VIBRANT TECHNOLOGY
IT Training Institute & IT Services

Python Lecture flow

The below given flow should be followed by each faculty while taking lectures. If the faculty decides to change the flow - he/she will need to first take permission from the Training coordinator at the HO (Ahmedabad office)

Module 13) Python - Fundamentals of python language
Lecture - 6

- Introduction of students
- Career in IT
- Using Lab
- Introduction of Python
- Programming Style
- Core python concepts
- Conditional Statements
- If- else Nested if-else
- Practical Examples: 1) How to the python code Structure work? 2) How to create variable in python? 3) How to take user input? 4) How to check the type of variable dynamically. 5) W.A.P to find greater and less than number using If_else 6) W.A.P to find prime number using if_else 7) W.A.P to find the grade according to percentage using if_else ladder. 8) W.A.P to find that who can donate the blood using Nested if.
- Looping For , While
- Generators and Iterators
- Nested loops
- map(), reduce(), filter() and Closures and Decorators
- Control Statements

• 1) WAP to print each fruit in list using simple for loop. List1 (apple,banana,mango) 2) WAP to find the length of string using simple for loop List1 (apple,banana,mango) 3) WAP to find particular string using simple for loop and simple if condition. 4) Print this pattern using nested for Loop.

• Break

• Continue

• Pass

• Practical Example: 1) W.A.P to skip the (Banana) from the list using Continue Statement List1 - (apple,banana,mango) 2) W.A.P to break the for loop when (Banana) get in if Condition.

• String Manipulation

• Accessing Strings

• Basic Operations

• String slices

• Function and Methods

• 1) W.A.P to print (Hello) using string 2) W.A.P to allocate the string to a variable. 3) W.A.P to print String using three quotes 4) W.A.P to access the 1st position character using index value. 5) W.A.P to Access the string after the index value 1. 6) W.A.P to Access the string before the index value 5. 7) W.A.P to Access the String between the index value 1 to 4 8) W.A.P to print the string from the last index value. 9) W.A.P to print the String alternate character after the index value 1. 10) W.

Module 14) Python - Collections, functions and Modules in Python Lecture - 8

• Accessing list

• Operations

• Working with List

• Function and Method

• Practical Example: 1) W.A.P create the list of multiple datatype element. 2) W.A.P to find the length of the list. 3) W.A.P to update the list using the insert() and append() 4) W.A.P to remove the element using the pop() and remove()

• Tuple

• Accessing Tuples

• Operations Working

• Functions and Method

• Dictionaries

• Accessing value in dictionaries

• Working with dictionaries

• Property

• Practical Example: 1) W.A.P to access value on index value in the list 2) W.A.P to access the value after the index value 1. 3) W.A.P to access the value between 1 to 5 4) W.A.P to access the value till index 5. 5) W.A.P to update the list using the index value. 6) W.A.P to irate the list using for loop. 7) W.A.P to insert the value in empty list using for loop and append(). 8) W.A.P to delete the element using del() 9) W.A.P to sort the list using sort() and sorted()

• 10) W.A.P to round the value in list using round() and for loop. 11) W.A.P to convert the list into tuple. 12) W.A.P to create tuple with multiple data type. 13) W.A.P to concate the two tuple into one tuple. 14) W.A.P to access the value of index value 1st in tuple. 15) W.A.P to access the value from last in tuple. 16) W.A.P to access the value between index 1st to 5th from the tuple. 17) W.A.P to access the alternate value between index 1st to 5th.

• 18) W.A.P to create the dictionary of having 6 key and value pair. 19) W.A.P to access the value using the key from dictionary. 20) W.A.P to update the value on particular key. 21) W.A.P to separate the key and value from dictionary using keys() and values() of dictionary. 22) W.A.P to convert the two list into one dictionary using for loop. 23) W.A.P to convert the list using zip() of

dictionary. 24) W.A.P to count the character repeat in string.

• Function

• Types of Function

• Function Argument

• anonymous function

• Practical Example: 1) W.A.P to print the String using the function. 2) W.A.P to create the parameterized function. 3) W.A.P to print multiple string using function. 4) W.A.P to create calculator using function. 5) W.A.P to create lamba function using one expression. 6) W.A.P to create lamba function using two expression. 7) W.A.P to create lamba function using three expression. 8) W.A.P to create a return type function using lamda function.

• Modules

• Importing Module

• Math Module

• Random module

• Packages

• Practical Example: 1) W.A.P to import another module into one module. 2) W.A.P to use all the Math module function.

Module 15) Python - Advance python programming Lecture - 10

• Printing on screen

• Reading data from keyboard

• opeaning and closing file

• reading and writing file

• Practical Example : 1) W.A.P to create the file using the python. 2) W.A.P to create a file and print the string into the file. 3) W.A.P to read a file and print the data on console. 4) W.A.P to write the multiple String into file 5) W.A.P to read multiple String from the file. 6) W.A.P to

check where is the cursor in the file.

• Exception Handling

• Handling Exception

• Finally Clause

• PRactical Example: a) W.A.P to handle exception in calculator. b) W.A.P to handle multiple exception at time in one program. c) W.A.P to handle File Exception and use finally block for closing the file. d) W.A.P to print multiple exception using if else. e) W.A.P to print user define exception.

• class and object

• Attribute

• Inheritance

• Overloading

• Overriding

• Practical Example: 1) W.A.P to create a class and access the property of class using object. 2) W.A.P to create local variable and global variable. 3) W.A.P to show single inheritance. 4) W.A.P to show Multilevel inheritance. 5) W.A.P to show Multiple inheritance. 6) W.A.P to show Hierarchical inheritance. 7) W.A.P to show Hybrid inheritance. 8) W.A.P to using super() in inheritance. 9) W.A.P to show method overloading. 10) W.A.P to show Method overriding.

• sqlite3 and pymysql modules (database connectors)

• Search Function

• Match Function

• Matching Vs Searching

• Modifiers

• Practical Examples: 1) W.A.P to search a word from the string using Search() 2) W.A.P to match the word in string using Match().

• GUI Programming Introduction Tkinter programming

• Tkinter widgets

• Practical Example: 1) W.A.P to create GUI Frame. 2) W.A.P to create all the widgets using Tkinter.

Module 16) Python - DB and Python Framework - Industry Program
Lecture - 18

• HTML

• CSS

• javascript

• Django Introduction Advantages of django Django vs Flask

• Virtual Environment

• Project and app creation

• MVT pattern architecture

• Practical Example 1 Create Django Admin Panel 2 Creating the Doctor Finder Project. Project Practical Registration login , forgot password session management , email template , profile,updation , working with media , CRUD operations

• Practical Example: 1) Create Django Admin Panel 2) Creating the Doctor Finder Project.

• Django Admin

• URL pattern

• Template integration

• form validation using javascript

• Django Database connectivity mysql or sqllite

• ORM, Query set

• Django forms,Django authentication

• •Authentication(Sign up, login, logout, session, email sending, sms sending, otp verification, change password, forgot password, profile management)

• CRUD operationgs using AJAX

• Customization django admin panel

• Payment integration using paytm

• Github project deployment

• Live project deployment Python anywhere

• •Social authentication (For eg; Login with Google, Login with Facebook, Twitter, Github...etc)

• Google Maps API

Module 17) Python - Rest Framework - Industry Program Lecture - 3

• Introduction

• Requirements

• Serialization

• Requests and Responses

• Views

• URLs

• Pagination

• Settings

• Project setup

• Social Auth, Email and OTP Sending API, Third Party Integration

• RESTful API: Representational State Transfer (REST) is a widely used architectural style for building web services. Understanding REST principles and being able to create RESTful APIs is essential. CRUD API: CRUD stands for Create, Read, Update, and Delete, which are the basic operations performed on data. Creating APIs that allow these operations is fundamental to backend development. Authentication and Authorization API: Knowing how to implement user

authentication and authorization mechanisms is crucia

• OpenWeatherMap API: This API provides weather data for various locations worldwide. You can retrieve current weather conditions, forecasts, and historical weather data.

• Google Maps Geocoding API: This API allows you to convert addresses into geographic coordinates (latitude and longitude) and vice versa. You can use it to retrieve location data, calculate distances between points, and display maps.

• GitHub API: GitHub provides an API that enables you to interact with repositories, issues, pull requests, and more. You can perform actions like retrieving repository information, creating issues, and accessing user data.

• Twitter API: Twitter offers an API that allows you to integrate Twitter functionality into your applications. You can fetch tweets, post tweets, retrieve user information, and perform searches.

• REST Countries API: This API provides information about countries, including details like population, languages spoken, currencies, and more. You can retrieve country-specific data and use it for various applications.

• endGrid provides an API for sending transactional and marketing emails. You can integrate it into your applications to send emails, manage templates, and track delivery statistics.

• Social authentication (For eg; Login with Google, Login with Facebook...etc)

• Email sending APIs (For eg; Mailchimp, Mailgun...etc)

• SMS sending APIs (For eg; Twilio)

• Normal payments (For eg; Paypal, Stripe)

• Google Maps API