|  |  |
| --- | --- |
| 1.Introductory Concepts covering installation on different OS, version history, interpreter. This section also covers questions like Why, Who, What and Where on Python. | |
| 1. Installing Python 2. Which Python is right for you? 3. Python & your OS 4. Interactive Shell 5. Summary |
| 2. Python Object Types, Numeric Types, Data Structures, Control Structures, Scopes and Arguments | |
| 1. Your first program 2. Declaring Functions 3. Python Data types vs Other Languages 4. Documenting Functions 5. Everything is an Object 6. The Import Search Path 7. What is an Object? 8. Indenting Code 9. Testing Modules 10. Native Datatypes     1. Dictionaries     2. List     3. Tuples 11. Variables & referencing 12. Types and Objects 13. Operators and Expressions 14. Program Structure and Control Flow 15. Functions and Functional Programming 16. Classes and Object Oriented Programming 17. Modules, Packages and Distribution 18. Input and Output 19. Execution Environment 20. Testing, Debugging, Profiling and Tuning 21. Python Overview     1. Built-in Data types     2. Control Structures     3. Module     4. OOPs 22. Basics     1. Lists     2. Dictionaries     3. Tuple     4. Sets     5. Strings     6. Control Flow 23. Functions 24. Modules and Scoping Rules 25. Python Programs 26. Introducing Python Object Types     1. Why use built-in Types?     2. Core data types     3. Numbers, Lists, Dictionaries, Tuples, Files, Other Core Types     4. User Defined Classes 27. Numeric Types     1. Literals, Built-in tools, expression operators     2. Formats, Comparisons, Division, Precision     3. Complex Numbers     4. Hexadecimal, Octal & Binary     5. Bitwise Operations 28. Decimal, Fraction, Sets, Booleans 29. Statements & Syntax 30. Assignments, Expressions & Syntax 31. If Tests & Syntax Rules 32. Scopes 33. Arguments |
| 3. Built-in functions, Function Design, Recursive Functions, Introspection, Annotations, Lambda, Filter and Reduce | |
| 1. Power of Introspection    1. Optional and Named Arguments    2. type, str, dir and other built-in functions    3. Object References with getattr    4. Filtering Lists    5. Lambda Functions    6. Real world Lambda functions 2. Built-in functions 3. Python run-time services 4. Advanced Function Topics    1. Function Design    2. Recursive Functions    3. Attributes and Annotation    4. Lambda    5. Mapping Functions over sequences    6. Filter and Reduce 5. Special Class Attributes 6. Display Tool |
| 4. OOPS, Modules | |
| 1. Objects and Object Orientation    1. Importing Modules    2. Defining Classes    3. Initializing and Coding Classes    4. Self & \_\_init\_\_    5. Instantiating Classes    6. Garbage Collection    7. Wrapper Classes    8. Special Class Methods    9. Advanced Class Methods    10. Class Attributes    11. Private Functions 2. Packages 3. Data Types and Objects 4. Advanced Object Oriented Features 5. Modules    1. Why use Modules?    2. Program Architecture    3. Module Search Path    4. Module Creation & Usage    5. Namespaces    6. Reloading Modules    7. Packages 6. Advanced Module Topics    1. Data Hiding in Modules    2. as Extension for import and from    3. Modules are Objects: Metaprograms    4. Transitive Module Reloads    5. Module Design Concepts    6. Module Gotchas 7. OOP    1. Why use classes?    2. Classes & Instances    3. Attribute Inheritance Search    4. Class Method Calls    5. Class Trees    6. Class Objects & Default Behavior    7. Instance Objects are Concrete Items    8. Intercepting Python Operators    9. Classes Vs. Dictionaries    10. Class customization by Inheritance    11. Operator Overloading    12. Subclasses    13. Polymorphism in Action    14. Designing with Classes    15. Mix-in Classes |
| 5. Advanced Class Topics   1. Advanced Class Topics    1. Extending Types by Embedding    2. Extending Types by Subclassing    3. Static and Class Methods    4. Decorators and Metaclasses    5. Class Gotchas | |
|  |
| 6. Exceptions | |
| 1. Exceptions and File Handling    1. Handling Exceptions    2. Using exceptions for other purposes 2. Exceptions Basics    1. Why use Exceptions?    2. Default Exception Handler    3. User-Defined Exceptions    4. Class Based Exceptions    5. Designing with Exceptions |
| 7. XML, HTTP, SOAP, Network Programming, I18N, Unicode | |
| 1. Regular Expressions 2. Parsing / Processing Mark-up languages (HTML, XML)    1. Unicode 3. HTTP Web Services    1. Headers    2. Debugging 4. SOAP Web Services 5. Networking 6. Internet 7. Email 8. Internationalization and Localization 9. Network Programming and Sockets 10. Internet Application Programming 11. Web Programming 12. Internet Data Handling & Encoding 13. Unicode and Bytes Strings |
| 8. Miscellaneous   1. Algorithms 2. Cryptography 3. Data compression and archiving 4. Processes and Threads 5. Data persistence & exchange 6. Extending & Embedding Python 7. GUI | |
|  |