

Branch: master ▼

Find file

Copy path

EWU-Python-Workshop / README.md

shaykhsiddique Update README.md

b6517a1 6 minutes ago

1 contributor

Raw Blame History



394 lines (314 sloc) | 9 KB

Workshop on Python



Topics:

- [Overview Language](#)
- [Environment Setup](#)
 - Setup Python3
 - Virtual Environment for web application
- [Basic Syntax](#)
 - Variables
 - Operators
 - Decision making
 - Loops
- [Lists, Tuples, Dictionaries](#)
- [Functions](#)
- [Modules](#)

- [Problem Solving](#)
- [OOP in Python](#)
- [Python GUI Programming - Tkinter](#)
- Building a simple application using Tkinter.

Overview

Scripting vs Programming: Figure out between interpreter and compiler. Python's design philosophy emphasizes code readability with its notable use of significant whitespace ([Wiki](#)).

- Web development (server-side),
- Software development,
- Machine Learning, Statistics, Data Science.
- System scripting.

Environment Setup

Installing Python3 on:

- **Windows:** Download and install ([python.org](#)). *Command: Based on Environment Path. Default:* `python`
- **Linux:**

```
$ sudo apt-get update
$ sudo apt-get install python3
```

Command: For different versions of installing - python2: `python` , `python3:` `python3` .

Checking the Current Version of Python: `python --version` or `python3 --version`

[Go to topics](#)

Basic Syntex

Text Type: `str`Numeric Types: `int`, `float`, `complex`Sequence Types: `list`, `tuple`, `range`Mapping Type: `dict`Set Types: `set`, `frozenset`Boolean Type: `bool`Binary Types: `bytes`, `bytearray`, `memoryview`

Type of Variables:

Indentation is very sensitive in python. Scopes are defined by **semicolon** and **indentation**.

Data Type Casting

```
x = float(10)      #value of x will be 10.0
y = int(5.5)       #value of y will be 2
z = str(20)        #value of x will be "20"
```

Operators:

- Arithmetic Operators: `+`, `-`, `*`, `/`, `+`, `%`, `**`, `//`.
- Assignment Operators: `+=`, `-=`, `*=` etc.
- Comparison Operators: `>`, `<`, `==`, `!=`, `>=`, `<=`.

Python Decisions:

IF ELSE

```
a = 43
b = 111
c = 73
if b > a:
    print("b is greater than a")
elif a == b:
    print("a and b are equal")
else:
    print("a is greater than b")
```

Joining conditions

```
if a > b and a > c:  
    print("a is the leargest number of all.")
```

Loops: *While Loop*

```
i = 1  
while i < 6:  
    print(i)  
    i += 1
```

For Loop

```
for x in range(2, 6):  
    print(x)  
  
#loops in list  
fruits = ["apple", "banana", "cherry"]  
for x in fruits:  
    if x == "banana":  
        continue  
    print(x)
```

Problem:

- Leap Year Checking.
- Even and odd number from 1 to 100.
- Find out the numbers **divisible** by 3, donot use Modulus .

[Go to topics](#)

Lists, Tuples, Dictionaries

Lists/Tuples: Python collections or arrays.

```
# empty list  
my_list = []  
  
# list of integers  
my_list = [1, 2, 3]  
  
# list with mixed datatypes  
my_list = [1, "Hello", 3.4]
```

Inserting & deleting values

```
my_list = []
my_list[0] = 5
my_list[1] = "Hello"
print(my_list)

del my_list[0]
print(my_list)

print(my_list[1][1])
```

Slice lists in Python

```
my_list = ['p','r','o','g','r','a','m','i','z']
# elements 3rd to 5th
print(my_list[2:5])

# elements 4th to end
print(my_list[3:])

# elements beginning to 4th
print(my_list[:-5])
```

Traverse in list

```
my_list = ["apple", "banana", "cherry"]
for x in my_list:
    print(x)
print(len(my_list)) #find out the length of array

# integer type list
numbers = [2, 5, 6, 1, 9, 13, 20, 11]
for number in numbers:
    print(number)
```

Dictionaries: Python dictionary is like associative array.

```
#declaration of dictionary
thisdict = { "brand": "Ford", "model": "Mustang", "year": 1964 }
my_dict = dict()
```

Value insert/update and delete

```
thisdict["year"] = 2018
thisdict.update({"color": "Grey"})
```

Traverse in dictionary

```
# print keys
for x in thisdict:
    print(x)

# print values
for x in thisdict.values():
    print(x)

# print both keys and values
for x_key, y_val in thisdict.items():
    print(x_key, y_val )
```

[Go to topics](#)

Functions

Function is something like a machine. There will be some inputs and based on input it will generating some output.

```
def my_function():
    print("Hello from a function")

#call function
my_function()
```

Functions with parameters

```
def my_function(name):
    print("Hello" + name)

my_function("Alice")
my_function("Charlie")
```

Functions with parameters & return value

```
#set default value 5, if function is not called with parameter
def my_function(a = 5):
    b = a * 5
    return b

result = my_function(3)
print(result)

result = my_function()
print(result)
```

Modules

Consider a module to be the same as a code library. A file containing a set of functions you want to include in your application. *Save this file named with*

`mymodules.py`

```
def greeting(name):
    print("Hello, " + name)
```

In other file from the same directory:

```
import mymodule

mymodule.greeting("Rattlesnake")
```

Other Ways

```
from mymodule import *
import mymodule as mymdl
```

There are many build in modules.

Module	Description
<code>math</code>	Mathematical functions (sin() etc.)
<code>multiprocessing</code>	Process-based parallelism
<code>datetime</code>	Basic date and time types.

Module	Description
<code>random</code>	Generate pseudo-random numbers with various common distributions.

To see all modules of python [click here](#).

[Go to topics](#)

Problem Solving

- Find the leargest number.
- Find the repetation of strings.
- Search a number- Linear and Binary search.

Random Numbers: 81, 91, 15, 64, 66, 89, 23, 81, 62, 9, 18, 8, 26, 14, 47, 60, 60, 82, 17, 7

Object Oriented Programming

Classes/objects define

```
class MyClass:
    x = 5

# declare a class
p1 = MyClass()
print(p1.x)
```

Frist letter of class name is capital. It's not a rule but all programmers follow this.

The examples above are classes and objects in their simplest form, and are not really useful in real life applications.

Cnstructor: A special type of function which will autometically execute when creating a object.

```
#self is the current object variable name
class Person:
    def __init__(self, name, age):
        self.name = name
```



```
        self.age = age

p1 = Person("Alice", 28)
print(p1.name)
print(p1.age)
```

Inheritance in Python

```
from math import *

class Shape:
    def __init__(self, length):
        self.length = length

    def getlength(self):
        return self.length

class Circle(Shape):
    def area(self):
        return acos(-1)*(self.length ** 2)

class Rectangle(Shape):
    def __init__(self, length, width):
        self.length = length
        self.width = width
    def area(self):
        return self.length*self.width
```

[Go to topics](#)

Python GUI Programming

Most popular GUIs

- Tkinter
- JPython
- wxPython
- PyQt

Tkinter

Tkinter is the Python interface to the Tk GUI toolkit shipped with Python. *Example*

```
import tkinter
window = tkinter.Tk()
# your modification goes here
window.mainloop()
```

****Widgets:** There are many build-in widgets in Tkinter([see all from here](#))

Greeting Apps

```
import tkinter
import random

window = tkinter.Tk()
window.title("App Title")
window.geometry("450x350")
# window.configure(bg="sky blue")

def helloCallBack():
    msg = ["Hello", "Hi", "Nice to meet you, ", "Welcome", "Good morning"]
    output = tkinter.Text(window, width=40, height=5)
    output.grid(columnspan=5, row=6)
    output.insert(tkinter.END, msg[random.randint(0, 7)]+" "+str(E1.get()))

LH = tkinter.Label(window, text = "Greeting Apps", font=("Courier", 30, "bold"))
LH.grid(column=1, row=0)

window.grid_rowconfigure(1, minsize=20)

L1 = tkinter.Label(window, text = "Your name: ")
L1.grid(column=0, row=2)

E1 = tkinter.Entry(window, bd = 5, width=50)
E1.grid(column=1, row=2)

window.grid_rowconfigure(3, minsize=20)

B1 = tkinter.Button(window, text = "Click Me", command = helloCallBack)
B1.grid(column=1, row=4)

window.grid_rowconfigure(5, minsize=20)

window.mainloop()
```

Some System Apps Idea:

- Calculator
- Strong Password Generator
- Media Player
- Snake Game
- Weather Reporter
- Currency Converter
- Phone Book
- Tax Calculator
- Automated Email Sander.
- Network Monitoring Apps.
- Machine Learning Related - Persional Assistant, Translator, Object Identification etc.

[Go to topics](#)