According to New Credit System Syllabus

DBATU

Second Year (S.Y.) B. TECH Semester - IV
Course in ELECTRONICS & TELECOMMUNICATION ENGINEERING/
ELECTRICAL & ELECTRICAL (SANDWICH) ENGINEERING

PYTHON PROGRAMMING

Dr. ARCHANA K. RATNAPARAKHI
APARNA B. BARBADEKAR
Dr. SMITA CHAUDHARI
NITIN N. SAKHARE



www.pragationline.com
niralipune@pragationline.com
www.facebook.com/niralibooks
@nirali.prakashan

NIRALI PRAKASHAN
ADVANCEMENT OF KNOWLEDGE

A TEXT BOOK OF

PYTHON PROGRAMMING

(Elective)

FOR

Semester – IV

SECOND YEAR (S.Y.) B. TECH COURSE IN
ELECTRONICS AND TELECOMMUNICATION ENGINEERING /
ELECTRICAL AND ELECTRICAL (SANDWICH) ENGINEERING

Strictly According to New Revised Credit System Syllabus of
Dr. Babasaheb Ambedkar Technological University (DBATU),
Lonere, Dist. Raigad, Maharashtra.
(w.e.f 2021 - 2022)

Dr. ARCHANA K. RATNAPARAKHI
Ph. D., (E&TC)
Assistant Professor,
E&TC Engg. Dept.,
Vishwakarma Institute of Information Technology,
Kondhwa (Bk.), Pune.

APARNA B. BARBADEKAR
M.E. (Electronics)
Assistant Professor,
E&TC Engg. Dept.,
Vishwakarma Institute of Information Technology,
Kondhwa (Bk.), Pune.

Dr. SMITA CHAUDHARI
Ph. D., (Comp. Science & Engg.)
Assistant Professor,
Computer Engg. Dept.,
Dr. D. Y. Patil Institute of Technology,
Pimpri, Pune.

NITIN N. SAKHARE
M.E. (Computer Networks)
Assistant Professor,
Computer Engg. Dept.,
Vishwakarma Institute of Information Technology,
Kondhwa (Bk.), Pune.

NIRALI PRAKASHAN
ADVANCEMENT OF KNOWLEDGE

N6221

PYTHON PROGRAMMING (DBATU) **ISBN: 978-93-5451-527-9**

First Edition : June 2022
© : Authors

The text of this publication, or any part thereof, should not be reproduced or transmitted in any form or stored in any computer storage system or device for distribution including photocopy, recording, taping or information retrieval system or reproduced on any disc, tape, perforated media or other information storage device etc., without the written permission of Authors with whom the rights are reserved. Breach of this condition is liable for legal action.

Every effort has been made to avoid errors or omissions in this publication. In spite of this, errors may have crept in. Any mistake, error or discrepancy so noted and shall be brought to our notice shall be taken care of in the next edition. It is notified that neither the publisher nor the authors or seller shall be responsible for any damage or loss of action to any one, of any kind, in any manner, therefrom. The reader must cross check all the facts and contents with original Government notification or publications.

Published By :
NIRALI PRAKASHAN
Abhyudaya Pragati, 1312 Shivaji Nagar
Off J.M. Road, PUNE 411005
Tel : (020) 25512336/37/39
Email : niralipune@pragationline.com

Polyplate

Printed By :
YOGIRAJ PRINTERS AND BINDERS
Survey No. 10/1A, Ghule Industrial Estate
Nanded Gaon Road
Nanded, Pune 411041

DISTRIBUTION CENTRES

PUNE

Nirali Prakashan
(For orders outside Pune)
S. No. 28/27, Dhayari Narhe Road, Near Asian College
Pune 411041, Maharashtra
Tel : (020) 24690204; Mobile : 9657703143
Email : bookorder@pragationline.com

Nirali Prakashan
(For orders within Pune)
119, Budhwar Peth, Jogeshwari Mandir Lane
Pune 411002, Maharashtra
Tel : (020) 2445 2044; Mobile : 9657703145
Email : niralilocal@pragationline.com

MUMBAI

Nirali Prakashan
Rasdhara Co-op. Hsg. Society Ltd., 'D' Wing Ground Floor, 385 S.V.P. Road
Girgaum, Mumbai 400004, Maharashtra
Mobile : 7045821020, Tel : (022) 2385 6339 / 2386 9976
Email : niralimumbai@pragationline.com

DISTRIBUTION BRANCHES

DELHI

Nirali Prakashan
Room No. 2 Ground Floor
4575/15 Omkar Tower, Agarwal Road
Darya Ganj, New Delhi 110002
Mobile : 9555778814/9818561840
Email : delhi@niralibooks.com

BENGALURU

Nirali Prakashan
Maitri Ground Floor, Jaya
Apartments, No. 99, 6th Cross, 6th
Main, Malleswaram, Bengaluru
560003 Karnataka; Mob : 9686821074
Email : bengaluru@niralibooks.com

NAGPUR

Nirali Prakashan
Above Maratha Mandir, Shop No. 3,
First Floor, Rani Jhanshi Square,
Sitabuldi Nagpur 440012 (MAH)
Tel : (0712) 254 7129
Email : nagpur@niralibooks.com

KOLHAPUR

Nirali Prakashan
438/2, Bhosale Plaza, Ground Floor,
Khasbag, Opp. Balgopal Talim,
Kolhapur 416 012 Maharashtra
Mob : 9850046155
Email : kolhapur@niralibooks.com

JALGAON

Nirali Prakashan
34, V. V. Golani Market, Navi Peth,
Jalgaon 425001, Maharashtra
Tel : (0257) 222 0395
Mob : 94234 91860
Email : jalgaon@niralibooks.com

SOLAPUR

Nirali Prakashan
R-158/2, Avanti Nagar, Near Golden
Gate, Pune Naka Chowk
Solapur 413001, Maharashtra
Mobile 9890918687
Email : solapur@niralibooks.com

marketing@pragationline.com | www.pragationline.com

Also find us on www.facebook.com/niralibooks

PREFACE

It gives us great pleasure to present the book **'Python Programming'** for the students of **Semester IV Second Year (S.Y.) B.Tech. Course in Electronics and Telecommunication Engineering / Electrical and Electrical (Sandwich) Engineering of Dr. Babasaheb Ambedkar Technological University (DBATU), Lonere, Dist. Raigad (Maharashtra).** This book is strictly as per the new revised syllabus, 2020 Pattern, effective from the Academic Year 2021 - 2022.

The Theory Course will have 4 Credits.

In New Revised Syllabus, there will be Internal Assessment (20 Marks), Mid Sem. Exam. (20 Marks) and End Sem. Exam. (60 Marks). End Sem. Exam. will be based on all Five Units.

The basic objective of this book is to bridge the gap between the vast contents of the reference books, written by the renowned International Authors and the concise requirements of Undergraduate Students. This book has been written in a comprehensive manner using Simple and Lucid language, keeping in mind student's requirements. The main emphasis has been given on exploring the basic concepts rather than merely the Information. Topic-wise Programs and Exercises have been provided throughout the book at the end of the Unit. Also we have given **Model Question Papers** for practice at the end of the book.

Our special thanks to our family members, students and all those who directly or indirectly supported us in this project.

We also take this opportunity to express our sincere thanks to Shri. Dineshbhai Furia, Shri. Jignesh Furia, Mrs. Nirali Verma and entire team of Nirali Prakashan, namely Mrs. Deepali Lachake (Co-ordinator), who really have taken keen interest and untiring efforts in publishing this text.

We wish to make a special mention of the valuable contribution made by Late Shri M.P. Munde during the last two decades in reaching out to students, parents and teachers which eventually made **Nirali Prakashan, a brand of trust and quality in technical books**. Without his perseverance and zeal, successive editions would not have been possible.

The advice and suggestions of our esteemed readers to improve the text are most welcome and will be highly appreciated.

Pune **Authors**

SYLLABUS

Unit I : Introduction **07 Hours**

History of Python, Need of Python Programming, Applications Basics of Python Programming Using the REPL(Shell), Running Python Scripts, Variables, Assignment, Keywords, Input-Output, Indentation.

Unit II : Types, Operators and Expressions **07 Hours**

Types – Integers, Strings, Booleans; Operators- Arithmetic Operators, Comparison (Relational) Operators, Assignment Operators, Logical Operators, Bitwise Operators, Membership Operators, Identity Operators, Expressions and order of evaluations Control Flow- if, if-elif-else, for, while break, continue, pass.

Data Structures Lists – Operations, Slicing, Methods; Tuples, Sets, Dictionaries, Sequences, Comprehensions

Unit III : Default Arguments **07 Hours**

Default Arguments, Variable-length arguments, Anonymous Functions, Fruitful Functions (Function Returning Values), Scope of the Variables in a Function- Global and Local Variables. Modules: Creating modules, import statement, From-import statement, name spacing, Python packages, Introduction to PIP, Installing Packages via PIP, Using Python Packages.

Unit IV : Object-Oriented Programming OOP in Python **07 Hours**

Classes, „self-variable", Methods, Constructor Method, Inheritance, Overriding Methods, Data hiding, Error, and Exceptions: Difference between an error and Exception, Handling Exception, try except for block, Raising Exceptions, User Defined Exceptions.

Unit V : Brief Tour of the Standard Library **07 Hours**

Operating System Interface – String Pattern Matching, Mathematics, Internet Access, Dates and Times, Data Compression, Multithreading, GUI Programming, Turtle Graphics Testing: Why testing is required? Basic concepts of testing, Unit testing in Python, Writing Test cases, Running Tests.

CONTENTS

Unit I : Introduction **1.1-1.22**

1.1 History of Python 1.1
1.2 Need of Python Programming 1.2
1.3 Applications of Python 1.3
1.4 Basics of Python 1.5
1.4.1 Script Mode Programming 1.6
1.4.2 Python Identifiers 1.6
1.4.3 Reserved Words 1.6
1.5 Lines and Indentation 1.6
1.6 Programming Using the REPL (Shell) 1.8
1.7 Execute Python Script 1.9
1.8 Variables and Assignment 1.13
1.8.1 Keywords 1.17
1.9 Input-Output 1.21
1.10 Indentation 1.22
- **Exercise** 1.22

Unit II : Types, Operators and Expressions **2.1-2.46**

2.1 Types 2.1
2.2 Operators 2.3
2.3 Indentation in Python 2.8
2.4 Python for Loop 2.11
2.5 Loop Control Statements 2.14
2.6 Data Structures 2.18
2.6.1 Lists 2.18
2.6.2 Characteristics of Lists 2.19
2.7 Methods 2.23
2.7.1 Creating a Function 2.24
2.7.2 Function Calling 2.24
2.7.3 Arguments in Function 2.25
2.7.4 Types of Arguments 2.25
2.8 Tuples 2.28
2.8.1 Tuple Indexing and Slicing 2.29
2.8.2 Negative Indexing 2.29
2.8.3 Deleting Tuple 2.30
2.9 Sets 2.31
2.9.1 Creating a Set 2.31
2.9.2 Adding Items to the Set 2.32
2.10 Python Set Operations 2.34
2.10.1 Union of Two Sets 2.34
2.10.2 Intersection of Two Sets 2.35
2.10.3 Set Comparisons 2.36

2.11 Dictionaries 2.38
2.11.1 Creating the Dictionary 2.38
2.11.2 Accessing the Dictionary Values 2.39
2.11.3 Adding Dictionary Values 2.39
2.11.4 Deleting Some of the Employee Data 2.40
2.11.5 Properties of Dictionary Keys 2.41
2.12 Sequences 2.42
2.13 Comprehensions 2.43
2.13.1 List Comprehension 2.43
2.13.2 Dictionary Comprehensions 2.44
2.13.3 Set Comprehension 2.44
2.13.4 Generator Comprehension 2.44
- **Exercise** 2.46

Unit III : Default Arguments **3.1-3.18**

3.1 Default Arguments 3.1
3.2 Variable-Length Arguments 3.3
3.3 Anonymous Functions 3.4
3.4 Fruitful Functions (Function Retruning Values) 3.5
3.4.1 Return Values 3.5
3.4.2 Program Development 3.6
3.4.3 Debugging with Print 3.7
3.4.4 Composition 3.8
3.4.5 Boolean Functions 3.8
3.5 Scope of the Variables in a Function- Global and Local Variables 3.9
3.5.1 Local Variables 3.9
3.5.2 Global Variables 3.9
3.6 Modules 3.10
3.6.1 Creating Modules 3.10
3.6.2 The Import Statement 3.10
3.6.3 The from-import Statement 3.10
3.6.4 Name Spacing 3.11
3.6.5 Renaming a Module 3.12
3.6.6 The reload() Function 3.12
3.7 Python Packages 3.13
3.7.1 Importing a Module from a Package 3.13
3.7.2 Install a Package Globally 3.14
3.8 Introduction to PIP 3.15
3.9 Installing Packages via PIP 3.15
3.10 Using Python Packages 3.16
- **Exercise** 3.17

Unit IV : Object-Oriented Programming OOP in Python **4.1-4.18**

4.1 Introduction 4.1
4.2 Classes 4.1
4.3 Self-Variable 4.2
4.4 Methods 4.4
4.4.1 Python Method 4.5

4.5 Consructor Method 4.6
4.5.1 Creating the Constructor in Python 4.6
4.6 Inheritance 4.9
4.6.1 Python Multi-Level Inheritance 4.9
4.6.2 Python Multiple Inheritance 4.10
4.6.3 The isinstance (obj, class) Method 4.11
4.7 Method Overriding 4.11
4.8 Overriding Methods 4.12
4.9 Data Hiding 4.13
4.10 Error and Exceptions 4.14
4.11 Python Exception 4.14
4.12 Handling Exception 4.15
4.12.1 The try-expect Statement 4.15
4.13 Raising Exceptions 4.17
4.14 User Defined Exceptions 4.18
- **Exercise** 4.18

Unit V : Brief Tour of the Standard Library **5.1-5.38**

5.1 Introduction to Python Standard Library 5.1
5.2 Operating System Interface 5.1
5.2.1 String Pattern Matching 5.1
5.2.2 Mathematics 5.2
5.2.3 Internet Access 5.2
5.2.4 Dates and Times 5.2
5.2.5 Data Compression 5.3
5.2.6 Multithreading 5.3
5.3 GUI Programming 5.6
5.3.1 Tkinter Widgets 5.6
5.3.2 Python Tkinter Geometry 5.7
5.4 Turtle Graphics Testing 5.8
5.4.1 Introduction 5.8
5.4.2 Programming with Turtle 5.9
5.4.3 Why Testing is Required? 5.22
5.4.4 Basic Concepts of Testing 5.22
5.4.5 Unit Testing in Python 5.24
5.4.6 Writing Test Cases 5.28
5.4.7 Running Tests 5.32
- **Exercise** 5.37

- **Model Question Papers for**
  - ➢ **Mid-Semester Examination (20 Marks)** **P.1-P.1**
  - ➢ **End-Semester Examination (60 Marks)** **P.2-P.2**

✠ ✠ ✠

UNIT - I

INTRODUCTION

1.1 HISTORY OF PYTHON

- Python laid its foundation in the late 1980s.
- The implementation of Python was started in December 1989 by **Guido Van Rossum** at CWI in Netherland.
- In February 1991, **Guido Van Rossum** published the code (labeled version 0.9.0) to alt.sources.
- In 1994, Python 1.0 was released with new features like lambda, map, filter, and reduce.
- Python 2.0 added new features such as list comprehensions, garbage collection systems.
- On December 3, 2008, Python 3.0 (also called "Py3K") was released. It was designed to rectify the fundamental flaw of the language.
- *ABC programming language* is said to be the predecessor of Python language, which was capable of Exception Handling and interfacing with the Amoeba Operating System.
- The following programming languages influence Python:
  - ➢ ABC language.
  - ➢ Modula-3

Why the Name Python?

There is a fact behind choosing the name Python.

- Guido van Rossum was reading the script of a popular BBC comedy series "Monty Python's Flying Circus". It was late on-air 1970s.
- Van Rossum wanted to select a name which unique, sort, and little-bit mysterious. So he decided to select naming Python after the "Monty Python's Flying Circus" for their newly created programming language.
- The comedy series was creative and well random. It talks about everything. Thus it is slow and unpredictable, which made it very interesting.
- Python is also versatile and widely used in every technical field, such as
  - ➢ Machine Learning
  - ➢ Artificial Intelligence
  - ➢ Web Development
  - ➢ Mobile Application
  - ➢ Desktop Application
  - ➢ Scientific Calculation, etc.

Python Version List

- Python programming language is being updated regularly with new features and supports. There are lots of update in Python versions, started from 1994 to current release.
- A list of Python versions with its released date is given below.

Table 1.1

Python Version	Released Date
Python 1.0	January 1994
Python 1.5	December 31, 1997
Python 1.6	September 5, 2000
Python 2.0	October 16, 2000
Python 2.1	April 17, 2001
Python 2.2	December 21, 2001
Python 2.3	July 29, 2003
Python 2.4	November 30, 2004
Python 2.5	September 19, 2006
Python 2.6	October 1, 2008
Python 2.7	July 3, 2010
Python 3.0	December 3, 2008
Python 3.1	June 27, 2009
Python 3.2	February 20, 2011
Python 3.3	September 29, 2012
Python 3.4	March 16, 2014
Python 3.5	September 13, 2015
Python 3.6	December 23, 2016
Python 3.7	June 27, 2018
Python 3.8	October 14, 2019

1.2 NEED OF PYTHON PROGRAMMING

- The most common question asked by the beginners - **"What is the best way to learn Python"?** It is the initial and relevant question because first step in learning any programming language is to know how to learn.
- The proper way of learning will help us to learn fast and become a good Python developer.
- In this section, we will discuss various tips that we should keep in mind while learning Python.

1. Make it Clear Why We Want to Learn

- The goal should be clear before learning the Python. Python is an easy, a vast language as well. It includes numbers of libraries, modules, in-built functions and data structures.
- If the goal is unclear then it will be a boring and monotonous journey of learning Python. Without any clear goal, you perhaps won't make it done.
- So, first figure out the motivation behind learning, which can anything be such as knowing something new, develop projects using Python, switch to Python, etc. Below are the general areas where Python is widely used. Pick any of them.
  - ➢ Data Analysis and Processing
  - ➢ Artificial Intelligence
  - ➢ Games
  - ➢ Hardware / Sensor / Robots
  - ➢ Desktop Applications
- Choose any one or two areas according to your interest and start the journey towards learning Python.

2. Learn the Basic Syntax

- It is the most essential and basic step to learn the syntax of the Python programming language. We have to learn the basic syntax before dive deeper into learning it.
- As we have discussed in our earlier point, Python is easy to learn and has a simple syntax. It doesn't use semicolon and brackets. Its syntax is like the English language.
- So it will take minimum amount of time to learning its syntax. Once we get its syntax properly, further learning will be easier and quicker getting to work on projects.

3. Write Code by Own

- Writing the code is the most effective and robust way to learn Python. First, try to write code on paper and run in mind (Dry Run) then move to the system.
- Writing code on paper will help us get familiar quickly with the syntax and the concept store in the deep memory. While writing the code, try to use proper functions and suitable variables names.
- There are many editors available for Python programming which highlights the syntax related issue automatically. So we don't need to pay lot of attention of these mistakes.

4. Keep Practicing

- The next important step is to do the practice. It needs to implementing the Python concepts through the code. We should be consistence to our daily coding practice.
- Consistency is the key of success in any aspect of life not only in programming. Writing code daily will help to develop muscle memory.
- We can do the problem exercise of related concepts or solve at least 2 or 3 problems of Python. It may seem hard but muscle memory plays large part in programing. It will take us ahead from those who believe only the reading concept of Python is sufficient.

5. Make Notes as Needed

- Creating notes by own is an excellent method to learn the concepts and syntax of Python. It will establish stability and focus that helps you become a Python developer.
- Make brief and concise notes with relevant information and include appropriate examples of the subject concerned.
- Maintain own notes are also helped to learn fast. A study published in Psychological Science that -

6. Discuss Concepts with Other

- Coding seems to be solitary activity, but we can enhance our skills by interacting with the others. We should discuss our doubts to the expert or friends who are learning Python.
- This habit will help to get additional information, tips and tricks, and solution of coding problems. One of the best advantages of Python, it has a great community. Therefore, we can also learn from passionate Python enthusiasts.

7. Do small Projects

- After understanding Python's basic concept, a beginner should try to work on small projects. It will help to understand Python more deeply and become more competent in it.
- Theoretical knowledge is not enough to get command over the Python language. These projects can be anything as long as they teach you something.

- You can start with the small projects such as calculator app, a tic-toc-toe game, an alarm clock app, a to-do list, student or customer management system, etc.
- Once you get handy with a small project, you can easily shift toward your interesting domain (Machine Learning, Web Development, etc.).

8. Teach Others

- There is a famous saying that "If you want to learn something then you should teach other". It is also true in case of learning Python. Share your information to other students via creating blog posts, recording videos or taking classes in local training center.
- It will help us to enhance the understanding of Python and explore the unseen loopholes in your knowledge. If you don't want to do all these, join the online forum and post your answers on Python related questions.

9. Explore Libraries and Frameworks

- Python consists of vast libraries and various frameworks. After getting familiar with Python's basic concepts, the next step is to explore the Python libraries.
- Libraries are essential to work with the domain specific projects. In the following section, we describe the brief introduction of the main libraries.
  - ➢ **TensorFlow :** It is an artificial intelligence library which allows us to create large scale AI based projects.
  - ➢ **Django :** It is an open source framework that allows us to develop web applications. It is easy, flexible, and simple to manage.
  - ➢ **Flask :** It is also an open source web framework. It is used to develop lightweight web applications.
  - ➢ **Pandas :** It is a Python library which is used to perform scientific computations.
  - ➢ **Keras :** It is an open source library, which is used to work around the neural network.
- There are many libraries in Python. Above, we have mentioned a few of them.

10. Contribute to Open Source

- As we know, Python is an open source language that means it is freely available for everyone. We can also contribute to Python online community to enhance our knowledge.
- Contributing to open source projects is the best way to explore own knowledge. We also receive the feedback, comments or suggestions for work that we submitted.
- The feedback will enable the best practices for Python programming and help us to become a good Python developer.

1.3 APPLICATIONS OF PYTHON

- Python is a general purpose, open source, high-level programming language and also provides number of libraries and frameworks.
- Python has gained popularity because of its simplicity, easy syntax and user-friendly environment. The usage of Python as follows.
  - ➢ Desktop Applications
  - ➢ Web Applications
  - ➢ Data Science
  - ➢ Artificial Intelligence
  - ➢ Machine Learning
  - ➢ Scientific Computing
  - ➢ Robotics
  - ➢ Internet of Things (IoT)
  - ➢ Gaming
  - ➢ Mobile Apps
  - ➢ Data Analysis and Pre-processing

Applications

- Python is known for its general-purpose nature that makes it applicable in almost every domain of software development.
- Python makes its presence in every emerging field. It is the fastest-growing programming language and can develop any application.
- Here, we are specifying application areas where Python can be applied.

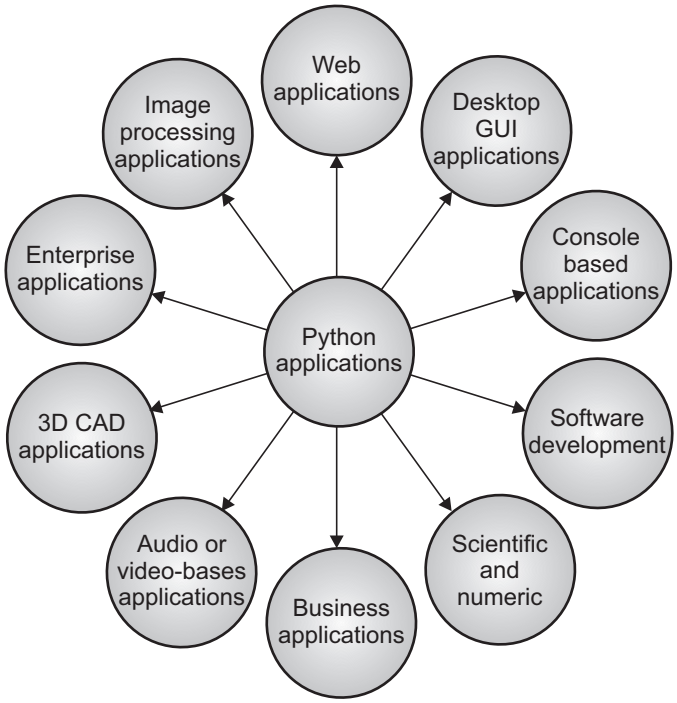


Fig. 1.1

1. Web Applications

- We can use Python to develop web applications. It provides libraries to handle internet protocols such as HTML and XML, JSON, Email processing, request, beautifulSoup, Feedparser, etc.
- One of Python web-framework named Django is used on Instagram. Python provides many useful frameworks, and these are given below:
  - ➢ Django and Pyramid framework (Use for heavy applications)
  - ➢ Flask and Bottle (Micro-framework)
  - ➢ Plone and Django CMS (Advance Content management)

2. Desktop GUI Applications

- The GUI stands for the Graphical User Interface, which provides a smooth interaction to any application. Python provides a Tk GUI library to develop a user interface.
- Some popular GUI libraries are given below.
  - ➢ Tkinter or Tk
  - ➢ wxWidgetM
  - ➢ Kivy (used for writing multitouch applications)
  - ➢ PyQt or Pyside

3. Console-based Application

- Console-based applications run from the command-line or shell. These applications are computer program which are used commands to execute. This kind of application was more popular in the old generation of computers.
- Python can develop this kind of application very effectively. It is famous for having REPL, which means **the Read-Eval-Print Loop** that makes it the most suitable language for the command-line applications.
- Python provides many free library or module which helps to build the command-line apps. The necessary **IO** libraries are used to read and write. It helps to parse argument and create console help text out-of-the-box. There are also advance libraries that can develop independent console apps.

4. Software Development

- Python is useful for the software development process. It works as a support language and can be used to build control and management, testing, etc.
  - ➢ SCons is used to build control.
  - ➢ Buildbot and Apache Gumps are used for automated continuous compilation and testing.
  - ➢ Round or Trac for bug tracking and project management.

5. Scientific and Numeric

- This is the era of Artificial intelligence where the machine can perform the task the same as the human. Python language is the most suitable language for Artificial intelligence or machine learning. It consists of many scientific and mathematical libraries, which makes easy to solve complex calculations.
- Implementing machine learning algorithms require complex mathematical calculation. Python has many libraries for scientific and numeric such as Numpy, Pandas, Scipy, Scikit-learn, etc.
- If you have some basic knowledge of Python, you need to import libraries on the top of the code. Few popular frameworks of machine libraries are given below.
  - ➢ SciPy
  - ➢ Scikit-learn
  - ➢ NumPy
  - ➢ Pandas
  - ➢ Matplotlib

6. Business Applications

- Business Applications differ from standard applications. E-commerce and ERP are an example of a business application.
- This kind of application requires extensively, scalability and readability, and Python provides all these features.
- Oddo is an example of the all-in-one Python-based application which offers a range of business applications. Python provides a Tryton platform which is used to develop the business application.

7. Audio or Video-based Applications

- Python is flexible to perform multiple tasks and can be used to create multimedia applications. Some multimedia applications which are made by using Python are TimPlayer, cplay, etc. The few multimedia libraries are given below.
  - ➢ Gstreamer
  - ➢ Pyglet
  - ➢ QT Phonon

8. 3D CAD Applications

- The CAD (Computer-aided design) is used to design engineering related architecture. It is used to develop the 3D representation of a part of a system. Python can create a 3D CAD application by using the following functionalities.

- ➢ Fandango (Popular)
- ➢ CAMVOX
- ➢ HeeksCNC
- ➢ AnyCAD
- ➢ RCAM

9. Enterprise Applications

- Python can be used to create applications that can be used within an Enterprise or an Organization. Some real-time applications are OpenERP, Tryton, Picalo, etc.

10. Image Processing Application

- Python contains many libraries that are used to work with the image. The image can be manipulated according to our requirements. Some libraries of image processing are given below.
  - ➢ OpenCV
  - ➢ Pillow
  - ➢ SimpleITK
- In this topic, we have described all types of applications where Python plays an essential role in the development of these applications.

1.4 BASICS OF PYTHON

Python is a general-purpose interpreted, interactive, object-oriented, and high-level 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).

Why to Learn Python?

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

Python is a MUST for students and working professionals to become a great Software Engineer specially when they are working in Web Development Domain. Below are the 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.

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.

Hello World using Python.

- Just to give you a little excitement about Python, I'm going to give you a small conventional Python Hello World program.

```
print("Hello, Python!");
```

- The Python language has many similarities to Perl, C, and Java. However, there are some definite differences between the languages.

First Python Program

Let us execute programs in different modes of programming.

Interactive Mode Programming

Invoking the interpreter without passing a script file as a parameter brings up the following prompt –

Program 1.1 :

```
$ python
Python2.4.3(#1,Nov112010,13:34:43)
[GCC 4.1.220080704(RedHat4.1.2-48)] on linux2
Type"help","copyright","credits"or"license"for more information.
>>>
```

Type the following text at the Python prompt and press the Enter –

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

If you are running new version of Python, then you would need to use print statement with parenthesis as in print ("Hello, Python!");. However in Python version 2.4.3, this produces the following result −

```
Hello, Python!
```

1.4.1 Script Mode Programming

- Invoking the interpreter with a script parameter begins execution of the script and continues until the script is finished. When the script is finished, the interpreter is no longer active.
- Let us write a simple Python program in a script. Python files have extension .py. Type the following source code in a test.py file −

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

- We assume that you have Python interpreter set in PATH variable. Now, try to run this program as follows −

```
$ python test.py
```

This produces the following result −

```
Hello, Python!
```

Let us try another way to execute a Python script. Here is the modified test.py file −

```
#!/usr/bin/python

print"Hello, Python!"
```

We assume that you have Python interpreter available in /usr/bin directory. Now, try to run this program as follows −

```
$ chmod+x test.py     # This is to make file executable
$./test.py
```

This produces the following result −

```
Hello, Python!
```

1.4.2 Python Identifiers

- A Python identifier is a name used to identify a variable, function, class, module or other object. An identifier starts with a letter A to Z or a to z or an underscore (_) followed by zero or more letters, underscores and digits (0 to 9).
- Python does not allow punctuation characters such as @, $, and % within identifiers. Python is a case sensitive programming language. Thus, Manpower and manpower are two different identifiers in Python.

Here are naming conventions for Python identifiers −

- Class names start with an uppercase letter. All other identifiers start with a lowercase letter.
- Starting an identifier with a single leading underscore indicates that the identifier is private.
- Starting an identifier with two leading underscores indicates a strongly private identifier.
- If the identifier also ends with two trailing underscores, the identifier is a language-defined special name.

1.4.3 Reserved Words

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.

Table 1.2

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

1.5 LINES AND INDENTATION

- Python provides no braces to indicate blocks of code for class and function definitions or flow control. Blocks of code are denoted by line indentation, which is rigidly enforced.
- The number of spaces in the indentation is variable, but all statements within the block must be indented the same amount. For example −

  if True:
  print "True"
  else:
  print "False"

- However, the following block generates an error −

Program 1.2 :

```
ifTrue:
print"Answer"
print"True"
else:
print"Answer"
print"False"
```