

Fundamentals of Coding and Cloud

COURSE OBJECTIVE

- Understand the fundamentals of Python programming, including syntax, data types, and control flow, to build a solid foundation in the language.
- Gain proficiency in object-oriented programming concepts such as classes, objects, inheritance, and polymorphism to write modular and reusable code.
- Develop problem-solving skills by working with data structures like arrays, lists, dictionaries, and implementing searching and sorting algorithms.
- Learn how to handle file input/output operations, perform string manipulation, and apply regular expressions for efficient data processing.
- Define cloud computing and explain its basic principles.
- Understand the benefits and advantages of adopting cloud computing.
- Identify and differentiate between the cloud service & Deployment models.
- Enhance quantitative aptitude through topics like mathematics,

COURSE OUTCOMES

- Gain a strong command over Python programming language, including its syntax, data types, and control flow, to develop robust and efficient code.
- Understand and apply object-oriented programming principles, such as encapsulation, inheritance, and polymorphism, to design and implement modular and reusable code.
- Develop skills in manipulating data structures like arrays, lists, and dictionaries, and implement common algorithms such as searching and sorting for efficient data processing.
- Enhance problem-solving abilities by applying quantitative aptitude, and algorithmic thinking to tackle complex coding challenges and develop optimal solutions.
- Prepare for coding interviews by gaining exposure to a variety of problem-solving scenarios, practicing coding techniques, and developing effective communication skills to articulate programming concepts and solutions.

COURSE CURRICULUM

UNIT - I	Fundamentals of Python	05 hours
Introduction to Python: history, features, and Applications-Setting up the development environment for Python-Basic syntax Variables and data types: numbers-strings-lists-tuples-dictionaries Operators: arithmetic-assignment-comparison-logical-bitwise operators Control flow: conditional statements (if, Elif, else) and loops (for and while) Functions: defining functions, function parameters, return values, and recursion File handling: reading from and writing to files Basic input and output operations Challenges: -		

1- Implement a recursive function to calculate the factorial of a given number.

2- Write a program that determines whether a year entered by the user is a leap year or not using if-elif-else statements.

UNIT - II	Object-Oriented Programming (OOP)	05 hours
------------------	--	-----------------

Introduction to OOP: principles- concepts-benefits.

Classes and objects: defining classes-creating objects-accessing attributes and methods.

Encapsulation: data hiding and access modifiers (public, private, protected).

Inheritance: creating derived classes- method overriding-super keywords.

Polymorphism: method overloading-method overriding-abstract classes.

Advanced OOP concepts: interfaces, composition, and static methods.

Challenges: -

1- Implement a class called BankAccount that represents a bank account. The class should have private attributes for account number, account holder name, and account balance. Include methods to deposit money, withdraw money, and display the account balance. Ensure that the account balance cannot be accessed directly from outside the class. Write a program to create an instance of the BankAccount class and test the deposit and withdrawal functionality.

2- Implement a class called Player that represents a cricket player. The Player class should have a method called play() which prints "The player is playing cricket. Derive two classes, Batsman and Bowler, from the Player class. Override the play() method in each derived class to print "The batsman is batting" and "The bowler is bowling", respectively. Write a program to create objects of both the Batsman and Bowler classes and call the play() method for each object.

UNIT - III	Data Structures and Manipulation	10 Hours
-------------------	---	-----------------

Arrays and lists: indexing-slicing-common list operations.

String manipulation: string methods-formatting-regular expressions.

Dictionaries and sets: manipulating dictionary -set objects.

Stack and queue data structures.

Searching algorithms: linear search-binary search.

Sorting algorithms: bubble sort- insertion sort- selection sort.

Challenges: -

1-Write a function called `linear_search_product` that takes the list of products and a target product name as input. The function should perform a linear search to find the target product in the list and return a list of indices of all occurrences of the product if found, or an empty list if the product is not found.

2- Implement a function called `sort_students` that takes a list of student objects as input and sorts the list based on their CGPA (Cumulative Grade Point Average) in descending order. Each student object has the following attributes: `name` (string), `roll_number` (string), and `cgpa` (float). Test the function with different input lists of students.

UNIT - IV	Fundamentals of Cloud Computing	05 Hours
Cloud Computing – Need of Cloud Computing- Cloud Service Models-Cloud Deployment Models-Essential Characteristics of Cloud Computing- Cloud Computing Providers-Fundamental Course		
UNIT - V	Quantitative Aptitude	05 Hours
Basic Mathematics - LCM & HCF - Divisibility - Numbers, decimal fractions and power –Averages - Ratio & Proportion – Algebra Applied Mathematics - Profit and Loss - Simple and Compound Interest - Time, Speed and Distance – Inverse - Time and Work - Allegation & Mixtures – Percentage - Area, shape, and perimeter		

Total: 30 Periods

REFERENCE	
1	Python Programming: - https://developers.google.com/edu/python
2	Python OOPs: - https://realpython.com/python3-object-oriented-programming/
3	Competitive Programming: - https://www.codewars.com/kata/python
4	Aptitude: - https://www.indiabix.com/aptitude/questions-and-answers/

SOFTWARE REQUIREMENT
<ul style="list-style-type: none"> Google Colab / Anaconda Navigator

HARDWARE REQUIREMENT
<ul style="list-style-type: none"> Desktop / Laptop with Core i3 or higher Processor, Windows 10 OS / 4 GB of RAM Minimum 5 MBPS download speed of internet connection