2 Marks Questions and Answers

Unit -I

FUNDAMENTALS OF PYTHON

1. What is meant by interpreter? [Remembering]

An interpreter is a computer program that executes instructions written in a programming language. It can either execute the source code directly or translate the source code in a first step into a more efficient representation and executes this code.

2. Define Python.[Understanding]

Python is a popular programming language. It was created by Guido van Rossum.

- web development (server-side),
- software development,
- mathematics,
- System scripting.

2. How will you invoke the python interpreter? [Remembering]

The Python interpreter can be invoked by typing the command "python" without any parameter followed by the "return" key at the shell prompt.

3. What are the advantages in python? [Remembering]

- Python works on different platforms (Windows, Mac, Linux, Raspberry Pi, etc).
- Python has a simple syntax similar to the English language.
- Python has syntax that allows developers to write programs with fewer lines than some other programming languages.
- Python runs on an interpreter system, meaning that code can be executed as soon as it is written. This means that prototyping can be very quick.
- Python can be treated in a procedural way, an object-orientated way or a functional way.

4. What is meant by interactive mode of the interpreter? [Understanding]

Interactive mode is a command line shell which gives immediate feedback for each statement, while running previously fed statements in active memory. As new lines are fed into the interpreter, the fed program is evaluated both in part and in whole.

5. Write a snippet to display “Hello World” in python interpreter.[Analysis]

In script mode:

```
>>> print "Hello World"
```

6. Define a variable and write down the rules for naming a variable. [Remembering]

A name that refers to a value is a variable. Variable names can be arbitrarily long. They can contain both letters and numbers, but they have to begin with a letter. It is legal to use uppercase letters, but it is good to begin variable names with a lowercase letter.

7. Define keyword and enumerate some of the keywords in Python. [Remembering]

A keyword is a reserved word that is used by the compiler to parse a program. Keywords cannot be used as variable names. Some of the keywords used in python are: and, del, from, not, while, is, continue.

8. Define statement and what are its types? [Understanding]

A statement is an instruction that the Python interpreter can execute. There are two types of statements: print and assignment statement.

9. What do you meant by an assignment statement? [Remembering]

An assignment statement creates new variables and gives them values:

Eg 1: Message = 'And now for something completely different'

Eg 2: n = 17

10. What is data type? List the different data types in python. [Remembering]

In programming, data type is an important concept.

Variables can store data of different types, and different types can do different things.

Python has the following data types built-in by default, in these categories:

Text Type:	str
Numeric Types:	int, float, complex
Sequence Types:	list, tuple, range
Mapping Type:	dict
Set Types:	set, frozenset
Boolean Type:	bool
Binary Types:	bytes, bytearray, memoryview

10. What is tuple? [Remembering]

A tuple is a sequence of immutable Python objects. Tuples are sequences, like lists. The differences between tuples and lists are, the tuples cannot be changed unlike lists and tuples use parentheses, whereas lists use square brackets. Creating a tuple is as simple as putting different comma-separated values. Comma-separated values between parentheses can also be used.

Example: tup1 = ('physics', 'chemistry',
1997,2000);

11. What is an expression? [Remembering]

An expression is a combination of values, variables, and operators. An expression is evaluated

using assignment operator.

Examples: Y=x + 17

12. What do you mean by an operand and an operator? Illustrate your answer with relevant example. [Remembering]

An operator is a symbol that specifies an operation to be performed on the operands. The data items that an operator acts upon are called operands. The operators +, -, *, / and ** perform addition, subtraction, multiplication, division and exponentiation. Example: 20+32

In this example, 20 and 32 are operands and + is an operator.

13. What is the order in which operations are evaluated? Give the order of precedence.

[Understanding]

The set of rules that govern the order in which expressions involving multiple operators and operands are evaluated is known as rule of precedence. Parentheses have the highest precedence followed by exponentiation. Multiplication and division have the next highest precedence followed by addition and subtraction.

14. Illustrate the use of * and + operators in string with example. [Apply]

The * operator performs repetition on strings and the + operator performs concatenation on

strings.

Example:

>>> ‘Hello*3’

Output: HelloHelloHello

>>>’Hello+World’

Output: HelloWorld

15. What is the symbol for comment? Give an example. [Understanding]

is the symbol for comments
in Python. Example:

compute the percentage of the hour that has elapsed

16. What is function call? [Remembering]

A function is a named sequence of statements that performs a computation. When we define a function, we specify the name and the sequence of statements. Later, we can "call" the function by its name called as function call.

17. Identify the parts of a function in the given example. [Understanding]

```
>>> betty = type("32")
>>> print betty
```

The name of the function is *type*, and it displays the type of a value or variable. The value or variable, which is called the argument of the function, is enclosed in parentheses. The argument is *32*. The function returns the result called return value. The return value is stored in *betty*.

18. What is a local variable? [Remembering]

A variable defined inside a function. A local variable can only be used inside its function.

19. What is a value? What are the different types of values? [Remembering]

A value is one of the fundamental things – like a letter or a number – that a program manipulates. Its types are: integer, float, boolean, strings and lists.

15. What is the output of the following? [Remembering]

a. float(32)
b. float("3.14159")

Output:

a. 32.0 The float function converts integers to floating-point numbers.

b. 3.14159 The float function converts strings to floating-point numbers.

16. What do you mean by flow of execution? [Remembering]

In order to ensure that a function is defined before its first use, we have to know the order in which statements are executed, which is called the flow of execution. Execution always begins at the first statement of the program. Statements are executed one at a time, in order from top to bottom.

17. Write down the output for the following program. [Understanding]

```
first = 'throat'
```

```
second =
'warbler'
print first
+ second
```

Output:

```
throatwarbler
```

18. Give the syntax of function definition. [Remembering]

```
def NAME( LIST OF PARAMETERS ):

STATEMENTS
```

19. Explain the concept of floor division. [Remembering]

The operation that divides two numbers and chops off the fraction part is known as floor division.

20. What is type coercion? Give example. [Remembering]

Automatic method to convert between data types is called type coercion. For mathematical operators, if any one operand is a float, the other is automatically converted to float.

Eg:

```
>>> minute = 59
>>> minute / 60.0
        0.983333333333
```

21. Write a math function to perform √2 / 2. [Remembering]

```
>>> math.sqrt(2) / 2.0
        0.707106781187
```

22. What is meant by traceback? [Remembering]

A list of the functions that tells us what program file the error occurred in, and what line, and what functions were executing at the time. It also shows the line of code that caused the error

Operators in Python

1. Define Boolean expression with example. [Remembering]

A boolean expression is an expression that is either true or false. The values true and false are

called Boolean values.

Eg :5==6

False

True and False are special values that belongs to the type bool; they are not strings:

2. What are the different types of operators? [Remembering]

- Arithmetic Operator (+, -, *, /, %, **, //)
- Relational operator (== , !=, <>, < , > , <=, >=)
- Assignment Operator (=, += , *= , - =, /=, %= ,**=)
- Logical Operator (AND, OR, NOT)
- Membership Operator (in, not in)
- Bitwise Operator (& (and), | (or) , ^ (binary Xor), ~(binary 1's complement , << (binary left shift), >> (binary right shift))
- Identity(is, is not)

3. Explain modulus operator with example. [Remembering]

The modulus operator works on integers and yields the remainder when the first operand is divided by the second. In Python, the modulus operator is a percent sign (%). The syntax is the same as for other operators:

Eg:

```
>>> remainder = 7 % 3
>>> print remainder
      1
```

So 7 divided by 3 is 2 with 1 left over.

4. Explain relational operators. [Understanding]

The == operator is one of the relational operators; the others are:

X! = y # x is not equal to y

x > y # x is greater than y

x < y # x is less than y

x >= y # x is greater than or equal to y

x <= y # x is less than or equal to y

5. Explain Logical operators. [Understanding]

There are three logical operators: and, or, and not. For example, x > 0 and x < 10 is true only if x is greater than 0 and less than 10. n%2 == 0 or n%3 == 0 is true if either of the conditions is true, that is, if the number is divisible by 2 or 3. Finally, the not operator negates a Boolean expression, so not(x > y) is true if x > y is false, that is, if x is less than or equal to y. Non-zero number is said to be true in Boolean expressions.

6. What is conditional execution? [Remembering]

The ability to check the condition and change the behavior of the program accordingly is called conditional execution. Example:

If statement:

The simplest form of if statement is:

Syntax:

if

statement:

Eg:

```
if x > 0:
print 'x is positive'
```

The boolean expression after 'if' is called the condition. If it is true, then the indented

statement gets executed. If not, nothing happens.

7. What is alternative execution? [Remembering]

A second form of if statement is alternative execution, that is, if …else, where there are two possibilities and the condition determines which one to execute.

Eg:

```
if x%2 == 0:
    print 'x is even'
else:
    print 'x is odd'
```

8. What are chained conditionals? [Remembering]

Sometimes there are more than two possibilities and we need more than two branches. One way to express a computation like that is a chained conditional:

Eg:

```
if x < y:

    print 'x is less than y'
elif x > y:
    print 'x is greater than y'
```

else:

print 'x and y are equal'

elif is an abbreviation of “else if.” Again, exactly one branch will be executed. There is no limit on the number of elif statements. If there is an else clause, it has to be at the end, but there doesn’t have to be one.

9. Explain while loop with example. [Remembering]

Eg:

```
def countdown(n):
    while n > 0:
        print n
        n = n-1
        print 'Blastoff!'
```

More formally, here is the flow of execution for a while statement:

1. Evaluate the condition, yielding True or False.
2. If the condition is false, exit the while statement and continue execution at the next statement.
3. If the condition is true, execute the body and then go back to step 1

10. Explain ‘for loop’ with example. [Remembering]

The general form of a for statement is

Syntax:

```
for variable in sequence:
    code block
```

Eg:

```
x = 4
for i in range(0, x):
print i
```

10. What is a break statement? [Remembering]

When a break statement is encountered inside a loop, the loop is immediately terminated and the program control resumes at the next statement following the loop.

Eg:

```
while True:
```

```
line = raw_input('>')
if line == 'done':
        break
        print line
        print'Done!'
```

11. What is a continue statement? [Remembering]

The continue statement works somewhat like a break statement. Instead of forcing termination, it forces the next iteration of the loop to take place, skipping any code in between.

Eg:

```
fornum in range(2,10):
if num%2==0;
        print “Found an even number”, num
        continue
print “Found a number”, num
```

12. Compare return value and composition. [Remembering]

13. Return Value:

Return gives back or replies to the caller of the function. The return statement causes our function to exit and hand over back a value to its caller.

Eg:

```
def area(radius):
        temp = math.pi *
radius**2 return temp
```

Composition:

Calling one function from another is called composition.

Eg:

```
defcircle_area(xc, yc, xp, yp):
        radius = distance(xc,
        yc, xp, yp) result =
        area(radius)
```

return result

14. What is recursion? [Understanding]

The process in which a function calls itself directly or indirectly is called recursion and the corresponding function is called as recursive function.

Eg:

```
def factorial(n):
    if n == 1:
        return 1
    else:
        return n * factorial(n-1)
```

14. Explain global and local scope. [Remembering]

The scope of a variable refers to the places that we can see or access a variable. If we define a variable on the top of the script or module, the variable is called global variable. The variables that are defined inside a class or function is called local variable.

Eg:

```
defmy_local():
        a=10
        print(“This is local variable”)
```

Eg:

```
a=10
defmy_global():
        print(“This is global variable”)
```

15. Compare string and string slices. [Remembering]

A string is a sequence of character.

Eg: fruit = ‘banana’

String Slices :

A segment of a string is called string slice, selecting a slice is similar to selecting a character. **Eg:** >>> s ='Monty Python'

16. Define string immutability. [Remembering]

Python strings are immutable. ‘a’ is not a string. It is a variable with string value. We can’t mutate the string but can change what value of the variable to a new string

Program:	Output:
a = “foo”	#foofoo
# a now points to foo	#foo
b=a	It is observed that ‘b’ hasn’t changed even
# b now points to the same foo that a points to	though ‘*a’* has changed.
a=a+a	

17.Mention a few string functions. [Understanding]

s.captilize() – Capitalizes first character of
string s.count(sub) – Count number of
occurrences of sub in string

s.lower() – converts a string to lower case

s.split() – returns a list of words in string

18. What are string methods? [Understanding]

A method is similar to a function—it takes arguments and returns a value—but the syntax is different. For example, the method upper takes a string and returns a new string with all uppercase letters:

Instead of the function syntax upper(word), it uses the method syntax word.upper() .>>> word = 'banana'

>>> new_word = word.upper()
>>> print new_word
BANANA

19. Explain about string module. [Remembering]

The string module contains number of useful constants and classes, as well as some deprecated legacy functions that are also available as methods on strings.

Eg:

Program:	**Output:**
import string text = "Monty Python's Flying Circus" print "upper", "=>", string.upper(text) print "lower", "=>", string.lower(text) print "split", "=>", string.split(text) print "join", "=>", string.join(string.split(text), "+") print "replace", "=>",string.replace(text, "Python", "Java") print "find", "=>", string.find(text, "Python"), string.find(text, "Java") print "count", "=>", string.count(text, "n")	upper => MONTY PYTHON'S FLYING CIRCUS lower =>monty python's flying circus split => ['Monty', "Python's", 'Flying', 'Circus'] join =>Monty+Python's+Flying+Circus replace => Monty Java's Flying Circus find => 6 -1 count => 3

20. What is the purpose of pass statement? [Remembering]

Using a pass statement is an explicit way of telling the interpreter to do nothing.

Eg:

```
def bar():
    pass
```

If the function bar() is called, it does absolutely nothing

LISTS, TUPLES AND DICTIONARIES

1. What is a list? [Remembering]

A list is an ordered set of values, where each value is identified by an index. The values that make up a list are called its elements. Lists are similar to strings, which are ordered sets of characters, except that the elements of a list can have any type.

2. Solve a)[0] * 4 and b) [1, 2, 3] * 3. [Understanding]

```
>>> [0] * 4 [0, 0, 0, 0]

>>> [1, 2, 3] * 3 [1, 2, 3, 1, 2, 3, 1, 2, 3]
```

3.Let list = ['a', 'b', 'c', 'd', 'e', 'f']. Find a) list[1:3] b) t[:4] c) t[3:] .

```
>>> list = ['a', 'b', 'c', 'd', 'e', 'f']

>>> list[1:3] ['b', 'c']

>>> list[:4] ['a', 'b', 'c', 'd']

>>> list[3:] ['d', 'e', 'f']
```

4. Mention any 5 list methods. [Remembering]

append() ,extend () ,sort(), pop(),index(),insert and remove()

5. State the difference between lists and dictionary. [Understanding]

List is a mutable type meaning that it can be modified whereas dictionary is immutable and is a key value store. Dictionary is not ordered and it requires that the keys are hashable whereas list can store a sequence of objects in a certain order.

6. What is List mutability in Python? Give an example. [Understanding]

Python represents all its data as objects. Some of these objects like **lists** and dictionaries are **mutable**, i.e., their content can be changed without changing their identity. Other objects like integers, floats, strings and tuples are objects that cannot be changed. Example:

>>> numbers = [17, 123]

>>> numbers[1] = 5

>>> print numbers [17, 5]

7. What is aliasing in list? Give an example. [Remembering]

An object with more than one reference has more than one name, then the object is said to be aliased. Example: If *a* refers to an object and we assign b = a, then both variables refer to the same object:

>>> a = [1, 2, 3]

>>> b = a

>>> b is a True

8. Define cloning in list. [Remembering]

In order to modify a list and also keep a copy of the original, it is required to make a copy of the list itself, not just the reference. This process is called cloning, to avoid the ambiguity of the word "copy".

9. Explain List parameters with an example. [Remembering]

Passing a list as an argument actually passes a reference to the list, not a copy of the list. For example, the function head takes a list as an argument and returns the first element:

```
def head(list):

return list[0]
```

output:

```
>>> numbers = [1, 2, 3]

>>> head(numbers)
```

10. Write a program in Python to delete first element from a list. [Remembering]

defdeleteHead(list): del list[0]

Here's how deleteHead is used:

>>> numbers = [1, 2, 3]

>>> deleteHead(numbers)

>>> print numbers [2, 3]

11. Write a program in Python returns a list that contains all but the first element of the given list. [Remembering]

Def tail(list): return
list[1:]

Here's how tail is used:

>>>numbers = [1, 2, 3]

>>>rest = tail(numbers)

>>> print rest [2, 3]

12. What is the benefit of using tuple assignment in Python? [Understanding]

It is often useful to swap the values of two variables. With conventional assignments a temporary variable would be used. For example, to swap a and b:

>>> temp = a
>>> a = b
>>> b = temp

>>> a, b = b, a

13. Define key-value pairs. [Remembering]

The elements of a dictionary appear in a comma-separated list. Each entry contains an index and a value separated by a colon. In a dictionary, the indices are called keys, so the elements are called key-value pairs.

14. Define dictionary with an example. [Remembering]

A dictionary is an associative array (also known as hashes). Any key of the dictionary is associated

(or mapped) to a value. The values of a dictionary can be any Python data type. So dictionaries are

unordered key-value-pairs.

Example:

```
>>> eng2sp = {}  # empty dictionary

 >>> eng2sp['one'] = 'uno'
>>> eng2sp['two'] = 'dos'
```

15. How to return tuples as values? [Remembering]

A function can only return one value, but if the value is a tuple, the effect is the same as returning multiple values. For example, if we want to divide two integers and compute the quotient and remainder, it is inefficient to compute x/y and then x%y. It is better to compute them both at the same time.

```
>>> t = divmod(7, 3)
 >>> print t (2, 1)
```

16. List two dictionary operations. [Remembering]

- Del -removes key-value pairs from a dictionary
- Len - returns the number of key-value pairs

17.Define dictionary methods with an example. [Remembering]

A method is similar to a function—it takes arguments and returns a value— but the syntax is different. For example, the keys method takes a dictionary and returns a list of the keys that appear, but instead of the function syntax keys(eng2sp), method syntax eng2sp.keys() is used.

```
 >>> eng2sp.keys() ['one', 'three', 'two']
```

18.Define List Comprehension. [Understanding]

List comprehensions apply an **arbitrary expression** to items in an iterable rather than applying function. It provides a compact way of mapping a list into another list by applying a function to each of the elements of the list.

19.Write a Python program to swap two variables.[Understanding]

```
x = 5

y = 10

temp = x

x = y

y = temp

print('The value of x after swapping: {}'.format(x))

print('The value of y after swapping: {}'.format(y))
```

20. Write the syntax for list comprehension. [Remembering]

The list comprehension starts with a '**[' and ']'**, to help us remember that the result is going to be a list. The basic syntax is [expression for item in list if conditional].

Unit –II
Functions and Exception

1. **What is meant by functions in python? [Understanding]**
   A function can be defined as the organized block of reusable code which can be called whenever required.

2. List the Advantages of functions in python. [Understanding]

- We can avoid rewriting same logic/code again and again in a program.
- We can call python functions any number of times in a program and from any place in a program.
- We can track a large python program easily when it is divided into multiple functions.
- Reusability is the main achievement of python functions.
- However, Function calling is always overhead in a python program.

3. Write the syntax for python. [Remembering]

```
def my_function():
    function-suite
    return <expression>
```

The function block is started with the colon (:) and all the same level block statements remain at the same indentation.

A function can accept any number of parameters that must be the same in the definition and function calling.

4. List the types of arguments. [Understanding]

1. Required arguments
2. Keyword arguments
3. Default arguments
4. Variable-length arguments

5. What are lambda functions in Python? [Understanding]

In Python, anonymous function is a function that is defined without a name.
While normal functions are defined using the def keyword, in Python anonymous functions are defined using the lambda keyword.

Hence, anonymous functions are also called lambda functions.

6. How to use lambda Functions in Python? [Remembering]

A lambda function in python has the following syntax.

Syntax

```
lambda arguments: expression
```

Lambda functions can have any number of arguments but only one expression. The expression is evaluated and returned. Lambda functions can be used wherever function objects are required.

7. Write the example of Lamda function in python. [Understanding]

```
double = lambda x: x * 2

print(double(5))

# Output: 10
```

8. Define Decorators. [Understanding]

A decorator is a function that takes a function as an argument and returns a function as a return value.

```
defadd(a,b):
returna+b
```

9. Write the syntax for nested decorator. [Analysis]

You can stack decorator expressions. The result is like calling each decorator in order, from bottom to top:

```
@decorator_two
@decorator_one
deffunc(x):
pass

deffunc(x):
pass
func=decorator_two(decorator_one(func))
```

10. How can you declare generator in python? [Apply]

Generators give you the iterator immediately:

- no access to the underlying data ... if it even exists

defa_generator_function(params):
some_stuff
yieldsomething

Generator functions “yield” a value, rather than returning a value.

11. Define exception in python. [Remembering]

An **exception** is an event, which occurs during the execution of a program that disrupts the normal flow of the program's instructions. In general, when a **Python** script encounters a situation that it cannot cope with, it raises an **exception**. An **exception** is a **Python** object that represents an error.

12.Why lambda forms in python does not have statements? [Understanding]

A lambda form in python does not have statements as it is used to make new function object and then return them at runtime.

13.In Python what are iterators? [Understanding]

In Python, iterators are used to iterate a group of elements, containers like list.

14. What is unittest in Python? [Remembering]

A unit testing framework in Python is known as unittest. It supports sharing of setups, automation testing, shutdown code for tests, aggregation of tests into collections etc.

15. In Python what is slicing? [Remembering]

A mechanism to select a range of items from sequence types like list, tuple, strings etc. is known as slicing.

16. What are generators in Python? [Understanding]

The way of implementing iterators are known as generators. It is a normal function except that it yields expression in the function.

Unit –III

Modules and Class

1.What is module and package in Python? [Understanding]

In Python, module is the way to structure program. Each Python program file is a module, which imports other modules like objects and attributes.

The folder of Python program is a package of modules. A package can have modules or subfolders.

2.How to import modules in python? [Remembering]

Modules can be imported using the **import** keyword. You can import modules in three ways-

```
1 import array          #importing using the original module name
2 import array as arr    # importing using an alias name
3 from array import *    #imports everything present in the array module
```

3) Mention what are the rules for local and global variables in Python? [Remembering]

Local variables: If a variable is assigned a new value anywhere within the function's body, it's assumed to be local.

Global variables: Those variables that are only referenced inside a function are implicitly global.

4) Explain how can you make a Python Script executable on Unix? [Understanding]

To make a Python Script executable on Unix, you need to do two things,

- Script file's mode must be executable and
- the first line must begin with # (#!/usr/local/bin/python)

5) Explain how can you generate random numbers in Python? [Understanding]

To generate random numbers in Python, you need to import command as

import random

random.random()

This returns a random floating point number in the range [0,1)

6) Explain how can you access a module written in Python from C?[Analysis]

You can access a module written in Python from C by following method,

Module = =PyImport_ImportModule("<modulename>");

7) How can you share global variables across modules?[Apply]

To share global variables across modules within a single program, create a special module. Import the config module in all modules of your application. The module will be available as a global variable across modules.

Does Python have OOps concepts?

Ans: Python is an object-oriented programming language. This means that any program can be solved in python by creating an object model. However, Python can be treated as procedural as well as structural language.

8.How is Multithreading achieved in Python? [Understanding]

Ans:

1. Python has a multi-threading package but if you want to multi-thread to speed your code up, then it's usually not a good idea to use it.
2. Python has a construct called the Global Interpreter Lock (GIL). The GIL makes sure that only one of your 'threads' can execute at any one time. A thread acquires the GIL, does a little work, then passes the GIL onto the next thread.
3. This happens very quickly so to the human eye it may seem like your threads are executing in parallel, but they are really just taking turns using the same CPU core.
4. All this GIL passing adds overhead to execution. This means that if you want to make your code run faster then using the threading package often isn't a good idea.

9. Explain Inheritance in Python with an example. [Remembering]

Ans: Inheritance allows One class to gain all the members(say attributes and methods) of another class. Inheritance provides code reusability, makes it easier to create and maintain an application. The class from which we are inheriting is called super-class and the class that is inherited is called a derived / child class.

They are different types of inheritance supported by Python:

1. Single Inheritance – where a derived class acquires the members of a single super class.
2. Multi-level inheritance – a derived class d1 in inherited from base class base1, and d2 are inherited from base2.
3. Hierarchical inheritance – from one base class you can inherit any number of child classes
4. Multiple inheritance – a derived class is inherited from more than one base class.

10.Explain Inheritance in Python with an example.[Analysis]

Ans: Inheritance allows One class to gain all the members(say attributes and methods) of another class. Inheritance provides code reusability, makes it easier to create and maintain an application. The class from which we are inheriting is called super-class and the class that is inherited is called a derived / child class.

They are different types of inheritance supported by Python:

1. Single Inheritance – where a derived class acquires the members of a single super class.
2. Multi-level inheritance – a derived class d1 in inherited from base class base1, and d2 are inherited from base2.
3. Hierarchical inheritance – from one base class you can inherit any number of child classes
4. Multiple inheritance – a derived class is inherited from more than one base class.

11. How are classes created in Python? [Remembering]

Ans: Class in Python is created using the **class** keyword.

Example:

```
1  classEmployee:
2  def __init__(self, name):
3  self.name = name
4  E1=Employee("abc")
5  print(E1.name)
```

Output: abc

12. What is monkey patching in Python? [Remembering]

Ans: In Python, the term monkey patch only refers to dynamic modifications of a class or module at run-time.

13. Does python support multiple inheritance?

```
1  # m.py
2  classMyClass:
3  deff(self):
4  print"f()"
```

Ans: Multiple inheritance means that a class can be derived from more than one parent classes. Python does support multiple inheritance, unlike Java.

14.What is Polymorphism in Python? [Understanding]

Ans: Polymorphism means the ability to take multiple forms. So, for instance, if the parent class has a method named ABC then the child class also can have a method with the same name ABC having its own parameters and variables. Python allows polymorphism.

15.**Define encapsulation in Python? [Understanding]**

Ans: Encapsulation means binding the code and the data together. A Python class in an example of encapsulation.

16.How do you do data abstraction in Python? [Remembering]

Ans: Data Abstraction is providing only the required details and hiding the implementation from the world. It can be achieved in Python by using interfaces and abstract classes.

17.Does python make use of access specifiers? [Understanding]

Ans: Python does not deprive access to an instance variable or function. Python lays down the concept of prefixing the name of the variable, function or method with a single or double underscore to imitate the behavior of protected and private access specifiers.

18. How to create an empty class in Python? [Remembering]

Ans: An empty class is a class that does not have any code defined within its block. It can be created using the *pass* keyword. However, you can create objects of this class outside the class

itself. IN PYTHON THE PASS command does nothing when its executed. it's a null statement.

19. What does an object() do? [Remembering]
Ans: It returns a featureless object that is a base for all classes. Also, it does not take any parameters.

Unit IV

Files and Data Bases

1. What is a text file? [Understanding]

A text file is a file that contains printable characters and whitespace, organized in to lines separated by newline characters.

2. Write a python program that writes "Hello world" into a file. [Understanding]

```
f =open("ex88.txt",'w')

f.write("hello world")

f.close()
```

3. Write a python program that counts the number of words in a file. [Understanding]

```
f=open("test.txt","r")

content
=f.readline(20)
words
=content.split()

print(words)
```

4. What are the two arguments taken by the open() function? [Understanding]

The open function takes two arguments : name of the file and the mode of operation.

Example: f = open("test.dat","w")

5. What is a file object? [Understanding]

A file object allows us to use, access and manipulate all the user accessible files. It maintains the state about the file it has opened.

6.**What is database connection in Python Flask? [Understanding]**

Flask supports database powered application (RDBS). Such system requires creating a schema, which requires piping the shema.sql file into a sqlite3 command. So you need to install sqlite3 command in order to create or initiate the database in Flask.

Flask allows to request database in three ways

- before_request() : They are called before a request and pass no arguments
- after_request() : They are called after a request and pass the response that will be sent to the client
- teardown_request(): They are called in situation when exception is raised, and response are not guaranteed. They are called after the response been constructed. They are not allowed to modify the request, and their values are ignored.

7. Mention what is Flask-WTF and what are their features? [Understanding]

Flask-WTF offers simple integration with WTForms. Features include for Flask WTF are

- Integration with wtforms
- Secure form with csrf token
- Global csrf protection
- Internationalization integration
- Recaptcha supporting
- File upload that works with Flask Uploads

8) Explain what is the common way for the Flask script to work?[Analysis]

The common way for the flask script to work is

- Either it should be the import path for your application
- Or the path to a Python file

9) Explain how you can access sessions in Flask? [Understanding]

A session basically allows you to remember information from one request to another. In a flask, it uses a signed cookie so the user can look at the session contents and modify. The user can modify the session if only it has the secret key Flask.secret_key.

10) Is Flask an MVC model and if yes give an example showing MVC pattern for your application?[Analysis]

Basically, Flask is a minimalistic framework which behaves same as MVC framework. So MVC is a perfect fit for Flask, and the pattern for MVC we will consider for the following example

from flask import Flask app = Flask(_name_) @app.route("/") Def hello(): return "Hello World"	In this code your, • Configuration part will be from flask import Flask app = Flask(_name_) • View part will be

app.run(debug = True)	@app.route("/") Def hello(): return "Hello World" • While you model or main part will be app.run(debug = True)

11 .What is CSV? [Understanding]

CSV is a simple **file format** used to store tabular data, such as a spreadsheet or database. **Files** in the **CSV format** can be imported to and exported from programs that store data in tables, such as Microsoft Excel or OpenOffice Calc. **CSV** stands for "comma-separated values".

12. How to define JSON in python? [Understanding]

Python has a built-in package called json, which can be used to work with JSON data.

```
import json

import json

# some JSON:
x =  '{ "name":"John", "age":30, "city":"New York"}'

# parse x:
y = json.loads(x)

# the result is a Python dictionary:
print(y["age"])
```

13.Write the syntax for creating python in mysql.[Apply]

```
import mysql.connector

mydb = mysql.connector.connect(
  host="localhost",
  user="yourusername",
  passwd="yourpassword")

mycursor = mydb.cursor()

mycursor.execute("CREATE DATABASE mydatabase")
```

14.Define SQL Lite. [Understanding]

SQLite is a C library that provides a lightweight disk-based database that doesn't require a separate server process and allows accessing the database using a nonstandard variant of the SQL query language. Some applications can use SQLite for internal data storage. It's also possible to prototype an application using SQLite and then port the code to a larger database such as PostgreSQL or Oracle.

Unit V

GUI AND WEB

1. **What Is A Graphical User Interface (GUI)? [Understanding]**

Graphical User Interface (GUI) is nothing but a desktop application which helps you to interact with the computers. They are used to perform different tasks in the desktops, laptops and other electronic devices.

- **GUI** apps like **Text-Editors** are used to create, read, update and delete different types of files.
- **GUI** apps like Sudoku, Chess and Solitaire are games which you can play.
- **GUI** apps like **Google Chrome, Firefox and Microsoft Edge** are used to
- browse through the **Internet**.

2. **What are the Python Libraries to Create Graphical User Interfaces ?** [Understanding]

Python has a plethora of libraries and these 4 stands out mainly when it comes to GUI. There are as follows:

- Kivy
- Python QT
- wxPython
- Tkinter

3. **What Is Tkinter?** [Understanding]

Tkinter is actually an inbuilt **Python** module used to create simple **GUI** apps. It is the most commonly used module for **GUI** apps in the **Python**.

You don't need to worry about installation of the **Tkinter** module as it comes with **Python** default.

4. **Write the syntax for Tkinter. [Understanding]**

```
1  importtkinter
2
3  window =tkinter.Tk()
4
5  # to rename the title of the window window.title("GUI")
6
```

```
# pack is used to show the object in the window

label =tkinter.Label(window, text ="Hello World!").pack()

window.mainloop()
```

5. **Define Tkinter Widgets.** [Understanding]

   Widgets are something like elements in the **HTML**.

6. List the different elements inTkinter Widgets. **[Understanding]**

   Canvas – **Canvas** is used to draw shapes in your **GUI**.

   Button – **Button** widget is used to place the buttons in the **Tkinter**.

   Checkbutton – **Checkbutton** is used to create the check buttons in your application. Note that you can select more than one option at a time.

   Entry – **Entry** widget is used to create input fields in the **GUI**.

   Frame – **Frame** is used as containers in the **Tkinter**.

   Label – **Label** is used to create a single line widgets like **text**, **images etc.**

   Menu – **Menu** is used to create menus in the **GUI**.

7. **What is Label Widget? [Understanding]**

As mentioned earlier, labels are used to create texts and images and all of that but it is important to note that it has to be a single line definition only.

Here's the code snippet:

```
l1 = Label (window, text="edureka!“ font=("Arial Bold", 50)

l1.grid (column=0, row=0)
```

8. Define Button Widget: [Understanding]

The button widget is very similar to the label widget. We create a variable and use the widget syntax to define what the button has to say.

```
bt =Button (window, text="Enter")
```

```
bt.grid (column=1, row=0)
```

9. How to define Geometry Management?[Apply]

All widgets in the **Tkinter** will have some geometry measurements. These measurements give you to organize the widgets and their parent frames, windows and so on.

10 .What are the classes in Geometry Manager classes? [Understanding]

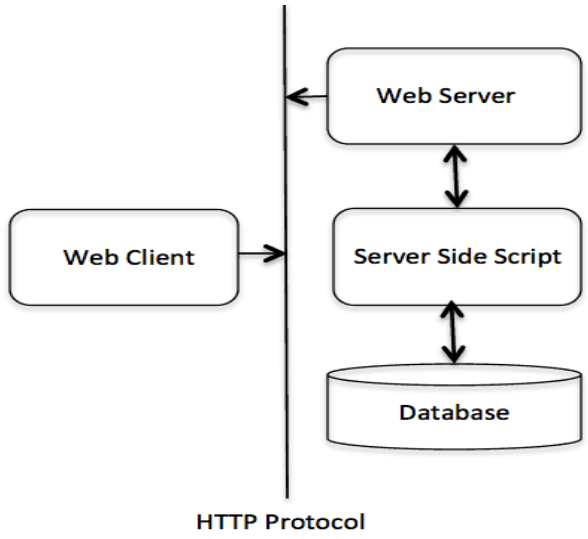
Tkinter has the following three Geometry Manager classes.

- **pack()**:- It organizes the widgets in the block, which mean it occupies the entire available width. It's a standard method to show the widgets in the window.
- **grid()**:- It organizes the widgets in table-like structure.
- **place()**:- It's used to place the widgets at a specific position you want.

11. What is CGI?[Analysis]

- The Common Gateway Interface, or CGI, is a standard for external gateway programs to interface with information servers such as HTTP servers.
- The current version is CGI/1.1 and CGI/1.2 is under progress.

12. Draw the CGI Architecture Diagram. ?[Apply]



13. List the CGI Environment variables. [Remembering]

- CONTENT_TYPE
- CONTENT-LENGTH
- HTTP_COOKIE
- HTTP_USER_AGENT
- PATH_INFO
- QUERY_STRING

14. Compare GET and POST requests using Python. [Remembering]

This post discusses two HTTP (Hypertext Transfer Protocol) request methods GET and POST requests in Python and their implementation in python.

1. **GET** : to request data from the server.
2. **POST** : to submit data to be processed to the server.

15. What is HTTP? [Remembering]

HTTP is a set of protocols designed to enable communication between clients and servers. It works as a request-response protocol between a client and server.
A web browser may be the client, and an application on a computer that hosts a web site may be the server.

16 marks Questions

Unit –I

1. Explain the data types in python. **[Remembering]**
2. Write short notes on types of operators in python with appropriate example. **[Remembering]**
3. Explain briefly constant, variables, expression, keywords and statements available in python.**[Apply]**
4. Write the following python programs. **[Remembering]**
   a. Exchange the value of two variables.
   b.Circulate the value of n variables
   c. Test whether a given year is leap year or not
   d. to find the sum of n natural numbers
   e. To find whether a given number is Armstrong number or not
   f. To print Fibonacci series
   g. To find factorial of a given number
   h. To convert Celsius to Fahrenheit

5. What is the difference between Lists, dictionaries and Tuples? **[Remembering]**

6. Explain the purpose of loop structure in a programming language. **[Understanding]**

7. Describe the syntax and semantics of any two loop structures in python. **[Apply]**

8. Explain the features of a dictionary. **[Remembering]**

9. What type of conditional structures are present in a programming language? **[Apply]**

10. Write a program to check whether entered string is palindrome or not**.[Analysis]**

Unit –II

1.Explain the function arguments in python. **Remembering]**

2. Explain call by value and call by reference in python. **[Understanding]**

3. Briefly explain about function prototypes. **[Analysis]** .

4. Discuss the following Lamda functions: **[Analysis]**

   i) Generators

   ii) Decorators

5. Write short notes on exception handling. **[Apply]**
6. Write a python program for Calendars and Clocks operation. **Remembering]**

Unit –III

1. What are modules in Python? Explain. **Remembering]**
2. Explain in details about namespaces and scoping. **Remembering]**
3. Explain about the import statement in modules. **[Understanding]**
4. What are packages? Give an example of package creation in Python. **[Understanding]**
5. Write short notes on OOPs concepts. **[Analysis]**
6. Write a program for method override. **[Analysis]**
7. Discuss the following: **[Apply]**
   i) Get and Set Attribute Values
   ii) Name Mangling
   iii) Method Types
   iv) Duck Typing
   v) Relationships.

Unit –IV

1. Explain in detail about Python Files, its types, functions and operations that can be performed on files with examples. **Remembering]**
2. Illustrate the directory operations. **[Apply]**
3. Describe the following Reading and Writing in Structured Files:i) CSV ii) JSON **[Understanding]**
4. Write short notes on Data manipulation using Oracle. **[Apply]**
5. Compare MySQL and SQLite. **Remembering]**

Unit –V

1. Explain in detail about Tkinter with an example. **[Understanding]**
2. Describe the various events in python. **Remembering]**
3. Write short notes on CGI Programming.**[Analysis]**
4. Illustrate the GET and POST Methods with an example. **[Understanding]**
5. Discuss the file uploading operation in python. **[Apply]**