PYTHON

Unit 1 and 2 Theory Questions

1. What are various ways of commenting in Python?
2. What is a Docstring?
3. What is Identation? Define Suite.
4. What are Variables?
5. What are Keywords?
6. What are Identifiers?
7. How do you declare a variable in python? What are the do's and don't's of declaring a variable? How do you find the data type of a declared variable?
8. What is a Data Type? How many datatypes are available in python? List them.
9. What is Typecasting? Define its types.
10. What are Operators and Operands in Python? List their types.
11. Differentiate between '== 'and 'is'. Is there any substitution of '===' in Python?
12. Define Ternary Operator? Write a program in Python to check if a number is greater than 5 and print suitable statements using ternary operators.
13. What are Control Structures/Decision Making Statements in Python? Name them all.
14. Define Loops in Python and list their kinds. Mention 3 advantages.
15. Differentiate between break, continue, and pass statements.
16. Which statement with the 'for' loop can be executed only when all the iterations are exhausted while keeping in mind no inclusion of 'break' statement?
16. Define the following with syntax and one example each.
a. range()
b. len()
17. What are Strings in Python? Is indexing allowed in strings? Support your answer with an example.
18. What do you mean by slicing in Strings? Define the following functions with syntax and one example each:
a. upper()
b. lower()
c. strip()
d. replace()
e. split()
19. How do you concatenate strings in Python? Mention the format specifier used for referencing strings. How do you escape single or double quotes

present in a string?
20. Explain the following with one example each:
a. Syntactical Errors
b. Logical Errors
c. Runtime Errors
d. Compilation Errors
e. Semantic Errors
21. How do you format an output in Python? Illustrate all the ways with examples.
22. What is meant by 'Positional Arguments' and 'Keyword Arguments' while using format()?
23. How do you specify a raw string in Python?
24. Name the following operators in respect to string operators in python:
a. '+'
b. '*'
c. '[]'
d. '[:]'
e. in/not in
f. '%
25. Define all the mutable and ordered datatypes in Python with examples.
26. Define the immutable and unordered datatype in Python with an example.
27. Deinfe the following List methods:
a. remove()
b. append()
c. extend()
d. insert()
e. index()
f. count()
g. clear()
h. copy()
I. pop()
j. reverse()
k. remove()
l. sort()
27. Define Tuples. What is packing and unpacking in Tuples? Give an example.
28. How do a Tuple differs from a List? Mention available in-built methods for tuples in python.
29. Write a Python program to create an empty list, a tuple, a set and a dictionary.

30. Elaborate the following:
a. update()
b. union()
c. intersection() and intersection_update()
d. difference() and difference_update()
e. add()
f. discard()
g. issubset()
h. issuperset()
I. pop()
j. remove()
k. isdisjoint()
31. What is a Dictionary in Python? How do you access it? Explain the use of get().
32. DIfferentiate between pop() and popitem() with respect to sets in python.
33. Define items(), keys(), and values().
34. Explain list, set, dictionary comprehension each with an example.
35. What are Functions? Also define the types.
36. Define Arguements and its various types.
37. What is meant by Recursion? Write a python code for finding factorial of a user entered number using recursion.
38. Differentiate between Traditional Functions and Lambda Function.
39. Differentiate between map() and filter().
40. How is sort() different from sorted()? List the parameters they allow.
41. How is an Inner Function different from a Closure Function in Python?
42. What do you mean by a Python Module?
43. Differentiate between:
a. import module1
b. from module1 import attribute
c. from module1 import *
44. What is meant by Alias in renaming a module?
45. What are Packages in Python? Discuss the importance of '__init__'.
46. Express the purpose of using os module. State the purpose of listdir(), rmdir(), chdir(), and getcwd(), mkdir(), remove() functions from this module by showing their usage.

Unit 3, 4 and 5 Theory Questions

1. What is meant by Exception Handling in Python?

2. Differentiate between try, except, else, and finally.
3. How do you raise an error in Exception? Validate your answer with syntax and an example.
4. Define the following:
a. NameError
b. RangeError
c. ArithmeticError
d. ZeroDivisionError
e. EOFError
f. TypeError
g. KeyError
5. How do you use 'assert' statement to raise AssertionError in Python?
6. What basic operations can be performed on a file in Python? Mention all the methods of accessing a file in different modes.
7. Differentiate between read() and readline() in respect to file handling in Python.
8. Write a python program to check if a file named 'demo.txt' exists and delete it otherwise print suitable statement.
9. Discuss the use of tell() and seek().
10. Implement stack using lists in Python.
11. Differentiate between Class and Object. Define Encapsulation.
12. What are constructors in Python? Define its types. Discuss the significance of '__init__()'.
13. Why constructor overloading is not allowed in Python?
14. What are built-in class methods and attributes?
15. Define destructor using a python code.
16. Define Inheritance and discuss its various levels.
17. How abstraction is achieved?
18. Give examples to support Polymorphism in python.
19. How does the use of '_' effects the scope of a variable in a class?
20. What are the two ways of defining a static method?
21. This different behavior of a single operator for different types of operands is called?
22. List some special functions which facilitate operator overloading in python.
24. Discuss on regular expressions. Express how to determine whether an email address entered by a user is valid using Python ‘re’ module.
25. State the meaning and usage of meta characters ‘*’ and ‘^’ in regular expressions.

Unit 6 Theory Questions

1. What is NumPy package for? What are the various ways of importing it?
2. What is the difference between 'ndarray' and 'matlib'?
3. How do you create an array using NumPy? Mention all the various methods.
4. 'NumPy is written completely in Python'. Do you agree with this statement? Support your answer with suitable statements.
5. Write a sentence about the following:
a. ndim
b. shape()
c. size()
d. random()
e. reshape()
f. repeat()
g. astype()
6. Differentiate between arange() and linspace(). Support your answer by examples.
7. How do you slice an array in NumPy?
8. How do you join and split arrays in NumPy.
9. Briefly explain Broadcasting.
10. Define Data Types in NumPy.
11. What is the difference between Copy and View.
12. Discuss numpy array search. How searchsorted() works.
13. How do you sort a 2D array?
14. Mention the submodule imported while importing matplotlib.
15. What is the key difference between bar() and barh().
16. Define the following:
a. title()
b. legend()
c. xlabel()
d. ylabel()
e. show()
f. savefig()
17. Mention all the kinds of charts that can be generated using matplotlib.
18. Mention the use of attribute explode in Pie charts.
19. What structures can be defined by using the Pandas package?
20. Define the following:
a. tail()
b. head()
c. shape()
d. size()

21. How do you iterate over columns and rows in Pandas?
22. Consider the following data 'demo.xlsx':

	Molar	Incisors	Pre-Molars
1	34	43	12
2	35	44	13
3	36	45	14

a. How would you display data for the column 'Molar' and row index 2?
b. Read the above file in pandas and save it as a csv file.
c. Download the above file as a xlsx file.

SOLVED LAST YEAR PROGRAMS

1. Consider a file containing record of students' performance in a test. Individual records are arranged in each row as per roll number, student's name and marks in the order given below:

1 Abhishek 23
2 Sandhya 45
...
10 Yogesh 61

Write a function to print the name of the student with the highest marks.

```
import pandas as pd

data = {
    "Roll No.": [1,2,3,4,5,6,7,8,9,10],
    "Names": ['Abhishek','Sandhaya', 'Rajneesh', 'Vaani', 'Palak', 'Pallavi',
              'Rohan','Varun','Vishal','Yogesh'],
    "Marks": [23, 45, 56, 23, 89, 45, 12, 13, 14, 61],
}

df = pd.DataFrame(data, index=[x for x in range(1,11)])

def max_marks(dataframe,max_value_col,result_col):
    """dataframe: Pass the dataframe you are using.
       max_value_col: Name of the column you want to find max value from the
                      database.
    result_col: Name of the column you want to print in respect to max value."""
    return dataframe.loc[dataframe[max_value_col].idxmax(), result_col]

max_marks(df, 'Marks', 'Names')
```

2. Employee record of a company is arranged under three heads; name, age, and salary. Utilize numpy, pandas, and matplotlib modules and write the code to:

- Prepare such a record for 10 employees.
- Create a dataframe of shape [10, 3] for the prepared record and show the first four rows of the dataframe.
- Use the dataframe to plot two graphs – name against salary and age against salary.

```
import pandas as pd
import numpy as np

Age=np.array([23, 25, 26, 23, 29, 35, 32, 33, 34, 31])
Salary=np.array([12000, 23000, 21000, 34000, 24000, 25000, 28000, 30000, 45000,
35000])
Name=['Abhishek','Sandhaya', 'Rajneesh', 'Vaani', 'Palak', 'Pallavi',
'Rohan','Varun','Vishal','Yogesh']
employee_record = {
```

```
    "Name": Name,
    "Age": Age,
    "Salary": Salary
}
data = pd.DataFrame(employee_record, index=[x for x in range(1,11)])
print(data.head(4))

plt.figure(figsize=(10, 4))

plt.plot(data['Name'], data['Salary'], ':', color='magenta', marker="P")
plt.title("Employee")
plt.xlabel("Name")
plt.ylabel("Salary")
plt.show()
```

ALTERNATIVE

```
import pandas as pd

employee_record = {
    "Name": ['Abhishek','Sandhaya', 'Rajneesh', 'Vaani', 'Palak', 'Pallavi',
'Rohan','Varun','Vishal','Yogesh'],
    "Age": [23, 25, 26, 23, 29, 35, 32, 33, 34, 31],
    "Salary": [12000, 23000, 21000, 34000, 24000, 25000, 28000, 30000, 45000,
35000]
    }
data = pd.DataFrame(employee_record, index=[x for x in range(1,11)])
print(data.head(4))

plt.figure(figsize=(10, 4))

plt.plot(data['Name'], data['Salary'], ':', color='magenta', marker="P")
plt.title("Employee")
plt.xlabel("Name")
plt.ylabel("Salary")
plt.show()
```

3. Consider two simple lists of integers,
   X = [5, 2, 9, 4, 7] and,
   Y = [10, 5, 8, 4, 2].
   Assuming them to be the horizontal and vertical axes values, respectively, write the Python code to draw a line graph.

```
from matplotlib.pyplot import *

X = [5, 2, 9, 4, 7]
```

```
Y = [10, 5, 8, 4, 2]
plot(X, Y, color='blue')
show()
```

4. Write a Python code to add two 3×3 matrices using appropriate sequence.

```
from numpy import *
matrix1 = arange(1,10).reshape(3,3)
matrix2 = arange(10,100,10).reshape(3,3)
matrix1+matrix2
```

6. Write a Python program to sort a list, li = [['Java', 1995], ['C++', 1983], ['Python', 1989]], by year using lambda function

```
li = [["Java", 1995], ["C++", 1983], ["Python", 1989]]
sorted_li = sorted(li, key=lambda x: x[1])
print(sorted_li)
```

7. Consider that a class BankAccount is to be inherited by the two subclasses; SavingAccount and CurrentAccount. Write a Python program to implement the given inheritance scenario by mentioning appropriate members for each class. Instantiate these classes to demonstrate object polymorphism. Finally, show a sample run.

```
class BankAccount:
    def __init__(self, branch):
        self.branch = branch

    def displayBranch(self):
        return self.branch.upper()

class SavingAccount(BankAccount):

    acc_type = 'SAVINGS'

    def __init__(self, name, id, acc_no, branch):
        super().__init__(branch)
        self.name = name
        self.id = id
        self.acc_no = acc_no
        self.type = 'Savings'
```

```
    def display(self):
        print(f"Bank Branch: {self.displayBranch()}")
        print(f"Account Number: {self.acc_no}")
        print(f"Account Holder Name: {self.name}")
        print(f"Account ID: {self.id}")
        print(f"Account Type: {self.acc_type}\n")

class CurrentAccount(BankAccount):

    acc_type = 'CURRENT'

    def __init__(self, name, id, acc_no, branch):
        super().__init__(branch)
        self.name = name
        self.id = id
        self.acc_no = acc_no
        self.type = 'Current'

    def display(self):
        print(f"Bank Branch: {self.displayBranch()}")
        print(f"Account Number: {self.acc_no}")
        print(f"Account Holder Name: {self.name}")
        print(f"Account ID: {self.id}")
        print(f"Account Type: {self.acc_type}\n")

user1=CurrentAccount('Joe', 567890, 500010012935, "Austria International")
user1.display()
user1=CurrentAccount('Castillo', 567890, 500010012935, "England Money House")
user1.display()
```

8. Express how to determine whether an email address entered by a user is valid using Python ‘re’ module.

```
import re

def is_valid_email(email):
    pattern = r'^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}$'

    match = re.match(pattern, email)

    return bool(match)
```

```
user_email = input("Enter an email address: ")

if is_valid_email(user_email):
    print("Valid email address!")
else:
    print("Invalid email address. Please enter a valid email.")
```

9. Write a Python program to:
    - read a file.
    - Add backslash (\) before every double quote in the file contents.
    - write it to another file in the same folder.
    - print the contents of both the files.

    For example:
    - If the first file is 'TestFile1.txt' with text as: Jack said, "Hello Pune".
    - The output of the file 'TestFile2.txt' should be: Jack said,\"Hello Pune\".

```
inputfile = open('file.txt', 'rt')
inputfile_contents = inputfile.read()

outputfile_contents = inputfile_contents.replace('"', '\\"')

outputfile_path = "newfile.txt"
outputfile = open("newfile.txt", 'wt')
outputfile.write(outputfile_contents)

outputfile.close()
inputfile.close()
for line in open("newfile.txt", "rt"):
    print(f"Modified File Content: {line}")

for line in open("file.txt", "rt"):
        print(f"Original File Content: {line}")

outputfile.close()
inputfile.close()
```