we made the challenge related to InfoSec; specifically “**Steganography**”.

- Write a Python program to read the HiddenMsg.txt file and extract the first character from every word in the file to regain and read the final message sent!

Summary

- We covered the basics of using Python
- What Lists, Tuples, and Strings are
- What is a dictionary and how to use them
- How to create programs with different structures and flows
- How to read and write to files
- How to deal with CSV files
- We showed some of the most used Python Modules

8 lines only !!!

```
from socket import *
serverSocket = socket(AF_INET,SOCK_STREAM)
serverSocket.bind((‘hostname’, 5555))
serverSocket.listen(1)
clientSocket, addr = serverSocket.accept()
print ( clientSocket.recv(1024) )
clientSocket.close()
serverSocket.close()
```

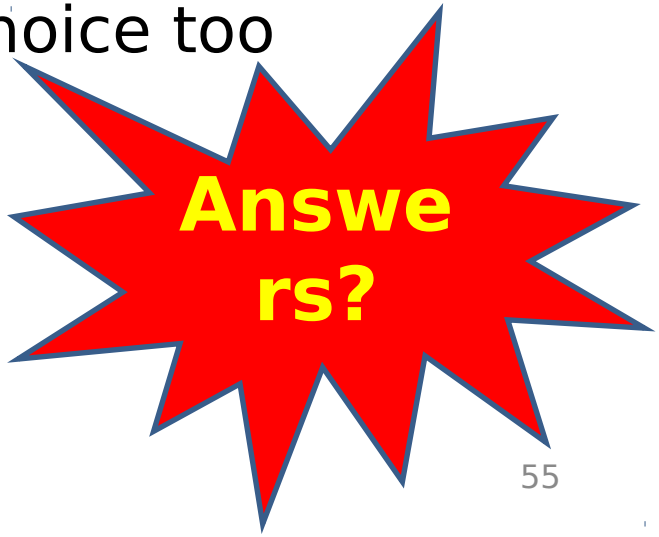
How simple is that?

Homework

- Write a the Rock, Paper or Scissors game.
- Use the **randint** function from the **random** module

Game rules:

- User chooses either rock, paper, or scissor
- Computer chooses randomly a choice too
- Finally,
  - Rock beats scissors
  - Scissors beats paper
  - Paper beats rock



Great Resources

- Python 3 Course, www.python-course.eu
- Udacity, www.udacity.com
- Python Challenges, www.pythonchallenge.com/
- Comma-separated value files, pymotw.com/2/csv/
- Python Exercises, www.ling.gu.se/~lager/python_exercises.html

What’s coming next?

- Modules, modules, modules
- Networking: TCP/UDP
- Packet Crafting: assembling your own packets
- Communicating with the Web
- Using Regex
- More Files: PDF, Images, etc
- Encryption, Decryption, and Hashing
- Security related stuff: to be added

- Anything specific in mind? Please mail it to us.