

Introduction to Python Programming

By: KnowledgeHut

Introduction

Like that of other languages such as C++, Java, Tcl, and JS, Python is also a high-level, interactive, object-oriented scripting language. It is easy to learn, read and understand. The syntactical constructions are easy and simpler as compared to other programming languages. Furthermore, it has few other key features and characteristics which make this language unique from other languages. In this chapter, you will get to know about what Python is and its advantages, features, difference with respect to Python 2 and why Python is preferred over other languages.

Characteristic Features of Python –

This language supports practical and planned programming methods along with OOP.

This language can be used as a lightweight scripting language or can be written so that it can be compiled in byte-code in order to build large applications.

This language offers very high-level, dynamic data types with real-time type checking.

This language is used for supporting automatic garbage collection.

This language has the capability to easily integrate with C, C++, ActiveX languages, COM, Java and CORBA.

Installations and versions 2.x and 3.x

To start coding the Python language, you have to setup its environment so that there you can write the Python code. So, in this chapter, you will learn about how to set up your Python environment in different OS and environment as Python is available in a wide variety of platforms.

Python Supporting Systems

Python can be used on any of the following –

Unix (Solaris, Linux, FreeBSD, AIX, HP/UX, SunOS, IRIX, etc.)

Win 9x/NT/2000

Macintosh (Intel, PPC, 68K)

OS/2

DOS (multiple versions)

PalmOS

Nokia mobile phones

Windows CE

Acorn/RISC OS

BeOS

Amiga

VMS/OpenVMS

QNX

VxWorks

Psion

Setup Python Environment

Installing Python for Linux –

The steps are:

First you have to update your APT Repository. You have to type the command in our terminal:

```
$ apt-get update
```

Now, install Python by typing the command –

```
$ apt-get install python3.6
```

Now, verify Python by typing

```
$ python
```

For Python3 type the command goes something like this.

```
$ python3
```

Setup Python Environment (cont....)

Installing Python in Macintosh systems –

You do not need to install or configure anything else for using Python 2. These instructions are documented in the installation of Python 3. The version of Python which ships with OS-X becomes good for learning.

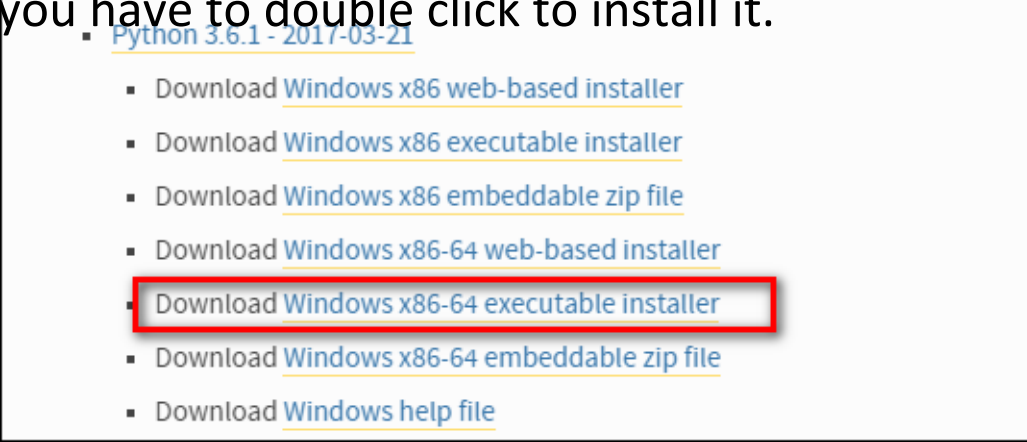
But if you want to have a stable release for Python, you will need to install GCC first. GCC can be obtained by downloading Xcode (<https://developer.apple.com/xcode/>), the smaller Command Line Tools (<https://developer.apple.com/downloads/>) (for this you must have an Apple account) or even the smaller OSX-GCC-Installer (<https://github.com/kennethreitz/osx-gcc-installer#readme>) package.

Setup Python Environment

Installing Python in Windows 10

Python doesn't come pre-packaged with Windows, but that doesn't mean Windows users won't find this flexible programming language useful. For this go to the official website of Python (<https://www.python.org/downloads/windows/>).

On there, you have to download your system compatible latest Python released version. It comes in both Windows x86-64 web-based installer as well as Windows x86 embeddable zip file / Windows executable installer. Once the file is downloaded, you have to double click to install it.

- 
- Python 3.6.1 - 2017-03-21
 - Download [Windows x86 web-based installer](#)
 - Download [Windows x86 executable installer](#)
 - Download [Windows x86 embeddable zip file](#)
 - Download [Windows x86-64 web-based installer](#)
 - Download [Windows x86-64 executable installer](#)
 - Download [Windows x86-64 embeddable zip file](#)
 - Download [Windows help file](#)

Soon as you run the setup file of the Python installer, you will see a screen where you will have to choose a radio button either “Install for all users” or “Install just for me”. Click Next > Next and let it install Python interpreter into your system.

Now to check whether Python has been installed in your system, you have to open command prompt in your system and type the following command –

python -v

History of python:

Python is a widely used general-purpose, high-level programming language. It was initially designed by Guido van Rossum in 1991 and developed by Python Software Foundation. It was mainly developed for emphasis on code readability, and its syntax allows programmers to express concepts in fewer lines of code.

Benefits

Presence of 3rd party modules: Python Package Index (PyPI) has a number of third-party modules which helps in making Python capable of cooperating with the majority of the other languages as well as platforms.

Extensive Support Libraries: This language offers a wide standard of libraries that includes areas such as internet protocols, string operations, web services tools as well as OS interfaces.

Open Source and Community Development: Python has been developed under an OSI-approved open source license that makes it free to use and distribute for personal as well as commercial purpose.

Easy to learn and understand.

Python has built-in list and dictionary as data structures which makes it very user friendly for constructing runtime data structure.

Python has a clean object-oriented design approach, and enhanced process management capabilities, along with easy text processing capabilities.

Python can be used in different technical domains such as data science, machine learning, robotics, game development, application development, embedded system development etc.

Invoking the Interpreter

The Python interpreter is usually installed as `/usr/local/bin/python3.8` on those machines where it is available; putting `/usr/local/bin` in your Unix shell's search path makes it possible to start it by typing the command:

```
python3.8
```

to the shell. Since the choice of the directory where the interpreter lives is an installation option, other places are possible; check with your local Python guru or system administrator. (E.g., `/usr/local/python` is a popular alternative location.)

How to Run Python Code Interactively

A widely used way to run Python code is through an interactive session. To start a Python interactive session, just open a command-line or terminal and then type `python`, or `python3` depending on your Python installation, and then hit Enter.

Here's an example of how to do this on Linux:

```
python3
```

```
Python 3.6.7 (default, Oct 22 2018, 11:32:17)
```

```
Copyright (c) 2001-2018 Python Software Foundation.  
[GCC 8.2.0] on linux
```

```
Type "help", "copyright", "credits" or "license()" for more information.
```

```
>
```

The standard prompt for the interactive mode is `>>>`, so as soon as you see these characters, you'll know you are in.

Now, you can write and run Python code as you wish,

Some Good Python IDEs & Code Editors

IDLE

PyDev

Sublime Text

Atom

GNU Emacs

Spyder

Basic Python Program

```
print "Hello, Python!"
```

Keywords in Python

Reserved Words/Keywords

The following list shows the Python keywords. These are reserved words and you cannot use them as constant or variable or any other identifier names. All the Python keywords contain lowercase letters only.

and	exec	not
assert	finally	or
break	for	pass
class	from	print
continue	global	raise
def	if	return
del	import	try
elif	in	while
else	is	with
except	lambda	yield

Built-in Python Functions

The Python interpreter has a number of functions that are always available for use. These functions are called built-in functions. For example, `print()` function prints the given object to the standard output device (screen) or to the text stream file.

Python Functions Exercise

Write a Python function (using def) to Print the Fibonacci series

What is String in Python?

A string is a sequence of characters. A character is simply a symbol. For example, the English language has 26 characters. Computers do not deal with characters, they deal with numbers (binary). Even though you may see characters on your screen, internally it is stored and manipulated as a combination of 0's and 1's.

String Program

all of the following are equivalent

```
my_string = 'Hello'  
print(my_string)
```

```
my_string = "Hello"  
print(my_string)
```

```
my_string = """Hello"""  
print(my_string)
```

triple quotes string can extend multiple lines

```
my_string = """Hello, welcome to  
the world of Python"""  
print(my_string)
```

How to access characters in a string?

We can access individual characters using indexing and a range of characters using slicing. Index starts from 0.

String Exercise

Write a Program using String to print your full name and fetch the surname only using specific index value.

Python Literals

Literals can be defined as a data that is given in a variable or constant.

- String Literals
- Numeric Literals
- Special Literals
- Literal Collections

Math Operators in Python

Addition

Subtraction

Division

Multiplication

Exponent

Modulo

Floor Division

Math Operators Exercise

Write a Python program to create a basic calculator containing all the basic operations.

Command line parameters –

Accessing Command Line Arguments

The Python `sys` module provides access to any of the command-line arguments via `sys.argv`. It solves two purposes:

`sys.argv`: is the list of command line arguments

`len(sys.argv)`: is the number of command line arguments that you have in your command line

`sys.argv[0]`: is the program, i.e. script name

What is Control Flow?

In computer programming, control flow or flow of control is the order in which function calls, instructions, and statements are executed or evaluated when a program is running. Many programming languages have what are called control flow statements, which are used to determine what section of code is run in a program at any time.

Python Indentation

Most of the programming languages like C, C++, Java use braces { } to define a block of code. Python uses indentation.

A code block (body of a function, loop etc.) starts with indentation and ends with the first un-indented line. The amount of indentation is up to you, but it must be consistent throughout that block.

if and elif Statements –

In Python, If Statement is used for decision making. It will run the body of code only when IF statement is true. When you want to justify one condition while the other condition is not true, then you use "if statement".

Syntax:

```
if expression
    Statement
else
    Statement
```

if statements

if test expression
statements

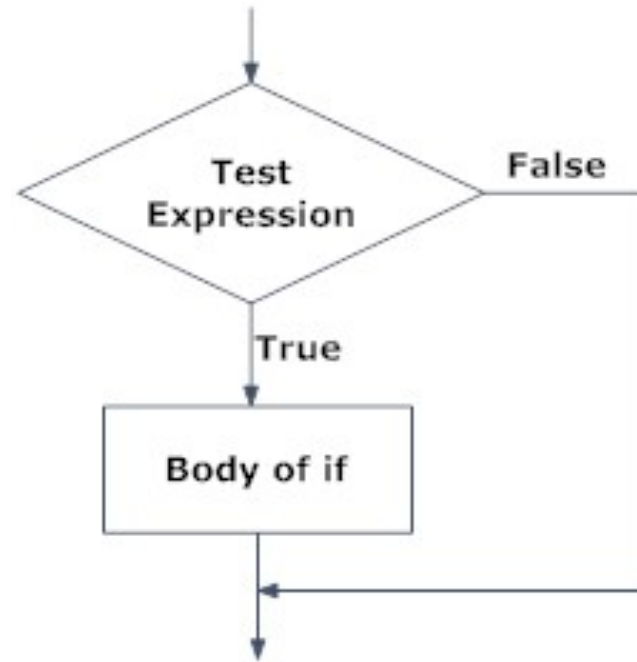


Fig: Operation of if statement

Python if...else Statement

Syntax of if...else

if test expression:

 Body of if

else:

 Body of else

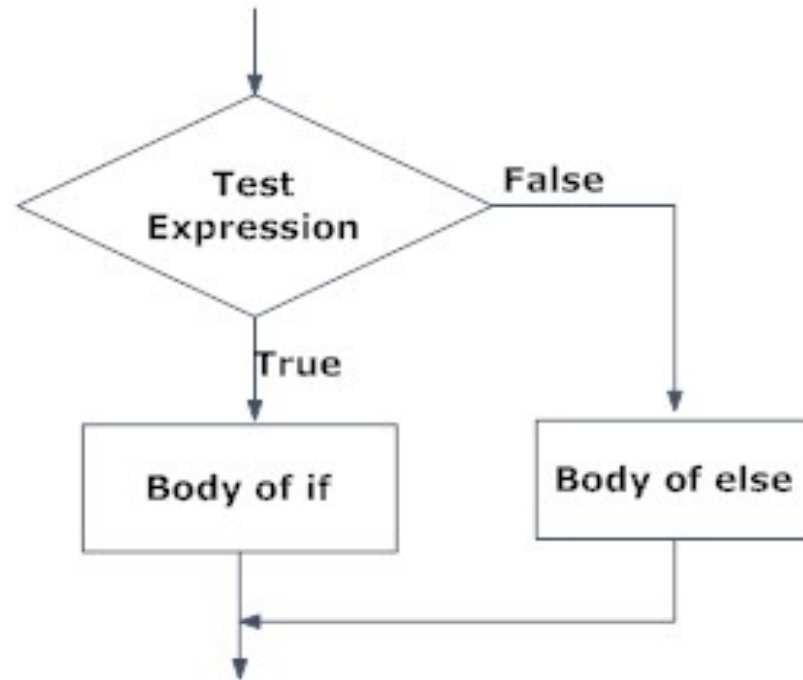


Fig: Operation of if...else statement

Nested if statements

We can have a `if...elif...else` statement inside another `if...elif...else` statement. This is called nesting in computer programming. Any number of these statements can be nested inside one another. Indentation is the only way to figure out the level of nesting.

Nested if Exercise

Write a program to check whether a number is even or odd

Write a program to find the greatest of 3 numbers

While Loop

The while loop in Python is used to iterate over a block of code as long as the test expression (condition) is true.

We generally use this loop when we don't know beforehand, the number of times to iterate.

Syntax of while Loop in Python

```
while test_expression:  
    Body of while
```

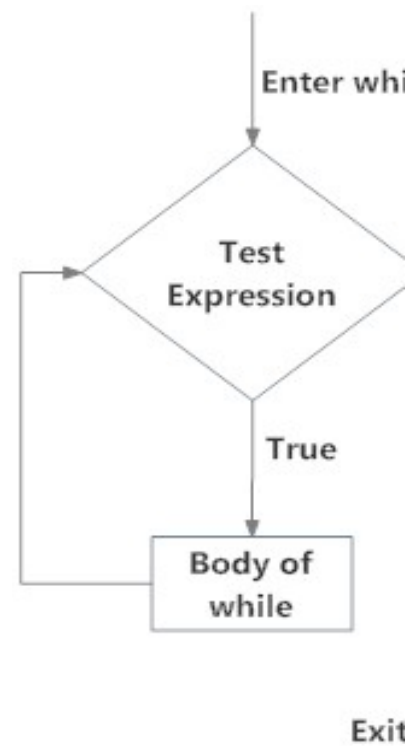


Fig: operation of while

For Loop

The for loop in Python is used to iterate over a sequence (list, tuple, string) or other iterable objects. Iterating over a sequence is called traversal.

Syntax of for Loop

```
for val in sequence:  
    Body of for
```

Loop Exercise

Write a program to print the multiplication table of 19 using Loop

The range() function: -

We can generate a sequence of numbers using range() function. range(10) will generate numbers from 0 to 9 (10 numbers).

We can also define the start, stop and step size as range(start, stop, step size). step size defaults to 1 if not provided. This function does not store all the values in memory, it would be inefficient. So it remembers the start, stop, step size and generates the next number on the go.

To force this function to output all the items, we can use the function list().

Sequences & File Operations

Lists

The most basic data structure in Python is the sequence. Each element of a sequence is assigned a number - its position or index. The first index is zero, the second index is one, and so forth. Python has six built-in types of sequences, but the most common ones are lists and tuples, which we would see in this tutorial.

Here are certain things you can do with all sequence types. These operations include indexing, slicing, adding, multiplying, and checking for membership. In addition, Python has built-in functions for finding the length of a sequence and for finding its largest and smallest elements.

The list is a most versatile datatype available in Python which can be written as a list of comma-separated values (items) between square brackets. Important thing about a list is that items in a list need not be of the same type. Creating a list is as simple as putting different comma-separated values between square brackets. For example –

```
list1 = ['physics', 'chemistry', 1997, 2000];
```

```
list2 = [1, 2, 3, 4, 5 ];
```

```
list3 = ["a", "b", "c", "d"]
```

Operations in Lists

Accessing Values in Lists

Updating The lists

Deleting List elements

Lists Exercise

Write a program to store the complete employee details in lists and display it.

Tuples

There is another type of sequence data structure that are immutable objects. They are called tuples. The Python tuples are similar to that of lists except that Tuples cannot be changed unlike lists. In this chapter you will get to know about how to use tuples in a Python program.

These are sequence data structures where the values are put as comma separated values. Tuple values are enclosed within parenthesis and they are immutable objects of Python. Example:

```
tupl1 = ('C++', 'Python', 4, 2);  
tupl2 = (8, 6, 4, 2 );  
tupl3 = "g", "k", "r", "s";
```

An empty tuple is defined with 2 parentheses having nothing inside –

```
tupl = ();
```

For writing a tuple with a single value, you have to make use of a comma, even though there is only a single value within it, like this –

```
tupl = (62,);
```

Like string indices, tuple indices start from 0.

Tuples Exercise

Create a Program to store values in a tuple name tup and then display all the stored values.

Indexing and Slicing –

Indexing

To retrieve an element of the list, we use the index operator ([]):

```
my_list[0] 'a'
```

Lists are “zero indexed”, so [0] returns the zero-th (i.e. the left-most) item in the list, and [1] returns the one-th item (i.e. one item to the right of the zero-th item). Since there are 9 elements in our list ([0] through [8]), attempting to access my_list[9] throws an IndexError: list index out of range, since it is actually trying to get the tenth element, and there isn't one.

Python also allows you to index from the end of the list using a negative number, where [-1] returns the last element. This is super-useful since it means you don't have to programmatically find out the length of the iterable in order to work with elements at the end of it. The indices and reverse indices of my_list are as follows:

```
0  1  2  3  4  5  6  7  8
```

```
'a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i']
```

```
-9 -8 -7 -6 -5 -4 -3 -2 -1
```

Indexing and Slicing (cont....)

Slicing: A slice is a subset of list elements. In the case of lists, a single slice will always be of contiguous elements. Slice notation takes the form

```
my_list[start:stop]
```

where start is the index of the first element to include, and stop is the index of the item to stop at without including it in the slice. So `my_list[1:5]` returns `['b', 'c', 'd', 'e']`:

0	1	2	3	4	5	6	7	8
x	↓	↓	↓	↓	x	x	x	x
'a',	'b',	'c',	'd',	'e',	'f',	'g',	'h',	'i']

Leaving either slice boundary blank means start from (or go to) the end of the list. For example:

```
my_list[5:]['f', 'g', 'h', 'i']
```

```
my_list[:4]['a', 'b', 'c', 'd']
```

Using a negative indexer will set the start/stop bounds relative to their position from the end of the list, so `my_list[-5:-2]` returns `['e', 'f', 'g']`:

'a',	'b',	'c',	'd',	'e',	'f',	'g',	'h',	'i']
x	x	x	x	↑	↑	↑	x	x
9	-8	-7	-6	-5	-4	-3	-2	-1

Exercise on Slicing

**Explain in your own words where you can make use of slicing concepts.
List some of the real-life scenarios where slicing has a significant
application.**

Iterating through Sequence (Lists & Tuples) –

```
list = [1, 3, 5, 7, 9]
```

```
# Using for loop
```

```
for i in list:
```

```
    print(i)
```

There are different ways to iterate through a tuple object. The for statement in Python has a variant which traverses a tuple till it is exhausted.

```
T = (10,20,30,40,50)
```

```
for var in T:
```

```
    print (T.index(var),var)
```

Built-in Functions

Python lists make use of the below mentioned functions –

`cmp(list1, list2)`: It is used for comparing elements of both lists.

`len(list)`: It is used for providing the total length of the list.

`max(list)`: This function is used for returning the item from the list having maximum value.

`min(list)`: This function is used for returning the item from the list having minimum value.

`list(seq)`: This function is used for converting a tuple into list.

Built-in Functions (cont....)

Python Tuples make use of the below mentioned functions –

`cmp(tuple1, tuple2)`: This function is used for comparing the different elements of both tuples.

`len(tuple)`: This function is used for giving the total length of the tuple.

`max(tuple)`: This function is used for returning the item from the tuple having maximum value.

`min(tuple)`: This function is used for returning the item from the tuple having minimum value.

`tuple(seq)`: This function helps in converting a list to tuple.

Enumerate() in Python

A lot of times when dealing with iterators, we also get a need to keep a count of iterations. Python eases the programmers' task by providing a built-in function `enumerate()` for this task. `Enumerate()` method adds a counter to an iterable and returns it in a form of `enumerate` object. This `enumerate` object can then be used directly in for loops or be converted into a list of tuples using `list()` method.

xrange Function:

This function returns the generator object that can be used to display numbers only by looping. Only particular range is displayed on demand and hence called “lazy evaluation”. The xrange object allows iteration, indexing, and the len() method. You can have a for loop to traverse it and gets the numbers in every iteration.

Generator Expressions in Python:

In Python, to create iterators, we can use both regular functions and generators. Generators are written just like a normal function but we use `yield()` instead of `return()` for returning a result. It is more powerful as a tool to implement iterators. It is easy and more convenient to implement because it offers the evaluation of elements on demand.

Unlike regular functions which on encountering a return statement terminates entirely, generators use yield statement in which the state of the function is saved from the last call and can be picked up or resumed the next time we call a generator function. Another great advantage of the generator over a list is that it takes much less memory.

Dictionaries and Sets –

Another Python data structure where each key is divided from its associated data/value using the colon (:). Also, the items within are comma separated values, and the entire thing is bounded within curly braces. You can also create an empty dictionary which can be written with just 2 curly braces - {}.

Dictionaries

It is to be noted that the keys need to be unique within the dictionary but the values can be redundant. The values within your dictionary can be of heterogenous type, but their keys need to be of an immutable data type (either of these: numbers, strings, or tuples).

Dictionaries Exercise

Implement the concept of dictionary to store a collection of username and password and display it

Sets in Python –

A Set is an unordered collection data type that is iterable, mutable and has no duplicate elements. Python's set class represents the mathematical notion of a set. The major advantage of using a set, as opposed to a list, is that it has a highly optimized method for checking whether a specific element is contained in the set. This is based on a data structure known as a hash table

Functions in Python: -

A function is a set of statements that take inputs, do some specific computation and produces output. The idea is to put some commonly or repeatedly done task together and make a function, so that instead of writing the same code again and again for different inputs, we can call the function.

Python provides built-in functions like `print()`, etc. but we can also create your own functions. These functions are called user-defined functions.

```
# A simple Python function to check
# whether x is even or odd
def evenOdd( x ):
    if (x % 2 == 0):
        print "even"
    else:
        print "odd"
evenOdd(2)
evenOdd(3)
```


File Handling in Python: -

Python too supports file handling and allows users to handle files i.e., to read and write files, along with many other file handling options, to operate on files. The concept of file handling has stretched over various other languages, but the implementation is either complicated or lengthy, but alike other concepts of Python, this concept here is also easy and short. Python treats file differently as text or binary and this is important. Each line of code includes a sequence of characters and they form text file. Each line of a file is terminated with a special character, called the EOL or End of Line characters like comma {,} or newline character. It ends the current line and tells the interpreter a new one has begun.

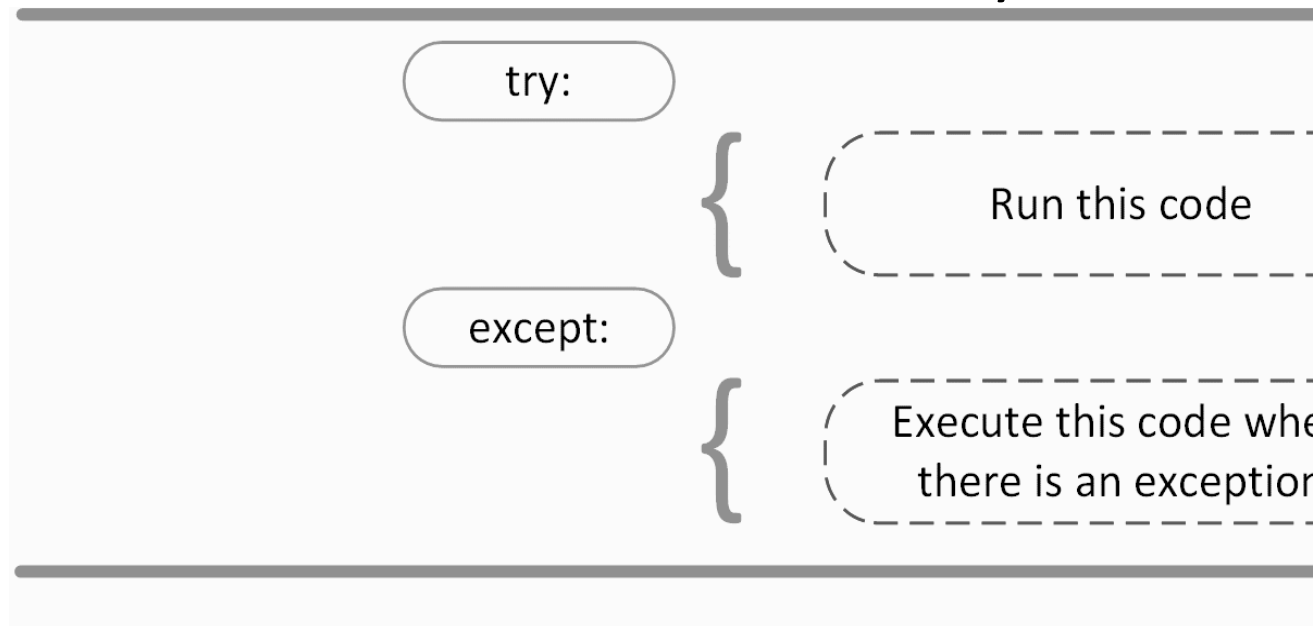
Different pre-defined file-handling methods are used to handle a file from within Python code.

Python pickle module

Python pickle module is used for serializing and de-serializing a Python object structure. Any object in Python can be pickled so that it can be saved on disk. What pickle does is that it “serializes” the object first before writing it to file. Pickling is a way to convert a python object (list, dict, etc.) into a character stream.

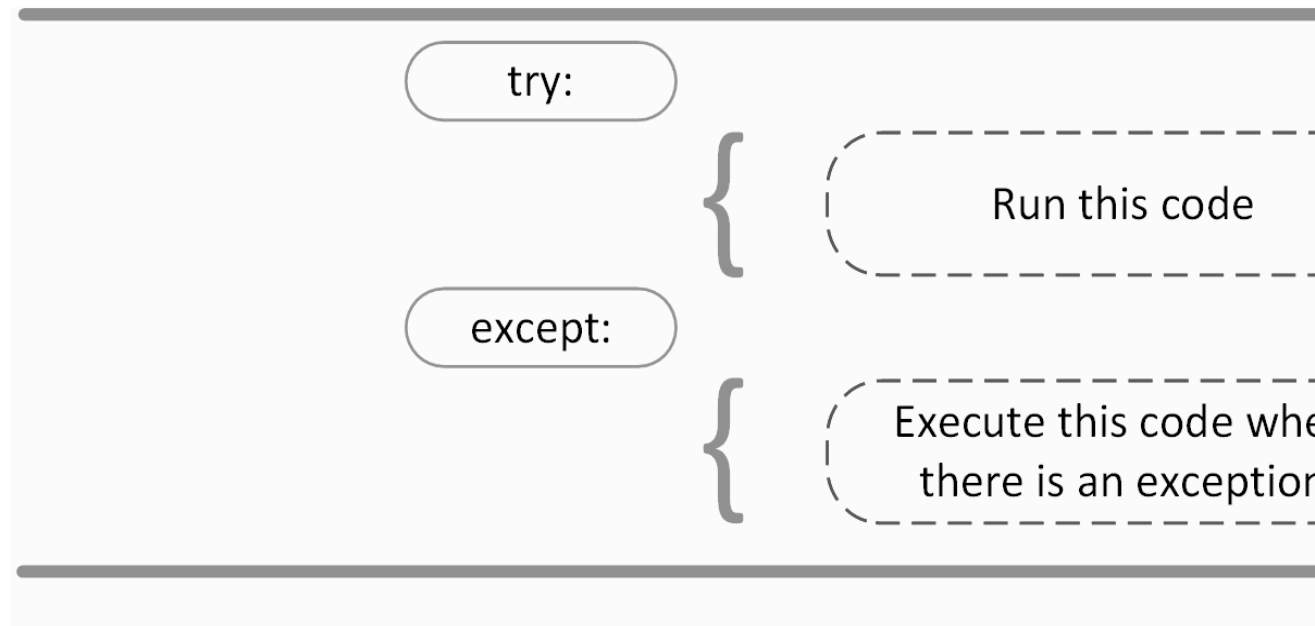
Errors & Exception Handling

A Python program terminates as soon as it encounters an error. In Python, an error can be a syntax error or an exception. In this article, you will see what an exception is and how it differs from a syntax error. After that, you will learn about raising exceptions and making assertions. Then, you'll finish with a demonstration of the try and except block



Errors & Exception Handling Exercise

Write a program to handle exception when a number is divided by zero.



SING MODULES

import module in Python

Import in python is similar to `#include header_file` in C/C++. Python modules can get access to code from another module by importing the file/function using `import`. The `import` statement is the most common way of invoking the import machinery, but it is not the only way.

```
import module_name
```

When `import` is used, it searches for the module initially in the local scope by calling `__import__()` function. The value returned by the function are then reflected in the output of the initial code.

```
Ex.  import math  
      print(math.pi)
```

Install Python Package

Set up your user environment

Install a package using pip

Install a package using its setup.py script

Regular Expression: -

Regular Expression (Regex) is a sequence of characters that defines a search pattern. For example,

- `^a...s$`

The above code defines a Regex pattern. The pattern is: any five-letter string starting with a and ending with s. A pattern defined using Regex can be used to match against a string.

Expression	String	Matched?
------------	--------	----------

<code>^a...s\$</code>	abs	No match
-----------------------	-----	----------

as		Match
----	--	-------

abyss		Match
-------	--	-------

as		No match
----	--	----------

abacus		No match
--------	--	----------

Python has a module named `re` to work with Regex. Here's an example:

```
import re
pattern = '^a...s$'
test_string = 'abyss'
result = re.match(pattern, test_string)
if result:
    print("Search successful.")
else:
    print("Search unsuccessful.")
```


MetaCharacters: -

Metacharacters are characters that are interpreted in a special way by a RegEx engine. Here's a list of metacharacters:

| . ^ \$ * + ? { } () \ |

Square brackets: Square brackets specifies a set of characters you wish to match.

Period: A period matches any single character (except newline '\n').

Caret: The caret symbol ^ is used to check if a string starts with a certain character.

Dollar: The dollar symbol \$ is used to check if a string ends with a certain character.

Star: The star symbol * matches zero or more occurrences of the pattern left to it.

Plus: The plus symbol + matches one or more occurrences of the pattern left to it.

Question Mark: The question mark symbol ? matches zero or one occurrence of the pattern left to it.

Braces: Consider this code: {n,m}. This means at least n, and at most m repetitions of the pattern left to it.

Alternation: Vertical bar | is used for alternation (or operator). Here, a|b match any string that contains either a or b

Group: Parentheses () is used to group sub-patterns. For example, (a|b|c)xz match any string that matches either a or b or c followed by xz

Pattern Matching –

Steps of Regular Expression Matching: While there are several steps to using regular expressions in Python, each step is fairly simple.

Import the regex module with `import re`.

Create a Regex object with the `re.compile()` function. (Remember to use a raw string.)

Pass the string you want to search into the Regex object's `search()` method. This returns a Match object.

Call the Match object's `group()` method to return a string of the actual matched text.

Parsing: -

When working with data, it is sometimes necessary to write a regular expression to look for properly entered values. Phone numbers in a dataset is a common field that needs to be checked for validity. Your job in this exercise is to define a regular expression to match US phone numbers that fit the pattern of xxx-xxx-xxxx.

Substitution & Complex Substitution –

The following rules for `$`-placeholders apply:

`$$` is an escape; it is replaced with a single `$`

`$identifier` names a substitution placeholder matching a mapping key of "identifier". By default, "identifier" must spell a Python identifier as defined in [2]. The first non-identifier character after the `$` character terminates this placeholder specification.

`${identifier}` is equivalent to `$identifier`. It is required when valid identifier characters follow the placeholder but are not part of the placeholder, e.g. `"${noun}ification"`.

Object Oriented Programming in Python

Python - OOP Language

Python is an “object-oriented programming language.” This means that almost all the code is implemented using a special construct called classes. Programmers use classes to keep related things together. This is done using the keyword “class,” which is a grouping of object-oriented constructs.

What is a class?

A class is a code template for creating objects. Objects have member variables and have behavior associated with them. In python a class is created by the keyword class.

An object is created using the constructor of the class. This object will then be called the instance of the class. In Python we create instance in the following manner

```
instance = class(arguments)
```

The simplest class can be created using the class keyword.

Attributes and Methods in class:

A class by itself is of no use unless there is some functionality associated with it. Functionalities are defined by setting attributes, which act as containers for data and functions related to those attributes. Those functions are called methods.

Attributes:

You can define the following class with the name Snake. This class will have an attribute name

Instance Methods in Python

Once there are attributes that “belong” to the class, you can define functions that will access the class attribute. These functions are called methods. When you define methods, you will need to always provide the first argument to the method with a self keyword.

Exercise OOP in Python

Write a program to fetch the employee details of a company and use classes and objects to display the different operations of an employee and its associated data.

Properties in Python

Python has a great concept called property which makes the life of an object oriented programmer much simpler.

Using Getters and Setters Properties –

Getters (also known as 'accessors') and setters (aka. 'mutators') are used in many object oriented programming languages to ensure the principle of data encapsulation. An obvious solution to the above constraint will be to hide the attribute temperature (make it private) and define new getter and setter interfaces to manipulate it.

What is a static method?

Static methods in Python are extremely similar to python class level methods, the difference being that a static method is bound to a class rather than the objects for that class.

This means that a static method can be called without an object for that class. This also means that static methods cannot modify the state of an object as they are not bound to it. Let's see how we can create static methods in Python.

Private Methods of a class

Private methods are those methods that should neither be accessed outside the class nor by any base class. In Python, there is no existence of Private methods that cannot be accessed except inside a class.

Inheritance in Python

What is Inheritance?

Inheritance is a powerful feature in object oriented programming. It refers to defining a new class with little or no modification to an existing class. The new class is called derived (or child) class and the one from which it inherits is called the base (or parent) class.

Python Inheritance Syntax

```
class BaseClass:
```

```
    Body of base class
```

```
class DerivedClass(BaseClass):
```

```
    Body of derived class
```

Aliasing Modules

It is possible to modify the names of modules and their functions within Python by using the `as` keyword. You may want to change a name because you have already used the same name for something else in your program another module you have imported also uses that name, or you may want to abbreviate a longer name that you are using a lot.

Syntax:

```
import [module] as [another_name]
```

Ex.

```
my_math.py  
import math as m  
print(m.pi)  
print(m.e)
```

Aliasing Modules Exercise

Create an alias of any of your module and use that alias name to call its function from your Python program.

Regular Expression Exercise

Write a program to check whether the number is from India or not. Make use of the (+91) number which will remain as a prefix number to all Indian number.

Thank You