

Kadi Sarva Vishwavidyalaya, Gandhinagar

MCA Semester I

MCA-15: Programming using Open Source

Rationale: This course introduces core programming basics—including data types, control structures, algorithm development, and program design with functions—via the Python programming language. It also covers the fundamental principles of Object-Oriented Programming and Graphical User Interface.

Prerequisite:

- Knowledge of Programming languages such as C, C++, JAVA and .NET
- Analysis of coding complexities.

Learning Outcomes:

Students will learn to program in interactive mode with lot of time saving in development resulting in lesser developmental cost by

- Applying decision and repetition structures in program design.
- Implementing methods and functions to improve readability of programs
- Demonstrating the use of Python lists and dictionaries
- Describing and applying object-oriented programming methodology with GUI Programming

Teaching and Evaluation Scheme: The objective of evaluation is to evaluate the students throughout the semester for better performance. Students are evaluated on the basis of continuous evaluation system both in theory and practical classes based on various parameters like term work, class participation, practical and theory assignments, presentation, class test, Regular Attendance, etc.

Sub Total Credit	Teaching scheme		Examination scheme				
	(per week)		MID	CEC	External		Total Marks
	Th	Pr	Th	Th	Th.	Pr.	
5	3	4	25	25	50	50	150

Course content:

Unit I: Basics of Python

[20%]

Introduction to Python, Components of the Python language System, Using Python in interactive mode with basic operations and built-in functions, Reading input from the console, Identifiers, Variables, Assignment Statements and Expression, Named Constants, Numeric Data types and Operators. Common Python functions, Objects & Methods, String Functions, Formatting Numbers and Strings.

Unit II: Programming Basics

[20%] Control

Flow- if, if-elif-else, for, while, break, continue, pass. Functions: Defining and Calling Functions, Functions with/without return values, Positional and keyword arguments, Passing arguments by reference values. Keyword Arguments, Default Arguments, Variable-length arguments, Scope of the Variables in a Function - Global and Local Variables. Dates and Times Functions - Creating Dates and Times, Dates

Arithmetic, Modules and Packages.

Unit III: Programming Advance Features

[20%]

Lists : List Basics, Copying Lists, Passing Lists to functions, Returning a list from a function, Searching list, Sorting list, Processing Two-Dimensional lists, Passing Two-Dimensional lists to functions. **Tuples**: Tuples, Accessing tuples, Operations, Functions. **Sets** : Accessing values in sets, Operations, Functions. **Dictionaries**: Accessing values in dictionaries, Operations, Functions.

Unit IV: Object Oriented Programming, Exceptions & Database

[20%]

OOP in Python: Classes, 'self variable', Methods, Constructor, Inheritance, Overriding Methods, Data hiding, **Handling Exception**: try except block, Raising Exceptions, User Defined Exceptions.

Database: Database Connection, CRUD Operations.

Unit V: GUI Development

[20%]

GUI Development: Getting started with Tkinter, Processing Events, The Widget classes, canvas, The Geometry Managers, Displaying Images, Menus, Popup Menus, Mouse, Key Events and Bindings, Standard Dialog Boxes, MVC Architecture.

Text Books:

1. Introduction to Programming Using Python, Y. Daniel Liang, Pearson Publications
2. Practical Programming - an Introduction to Computer Science Using Python by Jennifer Campbell, Paul Gries, Jason Montojo, Greg Wilson.

Reference Books:

1. Fundamentals of Programming Python by Richard L. Halterman.
2. The Quick Python Book, Vernon L. Ceder, Manning Publications
3. Python and Tkinter Programming, John E. Grayson, Manning Publications

Chapters as per Units

Unit I: Chapters 2, 3

Unit II: Chapters 4, 5, 6

Unit III: Chapters 10, 11, 14

Unit IV: Chapters 12, 13.6 – 13.9, (From Book 2 - Pg 339 – 360)

Unit V: Chapter 9

Practical List

1. Write a program that asks the user about textbook prices and reports how overpriced the textbooks are. (You may wish to read this number from the user with the input command. You can round numbers with the round command.)
2. Create a program that outputs the total cost of a lunch order. Users should be prompted to input the number of hamburgers, fries, and drinks they want and the program should print the total cost of the order. The hamburgers cost 2.00, fries cost 1.50, and drinks cost 1.00. Be creative and professional in prompting the user for the information and in displaying the output.
3. Write a Python program that prompts the user for the cost of two items to be purchased. Then prompt the user for their payment. If they enter an amount that is less than the total cost of the two items, print a message that tells them how much they still owe. Otherwise, print a message that thanks them for their payment and tells them how much change they will receive. Thoroughly test your code for all possible input.

4. Create a program which will allow the user to enter the state of two switches (either 1 (on) or 0 (off)). The program should work out if both switches are on and then output the message 'the light is on'. Otherwise, the program should output the message 'the light is off'.
5. Create a program that will keep track of items for a shopping list. The program should keep asking for new items until nothing is entered (no input followed by enter/return key). The program should then display the full shopping list.
6. Write a program that will store the schedule for a given day for a particular TV station. The program should ask you for the name of the station and the day of the week before asking you for the name of each show and the start and stop times. Once the schedule is complete it should be displayed as a table.
7. Write a program that maps a list of words into a list of integers representing the lengths of the corresponding words.
8. Create a function with signature: `integer(number)` The number parameter is either a number or a string that can be converted to a number. The function should return the number as type `int`, rounding it if the number passed in is a float. If the conversion fails, catch the `ValueError` exception, and return 0. Make sure it works for both strings and literal numbers, such as 4.5, 32, "23", and "-15.1", and that it correctly returns zero for invalid numbers like "tonsils".
9. Write a program to check whether an element „y“ and „a“ belongs to the tuple `My_tuple=(„p“,„y“,„t“,„h“,„o“,„n“)` and after Printing the result, delete the tuple.
10. Write a python program named Weather that is passed a dictionary of daily temperatures, and returns the average temperature over the Weekend for the weekly temperatures given.
11. Write a program to catch a Divide by zero exception. Add a finally block too.
12. Generalize a case study on the getting the students mark statements and analysis with exception handling.
13. If you ever find yourself buying a house, you'll want to know what your monthly payment for the loan is going to be. Write a complete program that asks for information about a loan and prints the monthly payment. The formula for computing monthly mortgage payments involves the loan amount, the total number of months involved (a value we call n) and the monthly interest rate (a value we call c). The payment formula is given by the following equation:

$$payment = loan \frac{c(1+c)^n}{(1+c)^n - 1}$$

14. Write a set of classes corresponding to the geometric shapes: cube, rectangle, cone and cylinder. Each class should have a constructor that allows for the creation of different sized objects (e.g., a cube is specified by the length of its side) and a method that return the area of the shape.
15. Write a procedure that evaluates polynomials. It should take two arguments. The first is a number x . The second is a list of coefficients ordered from highest to lowest:

$$a_n, a_{n-1}, \dots, a_2, a_1, a_0$$

Your procedure should return the value of the polynomial evaluated at x :

$$a_n x^n + a_{n-1} x^{n-1} + \dots + a_2 x^2 + a_1 x + a_0$$