Annexure I

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course (if any)
		Lecture	Tutorial	Practical/ Practice		
DSC 01 **Object Oriented Programming using Python**	**4**	**3**	**0**	**1**	Pass in Class XII	

Course Objective

This course is designed as the first course that introduces object oriented programming concepts using Python to Computer Science students. The course focuses on the development of Python programming to solve problems of different domains using object- oriented programming paradigm.

Course Learning Outcomes

On successful completion of the course, students will be able to:

1. Understand the basics of programming language
2. Develop, document, and debug modular Python programs.
3. Apply suitable programming constructs and built-in data structures to solve a problem.
4. Use and apply various data objects in Python.
5. Use classes and objects in application programs and handle files.
6. apply OOPs concepts such as encapsulation, inheritance and polymorphism in writing programs.

Syllabus

Unit 1 **(4 hours)**

Introduction to Programming: Problem solving strategies; Structure of a Python program; Syntax and semantics; Executing simple programs in Python.

Unit 2 **(10 hours)**

Creating Python Programs: Identifiers and keywords; Literals, numbers, and strings; Operators; Expressions; Input/output statements; Defining functions; Control structures (conditional statements, loop control statements, break, continue and pass, exit function), default arguments.

Unit 3 **(15 hours)**

Built-in data structures: Mutable and immutable objects; Strings, built-in functions for string, string traversal, string operators and operations; Lists creation, traversal, slicing and splitting operations, passing list to a function; Tuples, sets, dictionaries and their operations.

Unit 4 **(10 hours)**

Object Oriented Programming: abstraction, encapsulation, objects, classes, methods, constructors, inheritance, polymorphism, static and dynamic binding, overloading, abstract classes, interfaces and packages.

Unit 5 **(6 hours)**

File and exception handling: File handling through libraries; Errors and exception handling.

References

1. Allen B. Downey,**Think Python: How to Think Like a Computer Scientist,** O'Reilly Media, 2024.
2. J.V. Guttag, **Introduction to Computation and Programming Using Python: With Application to Understanding Data**, MIT Press, 2016.
3. Robert Sedgewick , Kevin Wayne, Robert Dondero, **Introduction to Programming in Python: An Interdisciplinary Approach,** Addison-Wesley Professional, 2015
4. Tony Gaddis, **Starting Out with Python**, Pearson, 2021.

Additional References

(i) Brown, Martin C. *Python: The Complete Reference*, 2nd edition, McGraw Hill Education, 2018.

Suggested Practical List

1. WAP to find the roots of a quadratic equation
2. WAP to accept a number 'n' and
    a. Check if 'n' is prime
    b. Generate all prime numbers till 'n'
    c. Generate first 'n' prime numbers

    This program may be done using functions
3. WAP to create a pyramid of the character '*' and a reverse pyramid

```
    *
   ***
  *****
 *******
*********
```

```
*********
 *******
  *****
   ***
    *
```

4. WAP that accepts a character and performs the following:
    a. print whether the character is a letter or numeric digit or a special character
    b. if the character is a letter, print whether the letter is uppercase or lowercase
    c. if the character is a numeric digit, prints its name in text (e.g., if input is 9, output is NINE)
5. WAP to perform the following operations on a string
    a. Find the frequency of a character in a string.
    b. Replace a character by another character in a string.
    c. Remove the first occurrence of a character from a string.
    d. Remove all occurrences of a character from a string. WAP to swap the first n characters of two strings.
6. Write a function that accepts two strings and returns the indices of all the occurrences of the second string in the first string as a list. If the second string is not present in the first string then it should return -1.
7. WAP to create a list of the cubes of only the even integers appearing in the input list (may have elements of other types also) using the following:
    a. 'for' loop
    b. list comprehension

8. WAP to read a file and

   a. Print the total number of characters, words and lines in the file.
   b. Calculate the frequency of each character in the file. Use a variable of dictionary type to maintain the count.
   c. Print the words in reverse order.
   d. Copy even lines of the file to a file named 'File1' and odd lines to another file named 'File2'.

9. Define a class *Employee* that stores information about employees in the company. The class should contain the following:

   (1) data members- count (to keep a record of all the objects being created for this class) and for every employee: an employee number, Name, Dept, Basic, DA and HRA.

   (2) function members:

      a. __init__ method to initialize and/or update the members. Add statements to ensure that the program is terminated if any of Basic, DA and HRA is set to a negative value.
      b. function salary, that returns salary as the sum of Basic, DA and HRA.
      c. __del__ function to decrease the number of objects created for the class
      d. __str __ function to display the details of an employee along with the salary of an employee in a proper format.

10. Write a program to define a class "2DPoint" with coordinates x and y as attributes. Create relevant methods and print the objects. Also define a method distance to calculate the distance between any two point objects.

11. Write a function that prints a dictionary where the keys are numbers between 1 and 5 and the values are cubes of the keys.

12. Inherit the above class to create a "3Dpoint" with additional attribute z. Override the method defined in "2DPoint" class , to calculate distance between two points of the " 3DPoint" class.

13. Consider a tuple t1=(1, 2, 5, 7, 9, 2, 4, 6, 8, 10). WAP to perform following operations:

   a. Print half the values of the tuple in one line and the other half in the next line.

b. Print another tuple whose values are even numbers in the given tuple.

c. Concatenate a tuple t2=(11,13,15) with t1.

d. Return maximum and minimum value from this tuple

14. WAP to accept a name from a user. Raise and handle appropriate exception(s) if the text entered by the user contains digits and/or special characters.