Core Python

Python3 Notes

Introduction to Python

- Python can be used as a general purpose as well as a scripting language.
- There are 2 versions of Python : Python 2.7 and Python 3.
- Version 2.7 is deprecated and ideally should not be used. Version 3+ 64 bit must be used.

Python3 Installation

- Download Python installer at : https://www.python.org/downloads/
- Start the installer by double clicking on the continue by clicking ‘Next’ till setup completes.
- Make sure to check all the checkboxes that pops up during the installation process.

Features

- Python is a compiled as well as interpreted language.
- No type-checking feature.
- Indentation is required to express blocks in code(Given by the Tab key).
- Supports object oriented style of programming.

Applications that can be built

- Desktop Applications
- Commandline Applications
- Server-side Applications

Domains Python is used

- Data science
- Artificial Intelligence
- Web development
- Automation Testing
- General purpose scripting
- Linux kernel dependencies

Variables

- Variables are containers for storing data values.
- A variable name must always start with an _ or an alphabet and are case sensitive.
- Declaration example :

  count = 5 or _value = “ABC”
- Variables do not need to be declared with any particular type and can even change type after they have been set.

Example program on variables

```
x = 7
y = "Test"
print(x) # prints 7
print(y) # prints Test
```

- To run the above program, paste it in a file with .py extension and run the file. On the terminal, go to the file location and “python <filename>.py”

More on variables...

- Supported data types are :

  int, float, complex, string, boolean, complex, list, tuples, dictionaries, set

- To print the data type of variable :

  x = 5

  print(type(x)) # type is an inbuilt function

- Python variables can store and process extremely large or small values.

Comments

- Comments are human readable sentences for understanding code in the program.
- Comments are ignored by Python during execution.
- There are two types of comments.

  single line (#) and multi-line (triple single quotes).
- Example :
- # This is a single line comment.

  ‘’’

  This

  is a

  multi-line

  comment.

  ‘’’

Accepting user input and printing.

- To accept user input, Python provides an inbuilt function called input.
- Example :

  x = input(“Give some value”)

  print(x)
- The accepted value gets stored in return variable.
- To print something on console, use the inbuilt print function.

If…else condition

- Example of if – else block

```
x = 30
y = 210
if x > y:
  print(“x is greater than y”)
else:
  print(“x is less than y”)
```

For loop

- For loop is used for iterating over a sequence (that is either a list, a tuple, a dictionary, a set, or a string).
- Example of for loop :

```
fruits = ["apple", "banana", "cherry"]
for x in fruits:
  print(x)
```

- Even strings are iterable objects, they contain a sequence of characters

Range function

- The range() function defaults to 0 as a starting value, however it is possible to specify the starting value by adding a parameter: range(2, 6), which means values from 2 to 6 (but not including 6).
- Example :

```
for x in range(2, 6):
  print(x)
```

- The range() function defaults to increment the sequence by 1.

While loop

- With the while loop we can execute a set of statements as long as a condition is true.
- Example of while loop :

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

Functions

- A function is a block of code which only runs when it is called.
- To let a function return a value, use the return statement:
- Example of function is :

```
def my_country(country):
  print("I am from " + country)


my_country("Bhutan")
my_country("India")
my_country("Brazil")
```

Arguments

- You can also send arguments in a key = value pair.
- Example is :

```
def my_function(c3, c2, c1):
  print("The youngest child is " + child3)
my_function(child1 = "Emily", child2 = "Toby", child3 = "Mathew")
```

- If you do not know how many arguments that will be passed into your function, add a * before the parameter name in the function definition. This should be the last parameter in this case.
- Example :

```
def my_function(*kid):
  print("The youngest child is " + kid[2])


my_function("Emily", "Toby", "Mathew")
```

Lists

- Lists is a way to store multiple values under a single variable.
- There is no type-checking in this case.
- No size limit on lists.
- Example of list :

```
l = [5,7,1]
for number in l:
    print(l)
```

Tuples

- Tuples are read-only lists.
- Example of tuple :

```
t = (1,4,7)
print(t)
```

- To update a tuple element or delete an element of tuple is not allowed.

Sets

- Sets are structures which contains only unique values.
- Example of set is :

```
s = {“Apple”, “Cherry”, “Banana”}
print(s)
```

- Sets are dynamic in nature.

Dictionary

- Dictionary is a key-value pair structure.
- It is dynamic in natue.
- Example of dictionary is :

```
person = {
“name”: “Jack”,
“age”: ”40”
}
```

Function del()

- To deallocate any variable memory, Python provides an inbuilt function del(). This is applicable on any variable type declaration.
- Example :

```
x = [1,4,5]

del(x)

print(x) # This would be error as
         # variable has been deallocated.
```

Object oriented progamming

- Python is an object oriented language.
- Almost everything in Python is an object, with its properties and methods. By default all properties are public. To make a propery private the variable name must start with__ (Example : self.__price);
- To create a class, use the class keyword.
- Example :

```
class Person:
  def __init__(self, name, age):
    self.name = name
    self.age = age

p1 = Person("John", 36)
print(p1.name)
print(p1.age)
```

More on OOPS.

- The word self points to the object that is getting initialized.
- The self object must be the first parameter of class methods by convention.
- The function __init__(self) is the constructor of the class. It is invoked everytime a new object is instantiated.
- There is no “new” keyword in Python.

Inheritance

- Inheritance allows us to define a class that inherits all the methods and properties from another class.
- Example of inheritance :

```
class Employee(Person): #Syntax
    pass # Empty class Employee
         # inheriting from Person
```

- super() function that will make the child class inherit all the methods and properties from parent.

Scopes

- There are two scopes in Python :

  Global scope and Local scope.
- There is no Block level scope.
- Any variable declared inside a function/method is only accessible inside that function (Local scope).
- Any variable declared outside of all functions are global variables. They are accessible throughout the lifetime of the program.

More on scopes

- Use the ‘global’ keyword if you want to make a change to a global variable inside a function.
- Example program :

```
x = 300

def myfunc():

  global x   #Had this been absent, Python would
   x = 200  # have treated x as new local variable


myfunc()

print(x) # Prints 200
```

File handling

- To open a file use the inbuilt function open().
- Modes in which files can be opened are read(“r”), write(“w”) and append(“a”).
- If the file does not exists. Python creates a new file in write and append mode.
- Example :

  file = open(“<filename>”,mode)

More on file handling

- In write mode, the previous content of the file gets truncated.
- In append mode, previous content of file remains intact.
- File can closed with close() function (file.close()).
- Always remember to close the file to avoid security/data corruption issues.

Exception Handling

- The try block lets us test a block of code for runtime errors.
- The except block lets us handle the error.
- The finally block lets us execute code, regardless of the result of the try- and except blocks.
- Example program :

```
try:
  print(x)
except:
  print("An exception occurred")
```

Executing commandline commands

- To execute shell or cmd commands, we need to import the os module.
- The os.system(“cmd”) is the function which allows us to execute commands on the terminal.
- Example :

```
import os
os.system(“ps -eclx”)
```