SEMESTER-VI

St. Xavier's College (Autonomous), Ahmedabad

Syllabus of Semester – VI of the following departments under Faculty of Computer Science based on Undergraduate Curriculum Framework to be implemented from the Academic Year 2025-26.

DEPARTMENT OF COMPUTER SCIENCE

Major Course – 1: Python Programming

CREDIT DISTRIBUTION, ELIGIBILITY AND PRE-REQUISITES OF THE COURSE

Course Title & Code	Credit Distribution of The Course			Eligibility Criteria	Pre-requisite(s) of the Course (if any)
	Lecture	Tutorial	Practical / Practice		
Python Programming BCAMC661C	2	0	2	Nil	Programming Basics

Learning Objectives:
This course introduces students to Python programming, covering fundamental concepts such as syntax, control structures, and data structures. Students will learn to write efficient programs, handle files, and implement Object-Oriented Programming (OOP).

Learning Outcomes:
Upon successful completion, students will be able to implement:

- Write and execute Python programs using variables, operators, and expressions.
- Implement decision-making statements (if-else) and loops (for, while).
- Define functions, pass arguments, and handle exceptions effectively.
- Work with strings, lists, tuples, dictionaries, and sets.
- Perform file handling operations like reading, writing, and appending files.
- Implement mathematical and logical operations such as factorial, prime numbers, Fibonacci series, HCF, and LCM.
- Use list comprehension and set operations for efficient data processing.
- Apply OOP principles including classes, inheritance, and operator overloading.
- Read and process files for statistical analysis, computing mean, median, and mode.
- Handle errors and exceptions in Python programs.

Unit	Particulars	Hours
I	▪ **Introduction to Python** ▪ History and evolution of Python ▪ Features and applications of Python ▪ Installing Python and setting up the environment ▪ **Python Basics** • Variables, constants, and data types • Operators: Arithmetic, Relational, Logical, Bitwise • Taking input and printing output • Type conversion and typecasting ▪ **Control Structures** • Conditional statements: if, elif, else • Looping structures: for, while • Loop control statements: break, continue, pass	15 30
II	▪ **Functions and Exception Handling** • Defining and calling functions • Function arguments and return values • Recursion vs. Iteration • Exception handling (try, except, finally) ▪ Basic Python Programs ▪ Demonstration of Decision Statements ▪ Implementation of Looping Structures ▪ Demonstration on Functions ▪ Implementation on Exception Handling	 15
III	▪ **Strings and String Manipulation** • String operations (split(), join(), replace(), find()) • String formatting (format(), f-strings) • Reversing and modifying strings ▪ **Data Structures** ▪ Lists, tuples, dictionaries, sets ▪ List comprehension and set comprehension ▪ Sorting and filtering data	 30
IV	▪ **File Handling** • Reading, writing, and appending files • Counting lines and words in a file • Copying and modifying files • Checking file status (closed or not) ▪ **Object-Oriented Programming** • Classes and objects • Constructors (__init__) • Operator overloading (e.g., Complex number operations) • Inheritance and polymorphism	

Textbook:

1. Python Programing for the absolute beginner 3rd edition – Michael Dawson – Cengage Learning.
2. Python 3 for Absolute Beginners – Tim Hall and J P Stacey – Apress.

Reference Books:

1. Introducing Python - by Bill Lubanovic - O'Reilly
2. Python Cookbook – Alex Martelli, Anna Martelli Ravenscroft and David Ascher - O'Reilly
3. Think Python – Allen B Downey - O'Reillyython Data Science Cookbook – Gopi Subramanian – PACKT Publishing.
4. Python for Data Analysis – Wes McKinney - O'Reilly
5. Python Machine Learning – Sebastian Rachka - PACKT Publishing

Sample Programs

1. Write a function to print the factorial of a number{no recursion}
2. Calculate simple interest. accept P,T,R from user.
3. To check the given number is armstrong.
4. To print all armstrong numbers between two range.
5. To check a given number is prime. use function
6. print all prime numbers between two range
7. To print nth prime number
8. To print all composite numbers within n. use function
9. Find the area of circle.
10. To print all fibonacci number till use{ 0,1,1,2,3,.}use function
11. To print nth fibonacci number
12. sum of the square of 1st n natural numbers using function
13. sum of the cube of 1st n natural numbers
14. to check palindrome.use 2 functions
15. Reverse words in a long string using function
16. To remove a character from a string at specifc position and print the remaining string.
17. input a string and split the string into words and print only the odd length of words in that string
18. Find the length of a string without using built in function len() 3.. swap commas and dots in a string
19. find words whose length greater than n in a given sentence.
20. count the number of vowels in a list.
21. prints all the numbers from 12 to 31 except 17 and 19.using 'continue statement
22. calculate the sum and average of n integer numbers given by user. Input 0 to finish
23. to check the triangle is equilateral, isosceles, scalene.(nested if)
24. print all numbers which are divisible by 2 and 7 in the range of within n
25. convert temperatures for celsius entry to fahrenheit and vice versa{c/5=f-32/9}
26. Input a year and check for leapyear.
27. count the number of even and odd numbers from a given tuple
28. Print mutiplication table of 11,12,13 (1*11 to 10*11)
29. print number of alphabets,special characters and numbers in the string input numbers between two numbers, print the odd numbers separated by comma{237=3,7}
30. Program to calculate the HCF and LCM of two numbers.
31. Program to make a simple calculator
32. Write a Python program to read an entire text file.
33. Write a Python program to read first n lines of a file.

34. Write a Python program to append text to a file and display the text.
35. Write a Python program to read a file line by line and store it into a list.
36. Write a Python program to count the number of lines in a text file.
37. Write a Python program to count the frequency of words in a file.
38. Write a Python program to get the file size of a plain file.
39. Write a Python program to write a list to a file.
40. Write a Python program to copy the contents of a file to another file.

41. Write a Python program to combine each line from first file with the corresponding line in second file.
42. Write a Python program to read a random line from a file.
43. Write a Python program to assess if a file is closed or not.
44. Write a Python program to remove newline characters from a file.
45. Write a function using exceptions that asks the user to enter a set of integers. Calculate mean. The user indicates that she is done by typing “Done”.
46. Write two versions of a function that calculates the harmonic mean of a list of integers The first version uses an if-statement to make sure that the list element is not zero. List elements with zero value are ignored. The second version uses exceptions to do the same thing.

    n/(1/a1+1/a2+1/a3….)

47. Write a function that takes a list as a single argument. The function returns the number of items that cannot be converted into a floating point number. Use exceptions.
48. Perform the above question with the help of isinstance() method
49. Program to add, subtract, equality of two complex numbers
50. Program to multiply and divide of two complex numbers.
51. Use list comprehension to generate a. list of all cube between 1 and 100
52. Use list comprehension to generate a list of all multiples of 7 between 1 and 100
53. Use set comprehension and intersection to find all numbers divisible by 15 and divisible y 21 between 1 and 1000
54. Create a list of the first 20 powers of 2 [1,2,4,8,16……]
55. Create a list of all numbers between 10000 and 20000that have last digit 3 and are divisible by 13
56. Work with all widgets of Tkinter
57. Write a program to perform descriptive statistics of a set of numbers

SEMESTER-VI

St. Xavier's College (Autonomous), Ahmedabad

Syllabus of Semester – VI of the following departments under Faculty of Computer Science based on Undergraduate Curriculum Framework to be implemented from the Academic Year 2025-26.

DEPARTMENT OF COMPUTER SCIENCE

Major Course – 2: Frameworks and Technologies for Web Development

CREDIT DISTRIBUTION, ELIGIBILITY AND PRE-REQUISITES OF THE COURSE

Course Title & Code	Credit Distribution of The Course			Eligibility Criteria	Pre-Requisite(s) of the Course (if any)
	Lecture	Tutorial	Practical / Practice		
Frameworks and Technologies for Web Development BCAMC662C	4	0	2	Students with basic coding knowledge	Nil

Course Outcome:

At the end of the course, the student will be able to:

- Introduce modern JavaScript-based front-end frameworks.
- Develop competency in component-based UI development.
- Encourage analytical thinking through state and data management.
- Promote experiential and project-based learning.
- Enable students to design industry-relevant web applications.

Unit 1: Introduction & Environment Setup

- Evolution of front-end development
- Introduction to React JS
- Features and advantages of React
- Importance of Node in React
- Single Page Applications (SPA) vs Multi Page Applications
- Architecture of SPA
- Differentiate between Traditional websites and Single Page Application

- Role of React in modern web development
- Installation of Node.js (LTS) and VS Code (Offline setup)
- Creating first React application using pre-downloaded template
- Understanding React project folder structure
- Introduction to JSX syntax and expressions

UNIT 2 Components, Props & State

- Function components and Class component architecture
- Component reusability and modular design
- Props: passing data between components
- Default props and props validation (basic concept)
- State management fundamentals
- useState() hook – syntax and usage
- Handling user events (click, change, submit)
- Conditional rendering techniques
- Component interaction and data flow

UNIT 3 Lists, Forms & Hooks

- Rendering lists using map()
- Keys and performance optimization (introductory)
- Controlled and uncontrolled components
- Form handling in React
- Basic form validation techniques
- Introduction to React hooks
- useEffect() hook for lifecycle management
- Dependency array and cleanup function
- Browser Local Storage and Session Storage
- Offline data persistence techniques

UNIT 4 Routing, Styling & Mini Project

- Introduction to client-side routing
- React Router: components and configuration (offline setup)
- Navigation between pages
- Route parameters and basic redirection
- Styling approaches in React
    - External CSS
    - Bootstrap (Offline integration)
    - Tailwind CSS (Offline – optional)
- Mini Project planning and design
- Implementation and debugging

- Project presentation and demonstration

PRACTICAL PROGRAM LIST

UNIT 1: Introduction to React and JSX

1. Write a simple program to display “Hello World”.
2. Write a simple program to create student card also add styling components.
3. Write a React component that shows a product’s name, price, and description inside a styled card layout.
4. Create a Reusable Greeting Card using react (use button click event to display msg on alert box also use basic movement over icon and image using css animation)
5. Create User Profile Card and display the information such as profile picture, name, age, country. (get value using prompt)
6. Write a program in react for temperature converter to display Fahrenheit and Celsius values.
7. Design the frame to display the quote of the day. (add 7 different quotes for each day using array and display them, highlight the particular quote as per the current day.)
8. Write a program to display favorite books list. Display a list of 5 books in a styled unordered list.
9. Write a program to display your age. “You are an adult” if age > 18 Otherwise show “You are a minor”.
10. Program to demonstrate Components in ReactJS environment.

UNIT 2: Components, Props & State

1. Write a React program to demonstrate state management using the useState() hook.
2. Write a React program to implement a counter application using state.
3. Write a React program to increment and decrement a value using buttons and state.
4. Write a React program to change displayed text dynamically using state.
5. Write a React program to toggle content visibility using conditional rendering.
6. Write a React program to toggle login and logout status using state.
7. Write a React program to demonstrate event handling using button click.
8. Write a React program to change background color dynamically on button click.
9. Write a React program to accept user input and display it dynamically using state.
10. Write a React program to enable or disable a button based on state condition.
11. Write a React program to display different messages based on user age using conditional rendering.
12. Write a React program to demonstrate multiple states in a single component.

UNIT 3: Lists, Forms & Hooks

1. Write a React program to display a list of subjects using the map() function.
2. Write a React program to dynamically add items to a list.
3. Write a React program to remove items from a list dynamically.
4. Write a React program to create a controlled input field using state.
5. Write a React program to create a student registration form.
6. Write a React program to implement basic form validation for empty fields.
7. Write a React program to validate email and password in a login form.

8. Write a React program to demonstrate the use of useEffect() hook on component load.
9. Write a React program to update document title using useEffect().
10. Write a React program to store user input data in Local Storage.
11. Write a React program to retrieve and display data from Local Storage.
12. Write a React program to create a simple notes application using list and Local Storage.

UNIT 4: Routing, Styling & Mini Project

1. Write a React program to create a multi-page application using React Router.
2. Write a React program to create Home, About, and Contact pages using React Router.
3. Write a React program to implement navigation using the Link component.
4. Write a React program to implement route-based component rendering.
5. Write a React program to apply external CSS styling to React components.
6. Write a React program to design a card layout using CSS in React.
7. Write a React program to create a simple calculator application using React.
8. Write a React program to create a To-Do list application using React.
9. Write a React program to add, update, and delete tasks in a To-Do list.
10. Write a React program to store To-Do list data in Local Storage.
11. Write a React program to integrate routing, styling, and storage in one application.
12. Develop and present a complete offline React-based mini project.

Suggested Mini Projects (Experiential Learning – NEP 2020)

- To-Do List Application
- Student Information Management System
- Notes Application
- Calculator Application
- Simple Offline Portfolio Website

Reference Book:

1. **The Road to React: Your Journey to Master Plain Yet Pragmatic React.js** – Robin Wieruch
2. **React: The Comprehensive Guide** – Sebastian Springer

St. Xavier's College (Autonomous), Ahmedabad

Syllabus of Semester – VI of the following departments under Faculty of Computer Science based on Undergraduate Curriculum Framework to be implemented from the Academic Year 2024-25.

DEPARTMENT OF COMPUTER SCIENCE

BCA (Hons.) Category – IV

Major Course - 3 Software Development Project-PART II

CREDIT DISTRIBUTION, ELIGIBILITY AND PRE-REQUISITES OF THE COURSE

Course Title & Code	Credit Distribution of The Course			Eligibility Criteria	Pre-requisite(s) of the Course (if any)
	Lecture	Tutorial	Practical / Practice		
Software Development Project-PART II BCAMC663L	4	0	0	10 + 2 from a recognized board in any stream	Nil

Course Objective:

1) Synthesize skills and knowledge gained into an innovative technology solution.
2) Develop systems development skills.
3) Gain experience in exploring/using software development and documentation tools.

Course Outcome:

At the end of the syllabus, the student will be able to:

To gain the knowledge of the new technology and explore the existing technology in depth.

To achieve the desired output of the proposed system by dividing the modules in group and collaborate their modules among team members. COBC6506.03:

To design the interface and implement the code.

To test their code in order to make the system error free and meet the needs of the conducted analysis.

To generate reports and provide the organization with neccesary information. To gain an experience of working in live environment.

Mode of study: One day off to work on the project in a week

Course Contents: 1. Developing System Design 2. Writing code for the project 3. Doing testing of the code

Deliverables by the students: At the end of the semester, the student should be able to successfully develop the project and prepare the documentation (hard copy) as well as presentation of the project details.

• **Documentation**: o A single hard bound documentation of SDP Part-II should also consist of the documentation prepared in SDP Part-I. o Although the students might have submitted the documentation of SDP Part-I, it should not be considered for evaluation.

A hard copy of the documentation should consist of the additional following details:

- Cover Page
- Company Certificate
- College Certificate
- Acknowledgement
- Index (with page nos.)
- Screen layouts
- Report layouts
- Sample coding (optional)
- Future Enhancements (optional)
- Conclusion
- Bibliography

Presentation: • Presentations can be prepared through slides using Open Source / Power Point / Flash or any other multimedia tool, covering the work shown in the documentation. • During viva exams, students will be expected to satisfactorily answer the questions pertaining to the tools used, the process, the reports /forms created and the results achieved.