Day1

May 26, 2021

1 Topic Outline:-

- Day-1 Basics, Variables, Types, Operators, Branching & Loops
- Day-2 Data Structures (list,dict,tuple,set), String Handling
- Day-3 Functions, Importing Modules
- Day-4 File Handling, Queries/Recall – Buffer slots
- Tools - PyCharm, Pylint, PyTest

2 LTTS Business Line:-

- Transportation - V
- HCM - V (Telecom, Semcon, M&E)
- Emb V&V - H (Test Automation)
- Medical & Life Sciences - V
- ICP - V
- DPS - H
- MECH Engg & Plant Engg
- DMS

3 Python Applied Areas:-

- Web Applications, Database
- Desktop Application
- Mobile Applications (little)
- AI, ML, Datascience
- Test Automation - V & V, Data Processing
- Image Processing, Video Analytics

4 Learning/Competency Pathway

- Python L1 - Essentials

- Python L2 - Intermediate/Advanced (Classes & Objects, Regex, Ex Handling, Modules & Package)
- Too Deep/Advanced into Core Areas (Fungible/Generic)
- Going through Applied Areas

4.1 Software Environment

- VS Code + plugins (ms-python.python, pylint)
- PyCharm IDE (Community Edition)
- Other ALternatives:-
  - Jupyter Notebook (for academic) / Jupyter Lab
  - Anaconda
  - Online Compilers
  - Sublime/Atom

4.2 Learning Resources:-

- sololearn - python core (python for beginners)
- learnpython.org - Learn the basics
- Additional
  - Programiz.com
  - py4e.com/lessons
  - freecodecamp.org
  - automatetheboringstuff
- Own examples – offline/live
- Assignments

5 Assessment

- MCQs
- 1 or 2 coding problems (given skeleton of a function, you need to fill the logic)

6 Comparison of langauge, build & exec models

- Python (vs) C/C++ (vs) Java
- C/C++ ==> compiled model
  - Compiler translates into stored binary
  - All syntax & linker errors are checked in advanced
- Python ==> interpreter model
  - No compilation, No stored binary
  - line by line translation , by platform specific interpreter
- Cons and Pros - compiled vs interpreted
  - Performance
  - Platform Dependency/Portability
  - Error checking/reporting
- Java - Hybrid Model, compiler generates platform
  independent intermediate code (byte code), such byte code interpeted by platform specific engine

7 Python versions

- Available versions
  - Python 2.7.x
  - Python 3.x
- First Program
  - Pycharm
  - Simple text editor + cmd line
  - REPL shell
  - VS Code
  - Online Compiler

8 Basics

- simple print
- comments – single line, multiline
- string literals – single line, multiline, using ", '
- docstring, **doc**
- Input in python
- Formatting String
- COnverting strint to integer
- Checking type and id
- Printing python version

[4]:

```
import sys
print(sys.version)
```

```
3.8.5 (default, Jan 27 2021, 15:41:15)
[GCC 9.3.0]
```

8.1 Types

- Typeless language/dynamic typing
- No prior declaration
- Based on assignments/usage context (implicit deduction of type)
- Types - int, float, bool, string, complex

[7]:

```
x=10
print(type(x))
x=2.3
print(type(x))
x="hello"
print(type(x))
x=False
print(type(x))
x=2+3j
print(type(x))
x=None
```

```
print(type(x))
```

```
<class 'int'>
<class 'float'>
<class 'str'>
<class 'bool'>
<class 'complex'>
<class 'NoneType'>
```

8.2 Variables, Types & Operators

- Name Error
- del
- Multi variable assignment
- Literal representation for various types
- type checking , type(x)
- isinstance
- Type Conversions – one type to other
- int(), float(), string(), bool(), ord(), number base in case of int

[8]:

```
a=10
b=20
c=30
print("sum is " + str(c))
print("sum is ", c)
print("num1=",a,",", "num2=",b)
print(f"num1={a},num2={b}")
print("num1={},num2={}".format(a,b))
print("num1={2},num2={0},num3={1}".format(a,b,c))
print("sum of {x},{y} is {z}".format(z=c,x=a,y=b))
```

```
sum is 30
sum is  30
num1= 10 , num2= 20
num1=10,num2=20
num1=10,num2=20
num1=30,num2=10,num3=20
sum of 10,20 is 30
```

8.3 Operators

- Arithmetic
  - +, -, *, % , same as C
  - / , //, **
- Relational
  - <, <=, >, >=, ==, !=

- Assignment
  - = , +=, -=, *=, %=, /=, //=, **=
- Logical - and, or, not
- Bitwise - &, |, ^, ~, <<, >>
- No incr/decr, a=a+1 or a+=1
- No conditional operator,sizeof, comma
- Special - in, not in, is

[9]:

```
a=10
b=6
c=0
print(a and b)
print(b and a)
print(a or b)
print(b or a)
print(a and c)
print(c and a)
print(a or c)
print(c or a)
```

```
6
10
10
6
0
0
10
10
```

8.4 TODO

- Complete examples till operators
- Self explore branching & loops
- Pre-read on strings & data structures (if time permits)

9 Assignment-1

- Write each problem as functions, call the function

9.0.1 Decision Making:-

- Biggest of three/four numbers
- Check if given number is armstrong or not
- Reverse the number, sum of digits
- GCD/HCF of two numbers

- LCM without computing GCD/HCF
- Check if given year is Leap year or not
- Type of triangle - equilateral, isosceles, scalene, right angled
- Roots of a quadratic equation
- Quadrant of a given point (x,y)
- Choice based arithmetic
- Fibonacci sequence, Tribonacci series
- Factorial of a given number
- Sum of the factors, n=30, 1+2+3+5+6+10+15
- Check if given character is vowel or consonant
- choice based arithmetic

9.0.2 Loops:-

- Digital root of a given number, 7895 -> 29 -> 11 -> 2
- List/count of prime numbers for given range
- Sum of triangular numbers, n=4, 1 + (1+2) + (1+2+3) + (1+2+3+4) = 20
- Maximum number by deleting single digit in a 4 digit number 5872 - 872, 9865 - 985
- Generate super prime numbers
- No.of combinations for n teams to play each other, i.e. nCr
- Generate number triangles/pyramids, pascal triangle

[ ]: