21AB201 PYTHON PROGRAMMING

Hours Per Week :

L	T	P	C
3	-	2	4

Total Hours :

L	T	P
45	-	30

Source :

https://content.techgig.com/thumb/msid-80100245,width-860,resizemode-4/8-Steps-to-master-Python-programming-for-Data-Science.jpg?88712

COURSE DESCRIPTION AND OBJECTIVES:

This course offers sufficient knowledge required to understand the fundamental concepts of Python programming language. This course enable the students to use different data structures like lists, dictionaries, tuples, sets etc. This course also enable the students to create reliable, modular and reusable programming and to create applications using Object-Oriented Programming approach

COURSE OUTCOMES:

Upon completion of the course, the student will be able to achieve the following outcomes:

COs	Course Outcomes
1	Analyze the usage of different data structures for practical and contemporary applications for a given problem
2	Develop functional, reliable and user friendly Python programs for given problem statement and constraints
3	Installing the python environment and related packages that are required for practical and contemporary applications
4	Design programs using the concepts of object oriented programming paradigm
5	Create simple programming solutions to the given problems

SKILLS:

- ✓ *Itentify suitable data types and data structures required an application*
- ✓ *Design structure and Object oriented programming solutions*
- ✓ *Design reliable and modular applications for a given problem*

UNIT– 1

Introduction: History of python, Features of python, Python installation on windows & Linux, Installing python packages via PIP, Running python commands using the REPL(Shell), Running

python scripts, Variables, Assignment, Keywords, Input-output, Indentation, Basic data types integers, booleans etc

Operators And Expressions: Operators- arithmetic operators, comparison (relational) operators, assignment operators, logical operators, bitwise operators, membership operators, identity operators; Expressions and order of evaluations

UNIT – 2

Control Structures: Conditional control structures - if, elif, else; Loop control structures - for, while, for... else, while..else, nested loops, break, continue, pass

Python Data Structures: Lists, Tuples, Dictionary - creation, accessing, basic operators and methods

UNIT – 3

Other Data Structures: Strings - creation, accessing, operators, methods; Sets - creation,

accessing, operators, methods; List comprehensions; Functions - defining functions, calling functions, passing arguments - keyword arguments, default arguments, variable-length arguments, anonymous functions (lambda), fruitful functions (function returning values), scope of the variables in a function - global and local variables

UNIT – 4

Modules: Creating modules, Import statement, From...import statement, Name spacing.

Errors and Exceptions: Difference between an error and exception, Handling exception, Try

except block, Raising exceptions, User defined exceptions

File Processing: Reading and Writing Files - creating a new file, writing to a file, reading files as text, opening and closing files, reading and writing, tell(), seek(), rename()

UNIT - 5

Object Oriented Programming in Python: Classes, 'self variable' methods, Constructor

method, Inheritance, Overriding methods, Data hiding

LABORATORY EXPERIMENTS

LIST OF EXPERIMENTS

1. Given an integer, n, depending on the value of n display suitable message:

   If n is odd, print Great.

   If n is even and in the inclusive range of 10 to 20 , print Not Great.

   If n is even and in the inclusive range of 21 to 30, print Great.

   If n is even and greater than 30, print Not Great.

2. Read integers a, b from standard input and display the following in different lines:

   The first line contains product of a and b.

   The second line contains the difference of the two numbers (a - b).

   The third line contains the exp(a,b).

   The fourth line contains the integer division of a, b.

   The fifth line contains the float division of a, b.

3. Read an integer n and display the following:

   1,2,3,4…….n.

   123…….n (without any spaces).

   1 2 3…….n (separated by spaces).

   1, 4, 9, 16, …….n2.

4. Use list comprehensions to solve Exercise 3.

5. Given the mid1 marks of python course for all the students, you are required to find the second highest mark. (Use lists).

6. Create two lists of integer numbers and write python code to generate a new list as described below (Make sure your code works on two lists of different sizes)

   a. A list with elements that are common in both the lists. (print without duplicates).
   b. A list with elements that are in either of the lists.
   c. A list that contains the first eleent of the first list and last element of the second list.
   d. A list that contains sum of elements of first list and sum of elements of second list.
   e. A list that contains largest number of both the lists.
   f. A list that contains least number of both the lists.

7. Use a dictionary to store the details of student name and marks in 3 subjects (data structures, DAA and python). Now display the details of students in ascending order of marks scored in python.

8. Write a Python program to generate a list, whose elements are tuples. Then display the list with tuples sorted in increasing order by the last element of each tuple.

   Sample Input: [(2, 5), (1, 2), (4, 4), (2, 3), (2, 1)]

   Expected output : [(2, 1), (1, 2), (2, 3), (4, 4), (2, 5)]

9. Write a program (function!) that reads a list and create a new list that contains all the elements of the original list without any duplicates.

   Write two different functions to do this - one using a loop and constructing a list, and another using sets.

10. Given a string and a substring, print the number of times that the substring occurs in the given string.

11. Read a string and write different python functions to achieve the following:

    a) To remove vowels in the given string using control transfer statements.

    b) To count number of uppercase and lowercase letters in the given string.

ACTIVITIES:

- o *Develop functional, reliable and user friendly Python programsfor given problem statement and constraints*
- o *Practice on structured and Object oriented programming solutions*
- o *Design programs using the concepts of object oriented programming paradigm*
- o *Practice on to create simple programming solutions to the given problems*

c) To remove all punctuation characters from given string.

d) To check whether the given string is palindrome or not.

e) To swap case of each letter in the string.

12. Write a python function to read a text document and do the following:

a) Count number of words in a given text.

b) To display the words in alphabetic order.

c) To count number of sentences in the text.

d) To display words of the document in reverse order.

e) Count the number of words with 3 characters.

13. Write a Python program to read data from two different text files. Display a line from first file followed by the corresponding line in the second file.

14. a. Define a function simple_int(p, t, r) that accepts 3 arguments and returns simple interest accordingly. call the function simple_int(p, t, r) with positional parameters.

b. Define a function simple_int(p, t, r) that accepts 3 arguments and returns simple interest accordingly. Call the function simple_int(p,t,r) with keyword arguments where the order of arguments does not matter.

c. Define a function simple_int(p, t, r=val) where r is set with a default value. Call the function simple_int() and pass all 3 parameters, pass only 2 parameters, pass the parameters in different order in function call.

d. Define a function mul(*args) which can handle variable length arguments. Call the function with 2, 5 and 7 parameters.

15. Write a function that receives 3 numbers and returns the median, i.e. the number that is not the min and not the max, but the one in between.

16. Define the following functions that are more robust to erroneous input/data

a) To divide two numbers (To handle ZeroDivisionError).

b) To read two integer numbers and display them (To handle ValueError).

c) To display elements of a list (To handle IndexError).

d) To open a file and display file contents (To handle FileNotFoundError).

17. Write python code to handle multiple exceptions and to raise an exception manually.

18. Write a Python class named Circle constructed by a radius and two methods which will compute the area and the perimeter of a circle. Define parameterized and nonparameterized constructors.

19. Write Python code to depict the following oops concepts:

a) Data hiding b) Inheritance c) Overriding.

20. Define a class Person and its two child classes: Male & Female. All classes have method "get Gender" which can print "Male" for Male class and "Female" for female class.
(To cover the above 20 topics in 30 hours of 15 Practical classes)

REFERENCE:

1 Vamsi Kurama, “Python Programming: A Modern Approach”, 1st edition, Pearson Publishers, 2018.

2 Mark Lutz, “Learning Python”, 5th edition, Orielly Publishers, 2013.

3 Allen Downey, “Think Python”, 2nd edition, Green Tea Press, 2016.

4 Ashok Namdev Kamthane and Amith Ashok Kamthane, “Programming and Problem Solving with Python”, 1st edition, McGraw Hill Education, 2016.

5 W. J. Chun, “Core Python Programming”, 3rd edition, Pearson Publishers, 2013.*Python Programming*