About the Tutorial

Today, Python is one of the most popular programming languages. Although it is a general-purpose language, it is used in various areas of applications such as Machine Learning, Artificial Intelligence, web development, IoT, and more. This Python tutorial is designed to be a self-learning guide for beginners, students looking for a career in software development and Data science. This tutorial shall also be useful for experienced software professionals to enhance their skills.

This Python tutorial is based on the latest Python 3.11.2 version.

What is Python?

Python is a very popular general-purpose interpreted, interactive, object-oriented, and high-level programming language. Python is dynamically-typed and garbage-collected programming language. It was created by Guido van Rossum during 1985-1990. Like Perl, Python source code is also available under the GNU General Public License (GPL).

Python supports multiple programming paradigms, including Procedural, Object Oriented and Functional programming language. Python design philosophy emphasizes code readability with the use of significant indentation.

This tutorial gives a complete understanding of Python programming language starting from basic concepts to advanced concepts. This tutorial will take you through simple and practical approaches while learning Python Programming language.

Python Jobs

Today, Python is very high in demand and all the major companies are looking for great Python Programmers to develop websites, software components, and applications or to work with Data Science, AI, and ML technologies. When we are developing this tutorial in 2023, there is a high shortage of Python Programmers whereas market demands more number of Python Programmers due to its application in Machine Learning, Artificial Intelligence etc.

Today a Python Programmer with 3-5 years of experience is asking for around $150,000 annual package and this is the most demanding programming language in America. Though it can vary depending on the location of the Job. It's impossible to list all of the companies using Python, to name a few big companies are:

- Google
- Intel
- NASA
- PayPal
- Facebook
- IBM
- Amazon
- Netflix
- Pinterest
- Uber
- Many more...

So, you could be the next potential employee for any of these major companies. We have developed a great learning material for you to learn Python Programming which will help you prepare for the technical interviews and certification exams based on Python. So, start
learning Python using this simple and effective tutorial from anywhere and anytime absolutely at your pace.

**Why Learn Python?**

Python is consistently rated as one of the world’s most popular programming languages. Python is fairly easy to learn, so if you are starting to learn any programming language then Python could be your great choice. Today various Schools, Colleges and Universities are teaching Python as their primary programming language. There are many other good reasons which makes Python as the top choice of any programmer:

- Python is Open Source which means its available free of cost.
- Python is simple and so easy to learn
- Python is versatile and can be used to create many different things.
- Python has powerful development libraries include AI, ML etc.
- Python is much in demand and ensures high salary

**Python** is a MUST for students and working professionals to become a great Software Engineer specially when they are working in Web Development Domain. I will list down some of the key advantages of learning Python:

- **Python is Interpreted** – Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.

- **Python is Interactive** – You can actually sit at a Python prompt and interact with the interpreter directly to write your programs.

- **Python is Object-Oriented** – Python supports Object-Oriented style or technique of programming that encapsulates code within objects.

- **Python is a Beginner's Language** – Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

**Python Online Compiler/Interpreter**

We have provided [Python Online Compiler/Interpreter](#) which helps you to Edit and Execute the code directly from your browser.

You can run the following Python code online to print conventional "Hello, World!".

The following code box allows you to change the value of the code. Try to change the value inside `print()` and run it again to verify the result.

```python
# This is my first Python program.
# This will print 'Hello, World!' as the output

print("Hello, World!");
```
Careers in Python

If you know Python nicely, then you have a great career ahead. Here are just a few of the career options where Python is a key skill:

- Game developer
- Web designer
- Python developer
- Full-stack developer
- Machine learning engineer
- Data scientist
- Data analyst
- Data engineer
- DevOps engineer
- Software engineer
- Many more other roles

Characteristics of Python

Following are important characteristics of Python Programming –

- It supports functional and structured programming methods as well as OOP.
- It can be used as a scripting language or can be compiled to byte-code for building large applications.
- It provides very high-level dynamic data types and supports dynamic type checking.
- It supports automatic garbage collection.
- It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

Applications of Python

The latest release of Python is 3.x. As mentioned before, Python is one of the most widely used language over the web. I’m going to list few of them here:

- **Easy-to-learn** – Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.
- **Easy-to-read** – Python code is more clearly defined and visible to the eyes.
- **Easy-to-maintain** – Python’s source code is fairly easy-to-maintain.
- **A broad standard library** – Python’s bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.
- **Interactive Mode** – Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.
• **Portable** – Python can run on a wide variety of hardware platforms and has the same interface on all platforms.

• **Extendable** – You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.

• **Databases** – Python provides interfaces to all major commercial databases.

• **GUI Programming** – Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.

• **Scalable** – Python provides a better structure and support for large programs than shell scripting.

**Audience**

This tutorial has been prepared for the beginners to help them understand the basics to advanced concepts of Python programming language. After completing this tutorial, you will find yourself at a great level of expertise in Python programming, from where you can take yourself to the next levels.

**Prerequisites**

Although it is a beginners’ tutorial, we assume that the readers have a reasonable exposure to any programming environment and knowledge of basic concepts such as variables, commands, syntax, etc.

**Copyright & Disclaimer**

© Copyright 2023 by Tutorials Point (I) Pvt. Ltd.

All the content and graphics published in this e-book are the property of Tutorials Point (I) Pvt. Ltd. The user of this e-book is prohibited to reuse, retain, copy, distribute or republish any contents or a part of contents of this e-book in any manner without written consent of the publisher.

We strive to update the contents of our website and tutorials as timely and as precisely as possible, however, the contents may contain inaccuracies or errors. Tutorials Point (I) Pvt. Ltd. provides no guarantee regarding the accuracy, timeliness or completeness of our website or its contents including this tutorial. If you discover any errors on our website or in this tutorial, please notify us at contact@tutorialspoint.com.
Table of Contents

About the Tutorial ........................................................................................................... i
Audience ............................................................................................................................ iv
Prerequisites ..................................................................................................................... iv
Copyright & Disclaimer ................................................................................................. iv
Table of Contents ........................................................................................................... v

PYTHON BASICS ............................................................................................................. 1
1. Python – Overview ....................................................................................................... 2
2. Python – History .......................................................................................................... 4
3. Python – Features ....................................................................................................... 6
4. Python vs C++ ............................................................................................................. 9
5. Python – Hello World .............................................................................................. 12
6. Python – Application Areas ..................................................................................... 15
7. Python – Interpreter ............................................................................................... 18
8. Python – Environment Setup .................................................................................... 23
   Install Python on MacOS ........................................................................................... 26
   Install Python from Source Code ............................................................................. 27
   Python Command Line Options ................................................................................ 30
   Python Environment Variables ................................................................................ 31
9. Python – Virtual Environment ................................................................................ 34
10. Python – Basic Syntax ............................................................................................ 37
    Python Keywords ....................................................................................................... 37
    Python Identifiers ..................................................................................................... 39
    Python Indents .......................................................................................................... 40
    Python Statements ................................................................................................... 41
11. Python – Variables .................................................................................................. 43
    Naming Convention ................................................................................................... 44
20. Python – Assignment Operators ................................................................. 83

19. Python – Augmented Addition Operator (+=) ............................................. 84

  - Python – Augmented Subtraction Operator (-=) ....................................... 85
  - Python – Augmented Multiplication Operator (*=) .................................... 85
  - Python – Augmented Division Operator (/=) ............................................ 86
  - Python – Augmented Modulus Operator (%=) ........................................... 87
  - Python – Augmented Exponent Operator (**=) ......................................... 88
  - Python – Augmented Floor division Operator (//=) ................................... 89

20. Python – Comparison Operators .................................................................. 90

21. Python – Logical Operators ......................................................................... 96

22. Python – Bitwise Operators .......................................................................... 100

  - Python – Bitwise AND Operator (&) ......................................................... 100
  - Python – Bitwise OR Operator (|) ............................................................. 101
  - Python – Binary XOR Operator (^) ......................................................... 102
  - Python – Binary NOT Operator (~) ......................................................... 102
  - Python – Left Shift Operator (<<) ............................................................ 103
  - Python – Right Shift Operator (>>) ......................................................... 103

23. Python – Membership Operators .................................................................. 105


25. Python – Comments ..................................................................................... 110
55. **Python – Escape Characters** ................................................................. 236

56. **Python – String Methods** .................................................................... 240
  Case Conversion Methods ........................................................................ 240
  Alignment Methods ................................................................................. 246
  Find and Replace Methods ....................................................................... 252
  Boolean String Methods .......................................................................... 260
  Split and Join Methods ........................................................................... 274
  Translation Methods ............................................................................... 285

57. **Python – String Exercises** .................................................................. 289

**PYTHON LISTS** ....................................................................................... 291

58. **Python – Lists** .................................................................................... 292
  Python – List Operations ......................................................................... 292

59. **Python – Access List Items** ................................................................. 293

60. **Python – Change List Items** ................................................................. 295

61. **Python – Add List Items** ....................................................................... 297

62. **Python – Remove List Items** ................................................................. 298
  Using the remove() Method ..................................................................... 298
  Using the pop() Method .......................................................................... 298
  Using the "del" Keyword ......................................................................... 298

63. **Python – Loop Lists** ............................................................................ 300

64. **Python – List Comprehension** ............................................................. 301
  List Comprehension Technique ................................................................. 301
  Nested Loops in List Comprehension ....................................................... 302
  Condition in List Comprehension ............................................................ 302

65. **Python – Sort Lists** ............................................................................... 303

66. **Python – Copy Lists** ............................................................................ 305
67. Python – Join Lists ................................................................. 307

68. Python – List Methods .......................................................... 309
   Python – Clear a List .............................................................. 309
   Python – Counting the List Items .......................................... 310
   Python – Extending a List ...................................................... 311
   Python – Finding the Index of a List Item ............................... 311
   Python – Reverse a List ....................................................... 312

69. Python – List Exercises ....................................................... 313

PYTHON TUPLES ........................................................................ 315

70. Python – Tuples ................................................................. 316
    Python – Tuple Operations .................................................. 316

71. Python – Access Tuple Items ................................................ 317
    Extracting a Subtuple from a Tuple ...................................... 317

72. Python – Update Tuples ........................................................ 319
    How to Update a Python Tuple? ............................................ 319

73. Python – Unpack Tuple Items ................................................. 321
    Using "*" to Unpack a Tuple .................................................. 321

74. Python – Loop Tuples .......................................................... 323

75. Python – Join Tuples ............................................................ 324

76. Python – Tuple Methods ........................................................ 326
    Finding the Index of a Tuple Item .......................................... 326
    Counting Tuple Items ......................................................... 326

77. Python – Tuple Exercises ..................................................... 328

PYTHON SETS ......................................................................... 330

78. Python – Sets .................................................................... 331
    set() Function ..................................................................... 331

79. Python – Access Set Items .................................................... 334
<table>
<thead>
<tr>
<th>Python – Add Set Items</th>
<th>335</th>
</tr>
</thead>
<tbody>
<tr>
<td>add() Method</td>
<td>335</td>
</tr>
<tr>
<td>update() Method</td>
<td>335</td>
</tr>
<tr>
<td>union() Method</td>
<td>336</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Python – Remove Set Items</th>
<th>338</th>
</tr>
</thead>
<tbody>
<tr>
<td>remove() Method</td>
<td>338</td>
</tr>
<tr>
<td>discard() Method</td>
<td>338</td>
</tr>
<tr>
<td>pop() Method</td>
<td>339</td>
</tr>
<tr>
<td>clear() Method</td>
<td>340</td>
</tr>
<tr>
<td>difference_update() Method</td>
<td>340</td>
</tr>
<tr>
<td>difference() Method</td>
<td>341</td>
</tr>
<tr>
<td>intersection_update() Method</td>
<td>341</td>
</tr>
<tr>
<td>intersection() Method</td>
<td>342</td>
</tr>
<tr>
<td>symmetric_difference_update() method</td>
<td>343</td>
</tr>
<tr>
<td>symmetric_difference() Method</td>
<td>343</td>
</tr>
</tbody>
</table>

| Python – Loop Sets | 345 |

<table>
<thead>
<tr>
<th>Python – Join Sets</th>
<th>346</th>
</tr>
</thead>
<tbody>
<tr>
<td>Using the &quot;</td>
<td>&quot; Operator</td>
</tr>
<tr>
<td>Using the union() Method</td>
<td>346</td>
</tr>
<tr>
<td>Using the update() Method</td>
<td>346</td>
</tr>
<tr>
<td>Using the unpacking Operator</td>
<td>347</td>
</tr>
</tbody>
</table>

| Python – Copy Sets | 348 |

<table>
<thead>
<tr>
<th>Python – Set Operators</th>
<th>349</th>
</tr>
</thead>
<tbody>
<tr>
<td>Union Operator (</td>
<td>)</td>
</tr>
<tr>
<td>Intersection Operator (&amp;)</td>
<td>349</td>
</tr>
<tr>
<td>Difference Operator (–)</td>
<td>350</td>
</tr>
<tr>
<td>Symmetric Difference Operator (^)</td>
<td>351</td>
</tr>
</tbody>
</table>

| Python – Set Methods | 353 |
87. Python – Set Exercises ................................................................................................................354

PYTHON DICTIONARIES ...........................................................................................................356

88. Python – Dictionaries ................................................................................................................357

Python – Dictionary Operators ..................................................................................................358

89. Python – Access Dictionary Items ........................................................................................359

Using the "[ ]" Operator ...........................................................................................................359
Using the get() Method ..............................................................................................................359

90. Python – Change Dictionary Items ........................................................................................361

91. Python – Add Dictionary Items .............................................................................................363

Using the "[ ]" Operator ...........................................................................................................363
Using the update() Method .......................................................................................................364
Using the Unpack Operator .......................................................................................................366
Using the Union Operator (|) .................................................................................................366
Using "|=" Operator ..................................................................................................................367

92. Python – Remove Dictionary Items .......................................................................................368

Using del Keyword ..................................................................................................................368
Using pop() Method .................................................................................................................369
Using popitem() Method ..........................................................................................................369
Using clear() Method ..............................................................................................................370

93. Python – Dictionary View Objects .......................................................................................371

items() Method .......................................................................................................................371
keys() Method ..........................................................................................................................372
values() Method ......................................................................................................................372

94. Python – Loop Dictionaries ...................................................................................................374

95. Python – Copy Dictionaries ...................................................................................................377

96. Python – Nested Dictionaries ..............................................................................................379

97. Python – Dictionary Methods ..............................................................................................381

Dictionary clear() Method .......................................................................................................381
107. **Python – Join Arrays** ................................................................. 407

108. **Python – Array Methods** ........................................................... 409
    array.reverse() Method .......................................................... 409
    array.count() Method ............................................................ 409
    array.index() method ........................................................... 410
    array.fromlist() Method ......................................................... 410
    array.tofile() Method .......................................................... 411
    array.fromfile() Method ....................................................... 411

109. **Python – Array Exercises** ...................................................... 413

**PYTHON FILE HANDLING** .................................................................. 415

110. **Python – File Handling** ............................................................ 416
    The open() Function ................................................................ 416
    File Opening Modes ............................................................... 417

111. **Python – Write to File** ............................................................. 419
    Writing in Binary Mode .......................................................... 419
   Appending to a File ................................................................ 420
    Using the "w+" Mode .............................................................. 420

112. **Python – Read Files** .................................................................. 422
    Reading in Binary Mode .......................................................... 423
    Read Integer Data from File .................................................... 423
    Read Float Data from File ....................................................... 423
    Using the "r+" Mode ............................................................... 424
    Python – Simultaneous Read/Write .......................................... 425
    The seek() Method ............................................................... 425

113. **Python – Renaming and Deleting Files** ...................................... 427
    rename() Method ................................................................. 427
    remove() Method ................................................................. 427

114. **Python – Directories** ............................................................... 428

xvi
## Python – File Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>close()</code> Method</td>
<td>431</td>
</tr>
<tr>
<td><code>flush()</code> Method</td>
<td>431</td>
</tr>
<tr>
<td><code>fileno()</code> Method</td>
<td>432</td>
</tr>
<tr>
<td><code>isatty()</code> Method</td>
<td>433</td>
</tr>
<tr>
<td><code>next()</code> function</td>
<td>434</td>
</tr>
<tr>
<td><code>read()</code> Method</td>
<td>435</td>
</tr>
<tr>
<td><code>readline()</code> Method</td>
<td>436</td>
</tr>
<tr>
<td><code>readlines()</code> Method</td>
<td>437</td>
</tr>
<tr>
<td><code>seek()</code> Method</td>
<td>438</td>
</tr>
<tr>
<td><code>tell()</code> Method</td>
<td>439</td>
</tr>
<tr>
<td><code>truncate()</code> Method</td>
<td>440</td>
</tr>
<tr>
<td><code>write()</code> Method</td>
<td>441</td>
</tr>
<tr>
<td><code>writelines()</code> Method</td>
<td>443</td>
</tr>
</tbody>
</table>

## Python – OS File/Directory Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>os.closerange()</code> Method</td>
<td>446</td>
</tr>
<tr>
<td><code>os.dup()</code> Method</td>
<td>447</td>
</tr>
<tr>
<td><code>os.fdatasync()</code> Method</td>
<td>448</td>
</tr>
<tr>
<td><code>os.fdopen()</code> Method</td>
<td>450</td>
</tr>
<tr>
<td><code>os.fsync()</code> Method</td>
<td>451</td>
</tr>
<tr>
<td><code>os.ftruncate()</code> Method</td>
<td>452</td>
</tr>
<tr>
<td><code>os.lseek()</code> Method</td>
<td>454</td>
</tr>
<tr>
<td><code>os.open()</code> Method</td>
<td>455</td>
</tr>
<tr>
<td><code>os.read()</code> Method</td>
<td>456</td>
</tr>
</tbody>
</table>
os.write() Method..................................................................................................................457

PYTHOON OBJECT ORIENTED PROGRAMMING................................................................459

117. Python – OOP Concepts .................................................................................................460
    Procedural Oriented Approach .........................................................................................460

118. Python – Object and Classes .......................................................................................464

119. Python – Class Attributes ............................................................................................466
    Class Variables ..............................................................................................................467

120. Python – Class Methods ...............................................................................................468

121. Python – Static Methods .............................................................................................470

122. Python – Constructors ..................................................................................................471
    Python – Instance Methods ..........................................................................................473

123. Python – Access Modifiers ..........................................................................................475
    Name Mangling ..............................................................................................................477

124. Python – Inheritance ...................................................................................................480
    Python – Multiple Inheritance .......................................................................................481
    Method Resolution Order (MRO) ...................................................................................483

125. Python – Polymorphism ...............................................................................................484

126. Python – Method Overriding ......................................................................................486
    Base Overridable Methods .............................................................................................488

127. Python – Method Overloading ....................................................................................489

128. Python – Dynamic Binding ..........................................................................................492
    Duck Typing ..................................................................................................................493

129. Python – Dynamic Typing ...........................................................................................495

130. Python – Abstraction ....................................................................................................497

131. Python – Encapsulation .................................................................................................499
132. Python – Interfaces .................................................................................................................. 502
133. Python – Packages ..................................................................................................................... 504
  Define Package List ......................................................................................................................... 505
  Package Installation ......................................................................................................................... 506
134. Python – Inner Classes .............................................................................................................. 508
135. Python – Anonymous Class and Objects .................................................................................... 510
136. Python – Singleton Class ........................................................................................................... 512
137. Python – Wrapper Classes ......................................................................................................... 513
138. Python – Enums ......................................................................................................................... 514
139. Python – Reflection .................................................................................................................... 517
  The type() Function ......................................................................................................................... 517
  The isinstance() Function ............................................................................................................. 518
  The issubclass() Function ............................................................................................................. 519
  The callable() Function ................................................................................................................. 519
  The getattr() Function ................................................................................................................... 520
  The setattr() Function ................................................................................................................... 521
  The hasattr() Function .................................................................................................................. 521
  The dir() Function ......................................................................................................................... 522

PYTHON ERRORS AND EXCEPTIONS .............................................................................................. 524
140. Python – Syntax Errors ............................................................................................................. 525
  What is Exception? .......................................................................................................................... 526
141. Python – Exceptions Handling .................................................................................................. 528
142. Python – The try-except Block .................................................................................................. 530
143. Python – The try-finally Block .................................................................................................. 531
  Exception with Arguments ............................................................................................................. 532
144. Python – Raising Exceptions ..................................................................................................... 533
145. Python – Exception Chaining ..................................................................................................... 535
146. Python – Nested try Block ........................................................................................................ 538
PYTHON SYNCHRONIZATION..................................................................................................................578

163. Python – Inter-Thread Communication ..........................................................................................579
    The Event Object ...............................................................................................................................579
    The Condition Object .......................................................................................................................581

164. Python – Thread Deadlock .............................................................................................................584
    The Lock Object ...............................................................................................................................584
    The Semaphore Object ....................................................................................................................586

165. Python – Interrupting a Thread ......................................................................................................589

PYTHON NETWORKING..........................................................................................................................591

166. Python – Network Programming ..................................................................................................592

167. Python – Socket Programming .....................................................................................................594
    What are Sockets? ..............................................................................................................................594
    Python – The socket Module ............................................................................................................595
    Server Socket Methods ....................................................................................................................595
    Client Socket Methods .....................................................................................................................596
    Python – Socket Server ....................................................................................................................596
    Python – Socket Client .....................................................................................................................597
    Python – File Transfer with Socket Module ....................................................................................599
    Python – The socketserver Module ..................................................................................................601

168. Python – URL Processing ................................................................................................................604
    The urllib.parse Module ..................................................................................................................604
    The urllib.request Module .............................................................................................................606
    The Request Object ..........................................................................................................................607
    The urllib.error Module ..................................................................................................................609

169. Python – Generics ..........................................................................................................................610

PYTHON MISCELLANEOUS......................................................................................................................612

170. Python – Date and Time ..................................................................................................................613
171. **Python – Maths** .......................................................................................................................... 647

Functions in Math Module .................................................................................................................. 647
Theoretic and Representation Functions ......................................................................................... 650
Power and Logarithmic Functions .................................................................................................. 676
<table>
<thead>
<tr>
<th>172. Python – Iterators</th>
<th>.......................................................... 713</th>
</tr>
</thead>
<tbody>
<tr>
<td>Asynchronous Iterator</td>
<td>.................................................................. 716</td>
</tr>
<tr>
<td>173. Python – Generators</td>
<td>.......................................................... 718</td>
</tr>
<tr>
<td>Asynchronous Generator</td>
<td>.................................................................. 720</td>
</tr>
<tr>
<td>174. Python – Closures</td>
<td>.......................................................... 723</td>
</tr>
<tr>
<td>Nested Functions</td>
<td>.................................................................. 723</td>
</tr>
<tr>
<td>What is a Closure?</td>
<td>.................................................................. 724</td>
</tr>
<tr>
<td>nonlocal Keyword</td>
<td>.................................................................. 724</td>
</tr>
<tr>
<td>175. Python – Decorators</td>
<td>.......................................................... 726</td>
</tr>
<tr>
<td>@classmethod Decorator</td>
<td>.................................................................. 728</td>
</tr>
<tr>
<td>@staticmethod Decorator</td>
<td>.................................................................. 729</td>
</tr>
<tr>
<td>@property Decorator</td>
<td>.................................................................. 730</td>
</tr>
<tr>
<td>176. Python – Recursion</td>
<td>.......................................................... 732</td>
</tr>
<tr>
<td>177. Python – Regular Expressions</td>
<td>.......... 735</td>
</tr>
<tr>
<td>Raw Strings</td>
<td>.................................................................. 735</td>
</tr>
<tr>
<td>Metacharacters</td>
<td>.................................................................. 736</td>
</tr>
<tr>
<td>re.match() Function</td>
<td>.................................................................. 737</td>
</tr>
<tr>
<td>re.search() Function</td>
<td>.................................................................. 738</td>
</tr>
<tr>
<td>Matching Vs Searching</td>
<td>.................................................................. 739</td>
</tr>
<tr>
<td>re.findall() Function</td>
<td>.................................................................. 740</td>
</tr>
<tr>
<td>re.sub() Function</td>
<td>.................................................................. 741</td>
</tr>
<tr>
<td>re.compile() Function</td>
<td>.................................................................. 742</td>
</tr>
<tr>
<td>re.initer() Function</td>
<td>.................................................................. 743</td>
</tr>
<tr>
<td>178. Python – PIP</td>
<td>745</td>
</tr>
<tr>
<td>------------------</td>
<td>-----</td>
</tr>
<tr>
<td>Install a Package</td>
<td>745</td>
</tr>
<tr>
<td>Use requirements.txt</td>
<td>747</td>
</tr>
<tr>
<td>pip uninstall</td>
<td>747</td>
</tr>
<tr>
<td>pip list</td>
<td>748</td>
</tr>
<tr>
<td>pip show</td>
<td>749</td>
</tr>
<tr>
<td>pip freeze</td>
<td>749</td>
</tr>
<tr>
<td>pip download</td>
<td>749</td>
</tr>
<tr>
<td>pip search</td>
<td>750</td>
</tr>
<tr>
<td>pip config</td>
<td>750</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>179. Python – Database Access</th>
<th>751</th>
</tr>
</thead>
<tbody>
<tr>
<td>The sqlite3 Module</td>
<td>752</td>
</tr>
<tr>
<td>The Connection Object</td>
<td>752</td>
</tr>
<tr>
<td>The Cursor Object</td>
<td>753</td>
</tr>
<tr>
<td>Creating a Database Table</td>
<td>753</td>
</tr>
<tr>
<td>INSERT Operation</td>
<td>754</td>
</tr>
<tr>
<td>READ Operation</td>
<td>755</td>
</tr>
<tr>
<td>Update Operation</td>
<td>756</td>
</tr>
<tr>
<td>DELETE Operation</td>
<td>757</td>
</tr>
<tr>
<td>Performing Transactions</td>
<td>757</td>
</tr>
<tr>
<td>COMMIT Operation</td>
<td>758</td>
</tr>
<tr>
<td>ROLLBACK Operation</td>
<td>758</td>
</tr>
<tr>
<td>The PyMySQL Module</td>
<td>759</td>
</tr>
<tr>
<td>Handling Errors</td>
<td>760</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>180. Python – Weak References</th>
<th>761</th>
</tr>
</thead>
<tbody>
<tr>
<td>The callback Function</td>
<td>762</td>
</tr>
<tr>
<td>Finalizing Objects</td>
<td>763</td>
</tr>
</tbody>
</table>
181. Python – Serialization ................................................................. 768
   Pickle Protocols ........................................................................ 768
   dump() and load() ...................................................................... 768
   dumps() and loads() ................................................................... 769
   Pickler Class ............................................................................ 770
   Unpickler Class ......................................................................... 770

182. Python – Templating ................................................................. 771
   substitute() ............................................................................ 771
   safe_substitute() ....................................................................... 772
   is_valid() ................................................................................ 773
   get_identifiers() ....................................................................... 773

183. Python – Output Formatting ..................................................... 775
   String Formatting Operator ....................................................... 775
   The format() Method .................................................................. 775
   f-strings .................................................................................. 776
   Template Strings ......................................................................... 776
   The textwrap Module .................................................................. 777
   The shorten() Function ............................................................. 779
   The pprint Module ..................................................................... 780

184. Python – Performance Measurement ....................................... 784

185. Python – Data Compression ..................................................... 787
   The zlib Module ......................................................................... 787
   The gzip Module ......................................................................... 788
   The bz2 Module ......................................................................... 790
   The lzma Module ....................................................................... 791
   The zipfile Module .................................................................... 793
### 186. Python – CGI Programming

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>What is CGI?</td>
<td>799</td>
</tr>
<tr>
<td>Web Browsing</td>
<td>799</td>
</tr>
<tr>
<td>CGI Architecture Diagram</td>
<td>800</td>
</tr>
<tr>
<td>Web Server Support and Configuration</td>
<td>800</td>
</tr>
<tr>
<td>First CGI Program</td>
<td>801</td>
</tr>
<tr>
<td>HTTP Header</td>
<td>802</td>
</tr>
<tr>
<td>CGI Environment Variables</td>
<td>802</td>
</tr>
<tr>
<td>GET and POST Methods</td>
<td>804</td>
</tr>
<tr>
<td>Passing Information using GET method</td>
<td>804</td>
</tr>
<tr>
<td>Simple URL Example – Get Method</td>
<td>804</td>
</tr>
<tr>
<td>Simple FORM Example – GET Method</td>
<td>805</td>
</tr>
<tr>
<td>Passing Radio Button Data to CGI Program</td>
<td>808</td>
</tr>
<tr>
<td>Passing Text Area Data to CGI Program</td>
<td>809</td>
</tr>
<tr>
<td>Passing Drop Down Box Data to CGI Program</td>
<td>810</td>
</tr>
<tr>
<td>Using Cookies in CGI</td>
<td>811</td>
</tr>
<tr>
<td>How It Works?</td>
<td>811</td>
</tr>
<tr>
<td>Setting up Cookies</td>
<td>811</td>
</tr>
<tr>
<td>Retrieving Cookies</td>
<td>812</td>
</tr>
<tr>
<td>File Upload Example</td>
<td>813</td>
</tr>
<tr>
<td>How To Raise a &quot;File Download&quot; Dialog Box</td>
<td>814</td>
</tr>
</tbody>
</table>

### 187. Python – XML Processing

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>What is XML?</td>
<td>815</td>
</tr>
<tr>
<td>Parsing XML with SAX APIs</td>
<td>817</td>
</tr>
<tr>
<td>Parsing XML with DOM APIs</td>
<td>820</td>
</tr>
<tr>
<td>ElementTree XML API</td>
<td>822</td>
</tr>
</tbody>
</table>
188. Python – GUI Programming ................................................................. 826

Tkinter Programming .............................................................................. 826
Tkinter Widgets ...................................................................................... 828
Tkinter Button ....................................................................................... 829
Tkinter Canvas ....................................................................................... 832
Tkinter Checkbutton ............................................................................... 835
Tkinter Entry ........................................................................................... 838
Tkinter Frame .......................................................................................... 842
Tkinter Label ............................................................................................ 844
Tkinter Listbox ........................................................................................ 847
Tkinter Menubutton .................................................................................. 851
Tkinter Menu ............................................................................................ 854
Tkinter Message ..................................................................................... 858
Tkinter Radiobutton ................................................................................. 860
Tkinter Scale ............................................................................................ 864
Tkinter Scrollbar ..................................................................................... 868
Tkinter Text ............................................................................................... 871
Tkinter Toplevel ...................................................................................... 876
Tkinter Spinbox ....................................................................................... 879
Tkinter PanedWindow .............................................................................. 882
Tkinter LabelFrame .................................................................................. 885
Tkinter tkMessageBox ............................................................................. 887
Standard Attributes ............................................................................. 888
Tkinter Dimensions ............................................................................... 888
Tkinter Colors ........................................................................................ 889
Python Basics
Python is a high-level, multi-paradigm programming language. As Python is an interpreter-based language, it is easier to learn compared to some of the other mainstream languages. Python is a dynamically typed language with very intuitive data types.

Python is an open-source and cross-platform programming language. It is available for use under **Python Software Foundation License** (compatible to GNU General Public License) on all the major operating system platforms Linux, Windows and Mac OS.

The design philosophy of Python emphasizes on simplicity, readability and unambiguity. Python is known for its batteries included approach as Python software is distributed with a comprehensive standard library of functions and modules.

Python's design philosophy is documented in the **Zen of Python**. It consists of nineteen aphorisms such as:

- Beautiful is better than ugly
- Explicit is better than implicit
- Simple is better than complex
- Complex is better than complicated

To obtain the complete Zen of Python document, type `import this` in the Python shell:

```
>>> import this

The Zen of Python, by Tim Peters

Beautiful is better than ugly.
Explicit is better than implicit.
Simple is better than complex.
Complex is better than complicated.
Flat is better than nested.
Sparse is better than dense.
Readability counts.
Special cases aren't special enough to break the rules.
Although practicality beats purity.
Errors should never pass silently.
Unless explicitly silenced.
In the face of ambiguity, refuse the temptation to guess.
There should be one-- and preferably only one --obvious way to do it.
Although that way may not be obvious at first unless you're Dutch.
```
Now is better than never.
Although never is often better than *right* now.
If the implementation is hard to explain, it's a bad idea.
If the implementation is easy to explain, it may be a good idea.
Namespaces are one honking great idea -- let's do more of those!

Python supports imperative, structured as well as object-oriented programming methodology. It provides features of functional programming as well.
2. Python – History

Guido Van Rossum, a Dutch programmer, created Python programming language. In the late 80's, he had been working on the development of ABC language in a computer science research institute named Centrum Wiskunde & Informatica (CWI) in the Netherlands. In 1991, Van Rossum conceived and published Python as a successor of ABC language.

For many uninitiated people, the word Python is related to a species of snake. Rossum though attributes the choice of the name Python to a popular comedy series "Monty Python's Flying Circus" on BBC.

Being the principal architect of Python, the developer community conferred upon him the title of "Benevolent Dictator for Life (BDFL). However, in 2018, Rossum relinquished the title. Thereafter, the development and distribution of the reference implementation of Python is handled by a nonprofit organization Python Software Foundation.

Important stages in the history of Python:

Python 0.9.0
Python's first published version is 0.9. It was released in February 1991. It consisted of support for core object-oriented programming principles.

Python 1.0
In January 1994, version 1.0 was released, armed with functional programming tools, features like support for complex numbers etc.

Python 2.0
Next major version – Python 2.0 was launched in October 2000. Many new features such as list comprehension, garbage collection and Unicode support were included with it.

Python 3.0
Python 3.0, a completely revamped version of Python was released in December 2008. The primary objective of this revamp was to remove a lot of discrepancies that had crept in Python 2.x versions. Python 3 was backported to Python 2.6. It also included a utility named as python2to3 to facilitate automatic translation of Python 2 code to Python 3.

EOL for Python 2.x
Even after the release of Python 3, Python Software Foundation continued to support the Python 2 branch with incremental micro versions till 2019. However, it decided to discontinue the support by the end of year 2020, at which time Python 2.7.17 was the last version in the branch.
Current Version
Meanwhile, more and more features have been incorporated into Python’s 3.x branch. As of date, Python 3.11.2 is the current stable version, released in February 2023.

What’s New in Python 3.11?
One of the most important features of Python’s version 3.11 is the significant improvement in speed. According to Python’s official documentation, this version is faster than the previous version (3.10) by up to 60%. It also states that the standard benchmark suite shows a 25% faster execution rate.

- Python 3.11 has a better exception messaging. Instead of generating a long traceback on the occurrence of an exception, we now get the exact expression causing the error.

- As per the recommendations of PEP 678, the add_note() method is added to the BaseException class. You can call this method inside the except clause and pass a custom error message.

- It also adds the cbroot() function in the maths module. It returns the cube root of a given number.

- A new module tomllib is added in the standard library. TOML (Tom’s Obvious Minimal Language) can be parsed with tomllib module function.
In this chapter, let's highlight some of the important features of Python that make it widely popular.

**Python is Easy to Learn**

This is one of the most important reasons for the popularity of Python. Python has a limited set of keywords. Its features such as simple syntax, usage of indentation to avoid clutter of curly brackets and dynamic typing that doesn't necessitate prior declaration of variable help a beginner to learn Python quickly and easily.

**Python is Interpreter Based**

Instructions in any programming languages must be translated into machine code for the processor to execute them. Programming languages are either compiler based or interpreter based.

In case of a compiler, a machine language version of the entire source program is generated. The conversion fails even if there is a single erroneous statement. Hence, the development process is tedious for the beginners. The C family languages (including C, C++, Java, C Sharp etc) are compiler based.

Python is an interpreter based language. The interpreter takes one instruction from the source code at a time, translates it into machine code and executes it. Instructions before the first occurrence of error are executed. With this feature, it is easier to debug the program and thus proves useful for the beginner level programmer to gain confidence gradually. Python therefore is a beginner-friendly language.

**Python is Interactive**

Standard Python distribution comes with an interactive shell that works on the principle of REPL (Read – Evaluate – Print – Loop). The shell presents a Python prompt `>>>`. You can type any valid Python expression and press Enter. Python interpreter immediately returns the response and the prompt comes back to read the next expression.

```
>>> 2*3+1
7

>>> print("Hello World")
Hello World
```

The interactive mode is especially useful to get familiar with a library and test out its functionality. You can try out small code snippets in interactive mode before writing a program.
Python is Multi-Paradigm

Python is a completely object-oriented language. Everything in a Python program is an object. However, Python conveniently encapsulates its object orientation to be used as an imperative or procedural language – such as C. Python also provides certain functionality that resembles functional programming. Moreover, certain third-party tools have been developed to support other programming paradigms such as aspect-oriented and logic programming.

Python’s Standard Library

Even though it has a very few keywords (only Thirty Five), Python software is distributed with a standard library made of large number of modules and packages. Thus Python has out of box support for programming needs such as serialization, data compression, internet data handling, and many more. Python is known for its batteries included approach.

Python is Open-Source and Cross-Platform

Python’s standard distribution can be downloaded from https://www.python.org/downloads/ without any restrictions. You can download pre-compiled binaries for various operating system platforms. In addition, the source code is also freely available, which is why it comes under open source category.

Python software (along with the documentation) is distributed under Python Software Foundation License. It is a BSD style permissive software license and compatible to GNU GPL (General Public License).

Python is a cross-platform language. Pre-compiled binaries are available for use on various operating system platforms such as Windows, Linux, Mac OS, Android OS. The reference implementation of Python is called CPython and is written in C. You can download the source code and compile it for your OS platform.

A Python program is first compiled to an intermediate platform independent byte code. The virtual machine inside the interpreter then executes the byte code. This behaviour
makes Python a cross-platform language, and thus a Python program can be easily ported from one OS platform to other.

**Python for GUI Applications**

Python's standard distribution has an excellent graphics library called Tkinter. It is a Python port for the vastly popular GUI toolkit called TCL/Tk. You can build attractive user-friendly GUI applications in Python. GUI toolkits are generally written in C/C++. Many of them have been ported to Python. Examples are PyQt, WxWidgets, PySimpleGUI etc.

**Python’s Database Connectivity**

Almost any type of database can be used as a backend with the Python application. DB-API is a set of specifications for database driver software to let Python communicate with a relational database. With many third party libraries, Python can also work with NoSQL databases such as MongoDB.

**Python is Extensible**

The term extensibility implies the ability to add new features or modify existing features. As stated earlier, CPython (which is Python's reference implementation) is written in C. Hence one can easily write modules/libraries in C and incorporate them in the standard library. There are other implementations of Python such as Jython (written in Java) and IPython (written in C#). Hence, it is possible to write and merge new functionality in these implementations with Java and C# respectively.

**Python’s Active Developer Community**

As a result of Python's popularity and open-source nature, a large number of Python developers often interact with online forums and conferences. Python Software Foundation also has a significant member base, involved in the organization's mission to "promote, protect, and advance the Python programming language"

Python also enjoys a significant institutional support. Major IT companies Google, Microsoft, and Meta contribute immensely by preparing documentation and other resources.
Both Python and C++ are among the most popular programming languages. Both of them have their advantages and disadvantages. In this chapter, we shall take a look at their characteristic features.

**Compiled vs Interpreted**

Like C, C++ is also a compiler-based language. A compiler translates the entire code in a machine language code specific to the operating system in use and processor architecture.

Python is interpreter-based language. The interpreter executes the source code line by line.

**Cross-platform**

When a C++ source code such as hello.cpp is compiled on Linux, it can be only run on any other computer with Linux operating system. If required to run on other OS, it needs to be compiled.

Python interpreter doesn't produce compiled code. Source code is converted to byte code every time it is run on any operating system without any changes or additional steps.

**Portability**

Python code is easily portable from one OS to other. C++ code is not portable as it must be recompiled if the OS changes.

**Speed of Development**

C++ program is compiled to the machine code. Hence, its execution is faster than interpreter based language.

Python interpreter doesn't generate the machine code. Conversion of intermediate byte code to machine language is done on each execution of program.

If a program is to be used frequently, C++ is more efficient than Python.

**Easy to Learn**

Compared to C++, Python has a simpler syntax. Its code is more readable. Writing C++ code seems daunting in the beginning because of complicated syntax rule such as use of curly braces and semicolon for sentence termination.

Python doesn't use curly brackets for marking a block of statements. Instead, it uses indents. Statements of similar indent level mark a block. This makes a Python program more readable.
Static vs Dynamic Typing

C++ is a statically typed language. The type of variables for storing data need to be declared in the beginning. Undeclared variables can't be used. Once a variable is declared to be of a certain type, value of only that type can be stored in it.

Python is a dynamically typed language. It doesn't require a variable to be declared before assigning it a value. Since, a variable may store any type of data, it is called dynamically typed.

OOP Concepts

Both C++ and Python implement object oriented programming concepts. C++ is closer to the theory of OOP than Python. C++ supports the concept of data encapsulation as the visibility of the variables can be defined as public, private and protected.

Python doesn't have the provision of defining the visibility. Unlike C++, Python doesn't support method overloading. Because it is dynamically typed, all the methods are polymorphic in nature by default.

C++ is in fact an extension of C. One can say that additional keywords are added in C so that it supports OOP. Hence, we can write a C type procedure oriented program in C++.

Python is completely object oriented language. Python's data model is such that, even if you can adapt a procedure oriented approach, Python internally uses object-oriented methodology.

Garbage Collection

C++ uses the concept of pointers. Unused memory in a C++ program is not cleared automatically. In C++, the process of garbage collection is manual. Hence, a C++ program is likely to face memory related exceptional behavior.

Python has a mechanism of automatic garbage collection. Hence, Python program is more robust and less prone to memory related issues.

Application Areas

Because C++ program compiles directly to machine code, it is more suitable for systems programming, writing device drivers, embedded systems and operating system utilities.

Python program is suitable for application programming. Its main area of application today is data science, machine learning, API development etc.

The following table summarizes the comparison between C++ and Python:

<table>
<thead>
<tr>
<th>Criteria</th>
<th>C++</th>
<th>Python</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution</td>
<td>Compiler based</td>
<td>Interpreter based</td>
</tr>
<tr>
<td>Typing</td>
<td>Static typing</td>
<td>Dynamic typing</td>
</tr>
<tr>
<td>Portability</td>
<td>Not portable</td>
<td>Highly portable</td>
</tr>
<tr>
<td>Garbage collection</td>
<td>Manual</td>
<td>Automatic</td>
</tr>
<tr>
<td></td>
<td>Tedious</td>
<td>Simple</td>
</tr>
<tr>
<td>----------------------</td>
<td>----------------------------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>Performance</td>
<td>Faster execution</td>
<td>Slower execution</td>
</tr>
<tr>
<td>Application areas</td>
<td>Embedded systems, device drivers</td>
<td>Machine learning, web applications</td>
</tr>
</tbody>
</table>
Hello World program is a basic computer code written in a general purpose programming language, used as a test program. It doesn't ask for any input and displays a Hello World message on the output console. It is used to test if the software needed to compile and run the program has been installed correctly.

It is very easy to display the Hello World message using the Python interpreter. Launch the interpreter from a command terminal of your operating system and issue the print statement from the Python prompt as follows:

```
PS C:\Users\mlath> python
Python 3.11.2 (tags/v3.11.2:878ead1, Feb 7 2023, 16:38:35) [MSC v.1934 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> print ("Hello World")
Hello World
```

Similarly, Hello World message is printed in Linux.

```
mvl@GNVBGL3:~$ python3
Python 3.10.6 (main, Mar 10 2023, 10:55:28) [GCC 11.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> print ("Hello World")
Hello World
```

Python interpreter also works in scripted mode. Open any text editor, enter the following text and save as Hello.py

```
print ("Hello World")
```

For Windows OS, open the command prompt terminal (CMD) and run the program as shown below:

```
C:\Python311>python hello.py
Hello World
```

The terminal shows the Hello World message.
While working on Ubuntu Linux, you have to follow the same steps, save the code and run from Linux terminal. We use vi editor for saving the program.

To run the program from Linux terminal

```
$ python3 hello.py
Hello World
```

In Linux, you can convert a Python program into a self executable script. The first statement in the code should be a shebang. It must contain the path to Python executable. In Linux, Python is installed in /usr/bin directory, and the name of the executable is python3. Hence, we add this statement to hello.py file

```
#!/usr/bin/python3
print ("Hello World")
```

You also need to give the file executable permission by using the chmod +x command

```
$ chmod +x hello.py
```
Then, you can run the program with following command line:

```
mvl@GNVBGL3:~$ ./hello.py
```

The **output** is shown below:

![Output](image)

Thus, we can write and run Hello World program in Python using the interpreter mode and script mode.

=========  
End of ebook preview  
If you liked what you saw...  
Buy it from our store @ https://store.tutorialspoint.com