दिल्ली विश्वविद्यालय
UNIVERSITY OF DELHI

Computer Science Courses
for
Physical Science/ Mathematical Science

(Effective from Academic Year 2019-20)

Revised Syllabus as approved by

Academic Council

Date: No:

Executive Council

Date: No:

Applicable for students registered with Regular Colleges, Non Collegiate Women's Education Board and School of Open Learning

List of Contents

Page No.

Preamble

The objective of any programme at Higher Education Institute is to prepare their students for the society at large. The University of Delhi envisions all its programmes in the best interest of their students and in this endeavour it offers a new vision to all its Under-Graduate courses. It imbibes a Learning Outcome-based Curriculum Framework (LOCF) for all its Under Graduate programmes.

The LOCF approach is envisioned to provide a focused, outcome-based syllabus at the undergraduate level with an agenda to structure the teaching-learning experiences in a more student-centric manner. The LOCF approach has been adopted to strengthen students' experiences as they engage themselves in the programme of their choice. The Under-Graduate Programmes will prepare the students for both, academia and employability.
Each programme vividly elaborates its nature and promises the outcomes that are to be accomplished by studying the courses. The programmes also state the attributes that it offers to inculcate at the graduation level. The graduate attributes encompass values related to well-being, emotional stability, critical thinking, social justice and also skills for employability. In short, each programme prepares students for sustainability and life-long learning.

The new curriculum of Computer Science Courses for BSc Physical Science/Mathematical Science is designed to develop computational thinking, analytical, and problem solving skills. It covers core computer science topics and offers electives so that students can apply these skills while studying subjects like Maths, Physics, Chemistry etc. The programme also lays down the foundation for higher studies in the field of Computer Science/Applications.

The University of Delhi hopes the LOCF approach of the Computer Science Courses for BSc Physical Science/Mathematical Science will help students in making an informed decision regarding the goals that they wish to pursue in further education and life, at large.

1. Introduction to Computer Science Courses for BSc Physical Science/Mathematical Science

BSc Physical Science/Mathematical Science programme with Computer Science is offered in several prestigious colleges of the University of Delhi. This programme aims to introduce the discipline of Computer Science to the students who wish to either pursue higher studies in computer science or wish to use computational skill in study of physical and mathematical sciences. The courses are designed to promote logical thinking, analytical skills, develop programming skills and application of knowledge of computing to solve problems in other disciplines.

The curriculum for computer science courses in BSc Physical Science/Mathematical Science programme was developed by the Department of Computer Science by following the due diligent process. Close consultations were held with the college teachers involved in teaching of the courses. Inputs were collected from the college teachers in general body meetings and the courses were decided. Subsequently, committees were formed for designing syllabi for each course. Draft syllabus of each course was thoroughly discussed and debated among the peers in DU colleges. After multiple iterations, the final syllabus of the programme was approved in the Committee of Courses for Undergraduate Studies. The draft was then published in public domain for review by all stakeholders, and additionally sent for peer review outside University of Delhi. The review comments were thoroughly discussed in the Committee of Courses for Undergraduate Studies, and the appropriate changes were incorporated. Finally, the syllabus was placed in the Faculty of Mathematical Sciences and approved.

2. Aims of Computer Science Courses for BSc Physical Science/Mathematical Science

The objective of BSc Physical Science/Mathematical Science Programme with Computer Science is to introduce the discipline to students who want to pursue either higher studies in science or branch off to other disciplines for higher studies, or those who want to be educators. Specifically, the program aims the following achievements for students.

1. To attain understanding of computer systems, their applications and fundamentals.
2. To develop ability to apply knowledge of computing to solve computational problems.
3. To analyze a problem, and identify the computing requirements appropriate to its solution.

4. To design, implement, and evaluate a computer-based system, process or program to meet the desired needs.
5. To communicate effectively with a range of audiences

3.1 Course Structure for Under -Graduate (Non-Hons.) Programme

Course	Credits	
	Theory+ Practical	**Theory+ Tutorial**
I. Core Course	12X4=48	12X5=60
(12 Papers) Core Course Practical / Tutorial* Two papers – English Two papers – MIL Four papers – Discipline 1. Four papers – Discipline 2.		
Core Course Practical / Tutorial (12 Practicals)	12X2=24	12X1=12
II. Elective Course (6 Papers) Two papers- Discipline 1 specific Two papers- Discipline 2 specific Two papers- Inter disciplinary Two papers from each discipline of choice and two papers of interdisciplinary nature.	6X4=24	6X5=30
Elective Course Practical / Tutorials (6 Practical/ Tutorials) Two papers- Discipline 1 specific Two papers- Discipline 2 specific Two papers- Generic (Inter disciplinary) Two papers from each discipline of choice including papers of interdisciplinary nature.	6X2=12	6X1=6
• Optional Dissertation or project work in place of one elective paper (6 credits) in 6th Semester		
III. Ability Enhancement Courses		
Ability Enhancement Compulsory Courses (AECC) (2 Papers of 4 credits each) Environmental Science English Communication/MIL	2X4=8	2X4=8
Skill Enhancement Courses (SEC) (4 Papers of 4 credits each)	4X4=16	4X4=16
Total credits	**132**	**132**

3.2 Semester Wise Placement of Computer Science Courses

Sem-ester	Discipline Specific Core Course (DSC) (4)	Ability Enhancement Compulsory Course (AEC) (2)	Skill Enhancement Course (SEC) (4)	Elective Discipline Specific (DSE) (2)
I	BSCS01	AECC 1		
II	BSCS02	AECC 2		
III	BSCS03		BSCS07A BSCS07B	
IV	BSCS04		BSCS08A BSCS08B	
V			BSCS09A BSCS09B	BSCS05A BSCS05B
VI			BSCS10A BSCS10B	BSCS06A BSCS06B BSCS06C

Discipline Specific Core Papers (DSC) for Computer Science (Credit: 06 each)

1. BSCS01: Problem Solving using Computers
2. BSCS02: Database Management Systems
3. BSCS03: Operating System
4. BSCS04: Computer System Architecture

Discipline Specific Elective Papers: (Credit: 06 each) (DSE-1, DSE -2)

Choose One from each group.

Options for DSE1:

1. BSCS05A: Data Structures
2. BSCS05B: Digital Image Processing

Options for DSE2:

1. BSCS06A: Computer Networks
2. BSCS06B: Analysis of Algorithms
3. BSCS06C: Project Work / Dissertation

Skill Enhancement Elective Courses
(SEC1, SEC2, SEC3, SEC4)

Choose one from each group.

Options for SEC1:

1. BSCS07A: Data Analysis using Python Programming
2. BSCS07B: Introduction to R Programming

Options for SEC2:

1. BSCS08A : Programming in C++
2. BSCS08B: Programming in Java

Options for SEC3:

1.BSCS09A: Advanced Programming in Java
2. BSCS09B: Web Design using HTML5

Options for SEC4:

1. BSCS10A: Android Programming
2. BSCS10B: PHP Programming

Note:

1. There will be one batch of 10-15 students for practical classes. The size of tutorial group for papers without practical is recommended to be 8-10 students.
2. Each practical will carry 50 marks including 25 marks for continuous evaluation and 5 marks for the oral viva.
3. Colleges are advised and encouraged to conduct the practical using Free and Open Source Software (FOSS)
4. At least two questions have to be compulsorily attempted in the final practical examination.
5. Softcopy of all the practical must be maintained by each student for each practical paper.
6. Discipline specific core and elective courses (DSC and DSE) are to be taught as 4 Hrs theory and 4 Hrs practical per week. In case the course has tutorials, it is to be taught as 5 Hrs theory and 1 Hr tutorial per week
7. Skill enhancement courses (SEC) are to be taught as 2 Hrs theory and 4 Hrs practical per week.
8. Practical given for the courses are only indicative, and by no means exhaustive. Instructor may add more complex problems in laboratory depending on the ability of the students.

4. Detailed Syllabi of Computer Science Courses for BSc Physical Science/Mathematical Science

---

Problem Solving using Computers
(BSCS01)
Core Course - (CC) Credit:6

Course Objective

This course is designed as the first course in programming to develop problem solving skills. The course focuses on modularity, reusability, code documentation, and debugging skills. It also introduces the concept of object-oriented programming.

Course Learning Outcomes

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

1. describe the components of a computer and the notion of an algorithm.
2. apply suitable programming constructs and data structures to solve a problem.
3. develop, document, and debug modular python programs.
4. use classes and objects in application programs.
5. use files for I/O operations.

Unit 1

Computer Fundamentals and Problem Solving: Basic Computer Organization: CPU, memory, I/O Units. Problem solving using computer, notion of an algorithm.

Unit 2

Introduction to Python Programming: Python interpreter, using python as calculator, python shell, indentation, identifiers and keywords, literals, strings, arithmetic, relational and logical operators.

Unit 3

Creating Python Programs: Input and output statements, defining functions, control statements default arguments, errors and exceptions.

Unit 4

Inbuilt Data Structures: strings, lists, sets, tuples, nested lists, built-in functions, dictionary and associated operation.

Unit 5

Object Oriented Programming: Introduction to Classes, Objects and Methods, Standard Libraries, File handling through libraries.

Unit 6

Sorting and Searching: Iterative and Recursive methods for searching and sorting

Practical

1. Execution of expressions involving arithmetic, relational, logical, and bitwise operators

in the shell window of Python IDLE.

2. Write a Python function to produce the following outputs.

(a)
```
*
**
***
*****
******
*******
```

(b)
```
$ $ $ $ $
$ $
$ $
$ $
$ $ $ $ $
```

3. Write a Python program to illustrate the functions of math module specified by the instructor in laboratory.
4. Write a Python program to produce a table of sins, cosines and tangents. Make a variable x in range from 0 to 10 in steps of 0.2. For each value of x, print the value of sin(x), cos(x) and tan(x).
5. Write a menu driven program to calculate the area of Square, rectangle, circle and triangle. Use suitable assertions.
6. Write a Python function that takes a number as an input from the user and computes its factorial. Then find the sum of the n terms of the following series:

$$1 + 1/1! + 1/2! + 1/3! + \ldots + 1/n$$

7. Write a Python function to return n th terms of Fibonacci sequence
8. Write a function that takes a number as an input and finds its reverse and computes the sum of its digits.
9. Write a function that takes two numbers as input parameters and returns their least common multiple.
10. Write a function that takes a number as an input and determine whether it is prime or not.
11. Write a Python function that takes a string as an input from the user and displays its reverse.

More exercises using sets, lists and dictionary, as announced by instructor in the laboratory.

References

1. Downey, A.B. (2008). *Think Python–How to think like a Computer Scientist*. Needham, Massachusetts : Green Tea Press.
2. Urban, M. & Murach, J. (2018). *Python Programming*. Shroff.

Additional Resources

1. Guttag, J. V. (2013). *Introduction to computation and programming using Python.* MIT Press.
2. Liang, Y. D. (2013). *Introduction to Programming using Python.* Pearson.
3. Taneja, S. & Kumar, N. (2018). *Python Programming - A modular Approach.* Pearson.

Teaching Learning Process

- Talk and chalk method
- Computer based presentations by teachers.
- Group Discussions
- Assignments
- Offline and online Quiz
- Presentations by group of students for enhanced learning.

Tentative weekly teaching plan is as follows:

Week 1 - 2	**Unit 1-Computer Fundamentals and Problem Solving**: Basic Computer Organization: CPU, memory, I/O Units. Problem solving using computer, notion of an algorithm.
Week 3 - 4	**Unit 2-Introduction to Python Programming:** Python interpreter, using python as calculator, python shell, indentation. identifiers and keywords, literals, strings, arithmetic, relational and logical operators.
Week 5 - 7	**Unit 3- Creating Python Programs:** input and output statements, defining functions, control statements, default arguments, errors and exceptions.
Week 7 - 9	**Unit 4 - Inbuilt Data Structures:** Strings, lists, sets, tuples, nested lists, built-in functions, dictionary and associated operation.
Week 10 - 11	**Unit 5 - Object Oriented Programming:** Introduction to Classes, Objects and Methods, Standard Libraries, File handling through libraries.

Week 12 - 15	**Unit 6 - Sorting and Searching:** Iterative and Recursive methods for searching and sorting

Assessment Methods

- Unit-wise assignments, presentations, viva, quiz,as announced by the instructor in the class
- End semester exam
- Internal assessment

Keywords

Problem Solving, Control structure, Functions, Strings, Lists, Object Oriented Programming

---

Database Management Systems
(BSCS02)
Core Course –(CC) Credit:6

Course Objective

The course introduces the students to the fundamentals of database management systems and methods to store and retrieve data. The course would give students hands-on practice of structured query language in a relational database management system.

Course Learning Outcomes

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

1. use database management system to manage data.
2. create entity relationship diagrams for modeling real-life situations and design the database schema.
3. use the concept of functional dependencies to remove data anomalies and arrive at normalized database design.
4. write queries using relational algebra and SQL.

Unit 1

Introduction to DBMS: Introduction to Database Management Systems, characteristics of database approach, data models, DBMS architecture and data independence.

Unit 2

Conceptual Modelling using ERD and EERD: Entity Relationship (ER) and Enhanced ER (EER) modeling, entity types, relationships, relationship constraints, and object modeling.

Unit 3

Relational Data Model and Relational Algebra: Relational data model concepts, relational constraints, queries in relational algebra.

Unit 4

Introduction to SQL: Data definition and data manipulation queries in SQL.

Unit 5

Database Design: Mapping of ER and EER diagrams to relational database, functional dependencies, Normalization and normal forms up to third normal form.

Practical

Note: MyAccess/MySQL may be used.
The following concepts must be introduced to the students:
Practicals based on DDL Commands

Create table, alter table, drop table

Practicals based on DML Commands

a) Select, update, delete, insert statements
b) Condition specification using Boolean and comparison operators (and, or, not,=,<>,>,<,>=,<=)
c) Arithmetic operators and aggregate functions(Count, sum, avg, Min, Max)
d) Multiple table queries (join on different and same tables)
e) Nested select statements
f) Set manipulation using (any, in, contains, all, not in, not contains, exists, not exists, union, intersect, minus, etc.)
g) Categorization using group by.....having
h) Arranging using order by

Queries for the given schema

a) Create tables with relevant foreign key constraints
b) Populate the tables with data
c) Display all the details of all records in table.
d) Display some fields of the records in table.
e) Conditionally display some fields of the records in table.
f) Retrieve all distinct values of an attribute
g) Retrieve records from table for numeric attribute in range
h) Retrieve records that satisfy complex condition.

Queries for typical (Employee, Project, Department) database

a) Retrieve the names of all employees who do not have supervisors
b) Retrieve SSN and department name for all employees

c) Retrieve the name and address of all employees who work for the 'Research' department

More exercises as announced by instructor in the laboratory.

References

1. Elmasri, R., Shamkan, & Navathe, B. (2017). *Fundamentals of Database Systems (7th Edition).*Pearson Education.

Additional Resources

1. Ramakrishanan, R., & Gehrke, J. (2002). *Database Management Systems (3rd Edition).* McGraw-Hill.

2. Silberschatz, A., Korth, H.F., & Sudarshan, S. (2011). *Database System Concepts (6th Edition).* McGraw Hill.

Teaching Learning Process

- Talk and chalk method
- Computer based presentations by teachers.
- Group Discussions
- Assignments
- Offline and online Quiz
- Presentations by group of students for enhanced learning.

Tentative weekly teaching plan is as follows:

Week 1 - 3	**Unit 1 - Introduction to DBMS:** Introduction to Database Management Systems, characteristics of database approach, data models, DBMS architecture and data independence.
Week 4 - 6	**Unit 2- Conceptual Modelling using ERD and EERD:** Entity Relationship (ER) and Enhanced ER (EER) modeling, entity types, relationships, relationship constraints, and object modeling
Week 7 - 9	**Unit 3 - Relational Data Model and Relational Algebra:** Relational data model concepts, relational constraints, queries in

	relational algebra
Week 10 - 11	**Unit 4 - Introduction to SQL:** Data manipulation and data definition queries in SQL
Week 12 -15	**Unit 5- Database Design:** Mapping of ER and EER diagrams to relational database, functional dependencies, Normalization and normal forms up to third normal form

Assessment Methods

- Unit-wise assignments, presentations, viva, quiz,as announced by the instructor in the class
- End semester exam
- Internal assessment

Keywords

DBMS architecture, Data Independence, Entity modeling, Relational Data Model, SQL, Normalization

---

Operating Systems
(BSCS03)
Core Course - (CC) Credit:6

Course Objective

This course introduces Operating System concepts and its importance in computer system. It focuses on the basic facilities provided in modern operating systems.

Course Learning Outcomes

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

1. understand the rationale behind the current design and implementation decisions in modern Operating Systems by considering the historic evolution.

2. identify modules of the operating systems and learn about important functions performed by operating system as resource manager.

3. use the OS in a more efficient manner.

Unit 1

Introduction: Operating systems, System software, Operating system resources, Operating System as resource manager.

Unit 2

Types of Operating Systems and Organization: Multiprogramming, batch, time sharing operating systems, personal computers & workstations. Basic OS functions, mechanisms of requesting operating system services – system calls and system programs.

Unit 3

Processor Management: Distinction between program and process, process address space, process states, process scheduling algorithms, process schedulers.

Unit 4

Memory Management: Mapping logical address space to physical address space, fixed partition, variable partition, paging, segmentation, virtual memory.

Unit 5

File and Input/Output Device Management: Classifications of I/O devices, I/O handling, file systems services, directory structure, disk storage.

Unit 6

Shell scripting: Shell variables, parameter passing, conditional statements, iterative statements, writing and executing shell scripts, utility programs (cut, paste, grep, echo, pipe, filter, etc.)

Practical

1. Usage of following commands: ls, pwd, tty, cat, who, who am I, rm, mkdir, rmdir, touch, cd.
2. Usage of following commands: cal, cat(append), cat(concatenate), mv, cp, man, date.
3. Usage of following commands: chmod, grep, tput (clear, highlight), bc.
4. Write a shell script to check if the number entered at the command line is prime or not.
5. Write a shell script to modify "cal" command to display calendars of the specified months.
6. Write a shell script to modify "cal" command to display calendars of the specified range

of months.

7. Write a shell script to accept a login name. If not a valid login name display message – "Entered login name is invalid".
8. Write a shell script to display date in the mm/dd/yy format.
9. Write a shell script to display on the screen sorted output of "who" command along with the total number of users .
10. Write a shell script to display the multiplication table of any number.
11. Write a shell script to compare two files and if found equal asks the user to delete the duplicate file.
12. Write a shell script to find the sum of digits of a given number.
13. Write a shell script to merge the contents of three files, sort the contents and then display them page by page.
14. Write a shell script to find the LCD(least common divisor) of two numbers.
15. Write a shell script to perform the tasks of basic calculator.
16. Write a shell script to find the power of a given number.
17. Write a shell script to find the factorial of a given number.
18. Write a shell script to check whether the number is Armstrong or not.
19. Write a shell script to check whether the file have all the permissions or not.
20. Program to show the pyramid of special character "*".

References

1. Silberschatz, A., Galvin, P.B., & Gagne, G. (2008). *Operating Systems Concepts (8th Edition).* John Wiley Publications.

Additional Resources

1. Milenkovic, M. (1992). *Operating Systems - Concepts and design.* Tata McGraw Hill.
2. Nutt, G. (1997). *Operating Systems: A Modern Perspective (2nd Edition).* Pearson Education.
3. Stallings, W. (2008). *Operating Systems, Internals & Design Principles (5th Edition).* Prentice Hall of India.
4. Tanenbaum, A.S. (2007). *Modern Operating Systems (3rd Edition).* New Delhi: Pearson Education.

Teaching Learning Process

- Talk and chalk method
- Computer based presentations by teachers.
- Group Discussions
- Assignments
- Offline and online Quiz
- Presentations by group of students for enhanced learning.

Tentative weekly teaching plan is as follows:

Week 1 - 2	**Unit 1 - Introduction :** System software, resource abstraction, Operating System

	strategies, Operating System as resource manager
Week 3 - 5	**Unit 2 - Types of operating Systems and organizations** Multiprogramming, batch, time sharing, personal computers & workstations, Basic OS functions, process modes, methods of requesting system services – system calls and system programs.
Week 6 - 7	**Unit 3- Processor Management:** Distinction between program and process, process address space, process states, process scheduling algorithms, process schedulers.
Week 8 - 10	**Unit 4 - Memory Management:** Mapping logical address space to physical address space, fixed partition, variable partition, paging, segmentation, virtual memory.
Week 11 - 13	**Unit 5 - File and Input/Output Device Management**: Classifications of I/O devices, I/O handling, file systems services, directory structure, disk storage.
Week 14 - 15	**Unit 6 - Shell scripting:** Shell variables, parameter passing, conditional statements, iterative statements, writing and executing shell scripts, utility programs (cut, paste, grep, echo, pipe, filter, etc.)

Assessment Methods

- Unit-wise assignments, presentations, viva, quiz,as announced by the instructor in the class
- End semester exam
- Internal assessment

Keywords

Memory Management, Process Management, File Management, Virtual memory

---

Computer System Architecture
(BSCS04)
Core Course - (CC) Credit:6

Course Objective

The course will introduce students to the fundamental concepts of digital computer organization, design and architecture. It aims to develop a basic understanding of the design of a computer system.

Course Learning Outcomes

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

1. design combinational circuits using basic building blocks. Simplify these circuits using Boolean Algebra and Karnaugh maps.

2. differentiate between combinational circuits and sequential circuits

3. represent data in binary form, convert numeric data between different number systems and perform arithmetic operations in binary.

4. determine various stages of instruction cycle, various instruction formats and instruction set.

5. describe interrupts and their handling.

6. explain how CPU communicates with memory and I/O devices.

Unit 1

Digital Logic Gates, Flipflops and their characteristic table, Logic circuit simplification using Boolean Algebra and Karnaugh Map, Don't Care conditions.
Combinational Circuits, Sequential Circuits.

Unit 2

Digital Components: Decoders, Encoders, Multiplexers, Binary Adder, Binary Adder-Subtractor, Binary Incrementer, Registers and Memory Units

Unit 3

Data Representation: Binary representation of both numeric and alphanumeric data, representation of numeric data in different number systems (Binary, Octal, Decimal and Hexadecimal), conversion from one number system to another, complements, representation of decimal numbers, representation of signed and unsigned numbers, addition and subtraction of signed and unsigned numbers and overflow detection.

Unit 4

Operations and Control: Arithmetic and logical micro-operations, instruction format, micro programmed control vs hardwired control, instruction set completeness, Timing and control,

instruction cycle, memory reference instructions and their implementation using arithmetic, logical, program control, transfer and input output micro operations, interrupt cycle.

Unit 5

Instructions: Instruction format illustration using single accumulator organization, general register organization and stack organization, zero-address instructions, one-address instructions, two-address instructions and three-address instructions, Addressing Modes

Unit 6

Peripheral Devices: I/O interface, I/O vs. Memory Bus, Isolated I/O, Memory Mapped I/O, Direct Memory Access

Practical

1. Write a program to convert a number in Radix 'R' to radix 10 and vice versa. Test the same by
   a. Converting an unsigned number from binary, octal, hex to decimal.
   b. Converting an unsigned number from decimal to binary, octal, hex.
2. Write a program that will prompt for the input of two integer values. Then using the bitwise shift operators show the result of
   a. Left shifting the first number by the second
   b. Right shifting the first number by the second
   c. Exclusive OR of the first number by the second bitwise
   d. OR of the first number by the second bitwise
   e. AND of the first number by the second bitwise
3. Write a program that will prompt for the input of a binary value and print:
   a. One's complement
   b. Two's complement
4. Write a program to print the values of a 5 bit binary up-down counter. User should be able to specify the up or down nature of the counter.
5. Write a program to implement the following binary operations:
   a. Addition
   b. Subtraction using 2's complement.

6. Program related to concepts taught in theory

References

1. Mano, M. (1992). *Computer System Architecture (3rd Edition).* Pearson Education.

Additional Resources

1.Mano, M. (2013). *Digital Design*. New Jersey: Pearson Education Asia.
2..Null, L., & Lobur, J. (2014). *The essentials of computer organization and architecture.* Jones & Bartlett Publishers.
3.Stallings, W. (2003). *Computer organization and architecture: designing for performance.*

India : Pearson Education.
.

Teaching Learning Process

- Talk and chalk method
- Computer based presentations by teachers.
- Group Discussions
- Assignments
- Offline and online Quiz
- Presentations by group of students for enhanced learning.

Tentative weekly teaching plan is as follows:

Week 1 - 4	**Unit 1 - Introduction**: Digital Logic Gates, Flipflops and their characterstic table, Logic circuit simplification using Boolean Algebra and Karnaugh Map, Don't Care conditions. Combinational Circuits, Sequential Circuits.
Week 5 - 6	**Unit 2 - Digital Components:** Decoders, Encoders, Multiplexers, Binary Adder, Binary Adder-Subtractor, Binary Incrementer, Registers and Memory Units
Week 7 - 9	**Unit 3 - Data Representation:** Binary representation of both numeric and alphanumeric data, representation of numeric data in different number systems (Binary, Octal, Decimal and Hexadecimal), conversion from one number system to another, complements, representation of decimal numbers, representation of signed and unsigned numbers, addition and subtraction of signed and unsigned numbers and overflow detection.
Week 10 - 11	**Unit 4 - Operations and Control:** Arithmetic and logical micro-operations, instruction format, micro programmed control vs hardwired control, instruction set completeness, Timing and control, instruction cycle, memory reference instructions and their implementation using arithmetic, logical, program control, transfer and input output micro operations, interrupt cycle.
Week 12 - 13	**Unit 5 - Instructions:** Instruction format illustration using single accumulator organization, general register organization and stack organization, zero-address instructions, one-address instructions, two-address

	instructions and three-address instructions, Addressing Modes
Week 14 -15	**Unit 6 - Peripheral Devices:** I/O interface, I/O vs. Memory Bus, Isolated I/O, Memory Mapped I/O, Direct Memory Access

Assessment Methods

- Unit-wise assignments, presentations, viva, quiz,as announced by the instructor in the class
- End semester exam
- Internal assessment

Keywords

Combinational circuits, Data representation, Interrupts, I/O interface

---

Data Structures
(BSCS05A)
Discipline Specific Elective - (DSE) Credit:6

Course Objective

The course introduces the students to the fundamentals of data structures. Students will learn about arrays, stacks, queues, linked lists, recursion and trees.

Course Learning Outcomes

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

1. demonstrate a thorough understanding of the behaviour of basic data structures.
2. implement data structures efficiently in programming language C++.
3. demonstrate an understanding of recursion by applying recursive techniques to solve problems.

Unit 1

Arrays and Sorting: Single and multi-dimensional arrays, sparse matrices, different sorting methods including bubble, selection, insertion, merge, quick sort, linear and binary searching.

Unit 2

Stacks: Implementing stack using array, prefix, infix and postfix expressions, application of

stacks for conversion of infix to prefix and postfix expressions, evaluation of postfix expressions.

Unit 3

Queue: Implementing simple queue, circular queues and priority queues using array.

Unit 4

Linked Lists: Single, double and circular lists, implementing stack and queue using linked lists.

Unit 5
Recursion: Recursive solutions to simple problems and their implementation, advantages and limitations of recursion.

Unit 6

Trees: Introduction to tree as a data structure, binary trees, binary search tree- creation and traversal techniques.

Practical

a. Implement sorting algorithms using arrays.
b. Implement searching algorithms using arrays and lists.
c. Implement stack data structure and its operations.
d. Convert Prefix expression to Infix and Postfix expressions, and evaluate.
e. Implementing queue using array, circular queues, priority queues.
f. Implementing stack and queue using arrays and linked lists.
g. Implementing recursive solutions to simple problems.
h. Creating and traversing Binary Trees and Binary Search Tree.

References

1. Drozdek, A. (2012). *Data Structures and algorithm in C++ (3rd Edition).* Cengage Learning.

Additional Resources

1. Sahni, S. (2011). *Data Structures, Algorithms and applications in C++ (2nd Edition).* Universities Press.
2. Tenenbaum, A. M., Augenstein, M. J., & Langsam, Y. (2009). *Data Structures Using C and C++ (2nd edition).* PHI.

Teaching Learning Process

- Talk and chalk method
- Computer based presentations by teachers.

- Group Discussions
- Assignments
- Offline and online Quiz
- Presentations by group of students for enhanced learning.

Tentative weekly teaching plan is as follows:

Week 1 – 4	**Unit 1 - Arrays and Sorting:** Single and multi-dimensional arrays, sparse matrices, different sorting methods including bubble, selection, insertion, merge, quick sort, linear and binary searching
Week 5 – 6	**Unit 2 - Stacks:** Implementing stack using array, prefix, infix and postfix expressions, application of stacks for conversion of infix to prefix and postfix expressions, evaluation of postfix expressions
Week 7 – 8	**Unit 3 - Queue:** Implementing simple queue, circular queues and priority queues using array
Week 9 - 11	**Unit 4 - Linked Lists:** Single, double and circular lists, implementing stack and queue using linked lists
Week 12 – 13	**Unit 5 - Recursion:** Recursive solutions to simple problems and their implementation, advantages and limitations of recursion
Week 14 – 15	**Unit 6 - Trees:** Introduction to tree as a data structure, binary trees, binary search tree- creation and traversal techniques

Assessment Methods

- Unit-wise assignments, presentations, viva, quiz,as announced by the instructor in the class
- End semester exam
- Internal assessment

Keywords

Arrays, linked lists, stacks, queues, tree, recursion

Digital Image Processing
(BSCS05B)
Discipline Specific Elective - (DSE) Credit:6

Course Objective

The course introduces the basic concepts and methodologies of digital image processing. The topics covered include image enhancement, spatial and frequency domain, image filtering, morphological image processing and image segmentation.

Course Learning Outcomes

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

1. describe general terminology of Digital Image Processing and the roles of image processing systems in a variety of applications.
2. describe the basic issues and the scope (or principal applications) of image processing.
3. explain representation and manipulation of digital images, image acquisition, reading, writing, enhancement, displaying and segmentation and image Fourier transform.
4. examine various types of images, intensity transformations and spatial filtering.

Unit 1

Introduction: Fundamental steps in digital image processing (DIP), applications of DIP, components of image processing system, image types (binary, grayscale, color, truecolor, cartoon).

Unit 2

Digital Image Fundamentals : Elements of visual perception (Human eye, electromagnetic spectrum), Image acquisition, sampling and quantization, basic relationships between pixels.

Unit 3

Image Enhancement in spatial domain : Basic gray level transformations, histogram processing, smoothing and sharpening filters.

Unit 4

Image enhancement in frequency domain : DCT transform, enhancement filters in frequency domain, JPEG Image Compression.

Unit 5

Morphological Image processing: Erosion, dilation, opening , closing, Hit-or-miss transform, some basic morphological algorithms including boundary extraction, convex hull, thinning and thickening.

Unit 6

Image Segmentation : Detection of discontinuities, edge linking and basic thresholding.

Practical

1) Implementing Image acquisition, sampling and quantization
2) Programs for image enhancement in spatial domain
3) Programs for image enhancement in frequency domain
4) Implementing Morphological Image processing

References

1. Gonzalez, R. C., & Woods, R. E. (2008).*Digital Image Processing. 3rd Edition*. Pearson Education.

Additional Resources

1. Castleman, K. R. (1996). *Digital Image Processing*. Pearson Education
2. Gonzalez, R. C., Woods, R. E., & Eddins, S. (2004). *Digital Image Processing using MATLAB*. Pearson Education, Inc.
3. Jain, A. K. (1994). Fundamentals of Digital Image Processing. Prentice Hall of India
4. Schalkoff, R.J. (1989). *Digital Image Processing and Computer Vision.* New York: John Wiley and Sons.

Teaching Learning Process

- Talk and chalk method
- Computer based presentations by teachers.
- Group Discussions
- Assignments
- Offline and online Quiz
- Presentations by group of students for enhanced learning.

Tentative weekly teaching plan is as follows:

Week 1 - 2	**Unit 1 - Introduction**: Fundamental steps in digital image processing (DIP), applications of DIP, components of image processing system, image types (binary, grayscale, color, truecolor, cartoon).
Week 3 - 5	**Unit 2 - Digital Image Fundamentals** : Elements of visual perception (Human eye, electromagnetic spectrum), Image acquisition, sampling and quantization, basic relationships between pixels.
Week 6 - 9	**UNIT III - Image Enhancement in spatial domain** : Basic gray level transformations, histogram processing, smoothing and sharpening filters.
Week 9 - 10	**Unit 4-Image enhancement in frequency domain**: DCT transform, enhancement filters in frequency domain, JPEG Image Compression.
Week 11 - 13	**Unit 5** **Morphological Image processing**: Erosion, dilation, opening , closing, Hit-or-miss transform, some basic morphological algorithms including boundary extraction, convex hull, thinning and thickening.
Week 14 -15	**Unit 6 - Image Segmentation** : Detection of discontinuities, edge linking and basic thresholding.

Assessment Methods

- Unit-wise assignments, presentations, viva, quiz,as announced by the instructor in the class
- End semester exam
- Internal assessment

Keywords

Visual perception, Image Segmentation, Fourier transform, DCT transform

---

Computer Networks
(BSCS06A)
Discipline Specific Elective - (DSE) Credit:06

Course Objective

This course provides an overview of the concepts of data communication and computer networks. Network topologies and their characteristics, different type of networks, transmission media along with their limitations and use, different protocols used in application layer are covered.

Course Learning Outcomes

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

1. understand the basics of data communication.
2. differentiate between various types of computer networks and their topologies.
3. understand the difference between the OSI and TCP/IP protocol suit.
4. explain merits and demerits of different types of communication media.
5. distinguish between different types of network devices and their functions.
6. use IP addressing and understand the need of various application layer protocols.

Unit 1

Introduction: Introduction to data communications and networking, use of Computer Networks, classification of networks, OSI model, function of the layers, TCP/IP Protocol suite.

Unit 2

Network Topologies: Bus, star, ring, mesh, tree, hybrid topologies with their features, advantages and disadvantages of each type. Transmission Modes: simplex, half duplex and full duplex.

Unit 3

Transmission Media: Guided Media (Wired) (Twisted pair, Coaxial Cable, Fiber Optics. Unguided Media (Radio Waves, Infrared, Micro-wave, Satellite).

Unit 4

Data Communication and Switching Techniques: Framing, flow control, error control, circuit switching, message switching, packet switching, routing.

Unit 5

Switching Devices: Repeaters, hubs, switches, bridges, routers, gateways. Multiplexing: (FDM,

WDM, TDM)

Unit 6

Internet: Internet Service Providers (ISP), internet addressing system: IP address with their classification and notation, application layer protocols: (DNS, URL, WWW, FTP, SMTP, HTTP, TELNET), web pages, introduction to HTML.

Practical

1. Simulate Cyclic Redundancy Check (CRC) error detection algorithm for noisy channel.
2. Simulate and implement stop and wait protocol for noisy channel.
3. Simulate and implement go back n sliding window protocol.
4. Simulate and implement selective repeat sliding window protocol.

More exercises as announced by instructor in the laboratory.

References

1. Comer, D. E. (2015). *Computer Networks and Internet (6th edition).* Pearson Publication.

Additional Resources

1. Forouzan, B. A. (2017). *Data Communications and Networking (5th edition).* McGraw Hill.
2. Tannenbaum, A. S., & Wetherall, D. J. (2011). *Computer Networks (5th edition).* Pearson Publication.

Teaching Learning Process

- Talk and chalk method
- Computer based presentations by teachers.
- Group Discussions
- Assignments
- Offline and online Quiz
- Presentations by group of students for enhanced learning.

Tentative weekly teaching plan is as follows:

Week 1 - 2	**Unit 1 - Introduction:** Introduction to data communications and networking, use of Computer Networks, classification of networks, OSI model, function of the layers, TCP/IP Protocol suite.

Week 3 - 4	**Unit 2 -** N**etwork Topologies:** Bus, star, ring, mesh, tree, hybrid topologies with their features, advantages and disadvantages of each type. Transmission Modes: simplex, half duplex and full duplex.
Week 5 - 6	**Unit 3 - Transmission Media:**Guided Media (Wired) (Twisted pair, Coaxial Cable, Fiber Optics). Unguided Media (Radio Waves, Infrared, Micro-wave, Satellite).
Week 7 - 8	**Unit 4 - Data Communication and Switching Techniques:** Framing, flow control, error control, circuit switching, message switching, packet switching, routing.
Week 9 - 11	Unit 5 - Switching Devices: Repeaters, hubs, switches, bridges, routers, gateways. Multiplexing: (FDM, WDM, TDM)
Week 12 - 15	Unit 6 - Internet: Internet Service Providers (ISP), internet addressing system: IP address with their classification and notation, application layer protocols: (DNS, URL, WWW, FTP, SMTP, HTTP, TELNET), web pages, introduction to HTML.

Assessment Methods

- Unit-wise assignments, presentations, viva, quiz,as announced by the instructor in the class
- End semester exam
- Internal assessment

Keywords

Data communication, Computer Networks, Switching, Internet Protocol, IP address

---

Analysis of Algorithms
(BSCS06B)
Discipline Specific Elective - (DSE) Credit:6

Course Objective

The course provides techniques for design and analysis of algorithms. Topics include sorting, searching, heaps, divide and conquer, greedy and dynamic programming, and graph algorithms.

Course Learning Outcomes

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

1. understand the idea of algorithm analysis.
2. understand characteristics of searching and sorting algorithms and compare efficiency of different solutions for an application at hand.

model simple problems as **graphs** and solve those using graph algorithms.

Unit 1

Mathematical Preliminaries: Growth of functions, Asymptotic notation, standard notations and common functions. Recurrences: The substitution method for solving recurrences, recursion-tree method for solving recurrences, the master method for solving recurrences.

Unit 2

Divide and Conquer Algorithms: General method, binary search, merge sort, quicksort algorithms, space and running time analysis of the algorithms.

Unit 3

Sorting Algorithms: Insertion sort, Heapsort, Sorting in linear time.

Unit 4

Graph Algorithms: Representations of graphs, Breadth-first search, Depth-first search, topological sort.

Unit 5

Greedy Algorithms and dynamic programming: Minimum spanning tree, shortest path in a graph, 0/1 knapsack problem and fractional knapsack problem.

Practical

1. Implement Insertion Sort and report the number of comparisons.
2. Implement Merge Sort and report the number of comparisons.
3. Implement Heap Sort and report the number of comparisons .
4. Implement Randomized Quick sort and report the number of comparisons.
5. Implement Radix Sort.

6. Implement Searching Techniques.
7. Implementation of Recursive function.
8. Array and Linked list implementation of Stack and Queue.
9. Implementation of Single, Double and circular Linked List.
10. Creation and traversal of Binary Search Tree.

References

1. Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2015). *Introduction to Algorithms (3rd Edition)*. PHI.

Additional Resources

1. Basse, S., & Gleder, A.V. (1999). *Computer Algorithm – Introduction to Design and Analysis (3rd edition).* Pearson.
2. Kleinberg, J., & Tardos, E. (2013). *Algorithm Design.* Pearson.

Teaching Learning Process

- Talk and chalk method
- Computer based presentations by teachers.
- Group Discussions
- Assignments
- Offline and online Quiz
- Presentations by group of students for enhanced learning.

Tentative weekly teaching plan is as follows:

Week 1 - 4	**Unit 1- Mathematical Preliminaries:** Growth of functions, Asymptotic notation, standard notations and common functions. Recurrences: The substitution method for solving recurrences, recursion-tree method for solving recurrences, the master method for solving recurrences.
Week 5 - 7	**Unit 2 - Divide and Conquer Algorithms**: General method, binary search, merge sort, Quick sort algorithms, space and running time analysis of the algorithms
Week 8 - 10	**Unit 3 - Sorting Algorithms:** Insertion sort, Heapsort, sorting in linear time
Week 11 - 12	**Unit 4 - Graph Algorithms:** Representations of graphs, Breadth-first search, Depth-first search, topological sort

Week 13 - 15	**Unit 5 - Greedy Algorithms and dynamic programming** Minimum spanning tree, shortest path in a graph, 0/1 knapsack problem and fractional knapsack problem

Assessment Methods

- Unit-wise assignments, presentations, viva, quiz,as announced by the instructor in the class
- End semester exam
- Internal assessment

Keywords

Asymptotic Notation, Divide and Conquer Algorithms, Sorting Algorithms, Graph Algorithms, Greedy Algorithms, Dynamic Programming

---

Project Work / Dissertation
(BSCS06C)
Discipline Specific Elective - (DSE) Credit:6

Course Objective

The students will undergo one semester of project work based on the concepts studied in a subject of their choice. The objective is to train the students for the industry by exposing them to prototype development of real life software.

Course Learning Outcomes

On successful completion of this course, a student will be able to:

1. develop a project plan based on informal description of the project.
2. implement the project as a team.
3. write a report on the project work carried out by the team and defend the work done by the team collectively.
4. present the work done by the team to the evaluation committee.

Unit 1

The students will work on any project based on the concepts studied in core/elective/ skill based elective courses. Specifically, the project could be a research study, or a software development project.

Unit 2

Project Group Organization/Plan

- Students will initially prepare a synopsis (500 words) and submit it to their respective department.
- For a given project, the group size could be a maximum of four (04) students.
- Each group will be assigned a teacher as a supervisor who will be responsible for their lab classes.
- A maximum of four (04) projects would be assigned to one teacher.

Unit 3

Project Evaluation

- 100 marks for end semester examination comprising Viva/presentation (50 marks) and project report evaluation (50 marks): to be awarded jointly by the examiner and supervisor / mentor.
- 50 marks for continuous evaluation (to be awarded by the supervisor/mentor). Work carried out in each lab session will be assessed out of five marks (zero for being absent). Finally, the marks obtained will be scaled out of a maximum of 50 marks. For example, if 30 lab sessions are held in a semester, and a student has obtained an aggregate of 110 marks, then he/she will be assigned round (110/(30*5)) i.e. 37 marks.
- The students will submit only the soft copies of the report.
- The reports may be retained by the examiners.

Practical

Practical/discussion sessions based on the area of the project.

Teaching Learning Process

- Group Discussions
- Presentations by group of students for enhanced learning.

Tentative weekly teaching plan is as follows:

Week 1 - 2	**Unit 1** The students will work on any project based on the concepts studied in core/elective/ skill based elective courses.

	Specifically, the project could be a research study, or a software development project
Week 3 - 12	**Unit 2 -Project Group Organization/Plan**
Week 13-15	**Unit 3 - Project Evaluation** Continue the project and start report writing.

Assessment Methods

- Unit-wise assignments, presentations, viva, quiz,as announced by the instructor in the class
- Internal assessment
- End semester exam

Keywords

Software Development, Project planning.

Data Analysis using Python Programming (BSCS07A) Skill-Enhancement Elective Course - (SEC) Credit:4

Course Objective

The course enables students to analyse data using python. They will learn how to prepare data for analysis and create meaningful data visualisations. They will learn to use Pandas, Numpy and Scipy libraries to work with different data sets.

Course Learning Outcomes

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

1. develop a python script for data analysis and execute it.
2. install, load and deploy the required packages.
3. clean and prepare the data for accurate analysis.
4. analyse the data stored in files in different formats.
5. experiment with data visualization methods.

Unit 1

Introduction to Pandas, NumPy, SciPy: Introduction to Pandas DataFrames, Numpy multi-dimentional arrays, and SciPy libraries to work with different datasets.

Unit 2

Import and Export of Data: Installing, loading and using packages for importing and exporting data in Python.

Unit 3

Data Preprocessing and Transformation: Handling of missing data, Data cleaning and transformation.

Unit 4

Data Exploration: Exploring data using statistical methods: mean, median, mode, quantiles. Building contingency table. Basics of grouping data and Correlation.

Unit 5

Data Visualization: Scatter Plot, line graph, histogram, boxplot, line plots regression, word clouds, exporting plots as images.

Practical

Use data set of your choice from Open Data Portal (https://data.gov.in/) for the following exercises.

1. Make visual representations of data using libraray Matplotlib and apply basic principles of data graphics to create rich analytic graphs for available datasets.
2. For the given data,
    i. Generate box whisker plot
    ii. Identify outliers, if any
    iii. Display 5 point summary of data distribution
3. Create a CSV file having employee data records. Each employee record has three features viz. age, home city and salary. Import employee file and :
    i. Draw scatter plot for age vs salary
    ii. Plot histogram for features age and salary
    iii. Plot Pie chart for the qualitative attribute city
4. Import iris data using sklearn library to:
    i. Compute mean, mode, median, standard deviation, confidence interval and standard error for each feature
    ii. Compute correlation between length and width of sepal feature
    iii. Find covariance between length of sepal and petal
    iv. Build contingency table for class feature

5. Download datasets Hepatitis and automobile from UCI repository
    i. Find the number of records which are noise free
    ii. Clean data after removing noise
    iii. Normalize quantitative features in range of [0,1]
6. Practical based on data analysis functions from Pandas, NumPy, SciPy
7. Practical based on visualizing data as Scatter Plot, line graph, histogram, boxplot, line plots regression, word clouds, and exporting plots as images

Projects and Case Studies to be done as decided by the department in the beginning of the semester.

References

1. Mckinney, W. (2017). *Python for Data Analysis. Second edition,* O'reilly (SPD).

Additional Resources

1. Grus, J. (2016). *Data Science from scratch. First edition,* O'reilly (SPD).
2. VanderPlas, J. (2016). *Python Data Science Handbook: Essential Tools for Working with Data.* Second edition, O'reilly (SPD).

Web Resources

https://docs.python.org/3/tutorial/
https://www.python.org/

Teaching Learning Process

- Talk and chalk method
- Computer based presentations by teachers.
- Group Discussions
- Unit-wise assignments
- Offline and online Quiz
- Presentations by group of students for enhanced learning.

Tentative weekly teaching plan is as follows:

Week 1 – 4	**Unit 1 - Introduction to Pandas, NumPy, SciPy:** Introduction to Pandas DataFrames, Numpy multi-dimentional arrays, and SciPy libraries to work with different datasets
Week 5 – 7	**Unit 2 - Import and Export of Data:** Installing, loading and using packages for importing and exporting data in Python
Week 8 – 10	**Unit 3- Data Preprocessing and Transformation:**

	Handling of missing data, Data cleaning and transformation
Week 11 – 13	**Unit 4 - Data Exploration:** Exploring data using statistical methods: mean, median, mode, quantiles. Building contingency table. Basics of grouping data and Correlation.
Week 14 - 15	**Unit 5 - Data Visualization:** Scatter Plot, line graph, histogram, boxplot, line plots regression, word clouds, exporting plots as images.

Assessment Methods

- Unit-wise assignments, presentations, viva, quiz,as announced by the instructor in the class
- End semester exam
- Internal assessment

Keywords

Data import and export, Cleaning, Transformation, Data Exploration, Data Visualization

---

Introduction to R Programming
(BSCS07B)
Skill-Enhancement Elective Course - (SEC) Credit:4

Course Objective

This course introduces statistical programming language R for data analysis. The objective is to expose the students to the strengths and capabilities of R for data analysis. It also encourages students to use open source softwares.

Course Learning Outcomes

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

1. develop an R script for data analysis and execute it.
2. install, load and deploy the required packages.
3. analyse the data stored in files in different formats.
4. identify suitable data visualization and exploration methods to answer a business question.
5. interpret the results of analysis.

Unit 1

Introduction to Programming Structures: R interpreter, Introduction to major R data structures like vectors, matrices, arrays, list and data frames, Flow control and loops, looping over list and array, user-defined functions.

Unit 2

File Handling: Installing, loading and using packages for reading data from files

Unit 3

Data Preprocessing and Transformation: Handling of missing data, Data cleaning and transformation

Unit 4

Data Exploration: Exploring data using statistical methods: mean, median, mode, quantiles. Building contingency table, correlation, co-variance.

Unit 5

Plotting Data: Data visualization using Scatter plot, Line graph, Histogram, Boxplot.

Practical

1. Find measures of central tendencies for the given data.
2. Draw the box plot for the given data and analyse skewness.
3. For the given data, plot the PDF.
4. Make visual representations of data using libraray Matplotlib and apply basic principles of data graphics to create rich analytic graphs for available datasets.
5. Randomly generate 30 numbers in the range of 1 to 40 and do the following

    a) Generate box plot
    b) Identify outliers, if any
    c) Display 5 point summary of data distribution

Use data set of your choice from Open Data Portal (https://data.gov.in/) for the following exercises.

6. Read the all rows from the CSV file, with header. Write an R script to:
    a) create a subset of the data records that satisfy a condition.
    b) find suitable descriptive statistics for each column.
    c) draw boxplot for the numeric attributes and identify outliers, if any
    d) find correlation for each pair of numeric attributes, and draw scatter plot matrix.
    e) draw histograms, pie charts for categorical attributes.

f) draw the probability density curve for numeric attributes.

7. Read the CSV file, without headers and
   a. Find the number of records which are noise free
   b. Clean data after removing noise
   c. Normalize quantative features in range of [0,1]

8. Practical based on vectors, arrays and lists, data frames
9. Practical based on reading/writing data from/to files.
10. Practical based on data cleaning and transformation
11. Practical based on linear regression
12. Practical based on visualizing data as Scatter Plot, line graph, histogram, boxplot, line plots regression, word clouds, and exporting plots as images

Projects and Case Studies to be done as decided by the department in the beginning of the semester.

References

Cotton, R. (2017). *Learning R, A step by step function guide to data analysis.* O'reilly (SPD).

Additional Resources

1. Gardener, M. (2017). *Beginning R, The statistical programming language.* WILEY.
2. Teetor, P. (2017). *R Cookbook (10th Edition reprint).* O'reilly (SPD).

Web Resources

https://jrnold.github.io/r4ds-exercise-solutions/index.html

https://www.r-project.org/

https://cran.r-project.org/

Teaching Learning Process

- Talk and chalk method
- Computer based presentations by teachers.
- Group Discussions
- Assignments
- Offline and online Quiz
- Presentations by group of students for enhanced learning.

Tentative weekly teaching plan is as follows:

Week 1 - 4	**Unit 1 - Introduction to Programming Structures:** R interpreter, Introduction to major R data structures like

	vectors, matrices, arrays, list and data frames, Flow control and loops, looping over list and array, user-defined functions.
Week 5 - 7	**Unit 2 -File Handling:** Installing, loading and using packages for reading data from files
Week 8- 9	**Unit 3 - Data Preprocessing and Transformation:** Handling of missing data, Data cleaning and transformation
Week 10 - 12	**Unit 4 - Data Exploration:** Exploring data using statistical methods: mean, median, mode, quantiles. Building contingency table, correlation, co-variance.
Week 13 - 15	**Unit 5 - Plotting Data:** Data visualization using Scatter plot, Line graph, Histogram, Boxplot.

Assessment Methods

- Unit-wise assignments, presentations, viva, quiz,as announced by the instructor in the class
- End semester exam
- Internal assessment

Keywords

Data exploration, Data analytics, Data visualization, Statistical analysis

---

Programming in C++
(BSCS08A)
Skill-Enhancement Elective Course - (SEC) Credit:4

Course Objective

The course introduces Object Oriented Programming Language C++ with the objective to use object oriented features to develop efficient programs. The focus is to equip the students with adequate high-level object-oriented programming features using C++.

Course Learning Outcomes

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

1. solve simple programming problems using iteration and selection, and basic constructs: structures, arrays and functions.
2. create classes and their objects and use access specifiers for data hiding depicting advantage of Abstraction.
3. construct classes for code reusability depicting advantage of Inheritance.
4. implement Function Overloading depicting advantage of Polymorphism.
5. create file, read/write from/to files.

Unit 1

Introduction to C++: Need and characteristics of Object-Oriented Programming, Structure of a C++ Program (main () function, header files, output, input, comments), compile and execute a simple program.

Unit 2

Data types and Expression: Keywords, built in data types, variables and constants, naming convention, Input-Output statements, expressions and operators, precedence of operators, typecasting, library functions.

Unit 3

Control Constructs in C++ : Decision making using selection constructs, looping constructs , control constructs.

Unit 4

User defined Data types and functions: User defined data types, defining and initializing structures, derived data types, defining and initializing single and multi dimensional arrays, and user defined functions, passing arguments to functions, returning values from functions, inline functions, default arguments.

Unit 5

Classes and Objects: Need of abstraction, encapsulation, inheritance and polymorphism, creating classes, objects as function arguments, modifiers and access control, constructors and destructors, Implementation of single level inheritance, implementation of polymorphism, function overloading.

Unit 6

File Handling: File I/O Basics, read and write operations.

Practical

1. Write a program to find the largest of n natural numbers.
2. Write a program to find whether a given number is prime or not.
3. Write a menu driven program for following:
   a) display a Fibonacci series
   b) compute Factorial of a number
   c) to check whether a given number is odd or even.
   d) to check whether a given string is palindrome or not.
4. Write a program to print the sum and product of digits of an Integer and reverse the Integer.
5. Write a program to create an array of 10 integers. Accept values from the user in that array. Input another number from the user and find out how many numbers are equal to the number passed, how many are greater and how many are less than the number passed.
6. Write a program that will prompt the user for a list of 5 prices. Compute the average of the prices and find out all the prices that are higher than the calculated average.
7. Design a class named Car, having registration number, model and engine as its private members. Here engine is an object of a class called Engine with the private members: Chassis number and make. Define a suitable constructor of Car and override toString() Method to print the details of a car. Assume appropriate data types for the instance Members of the classes. Write a Java program to test the above class.
8. Write a program that computes the area of a circle, rectangle and a Cylinder using function overloading.

References

1. Lafore, R. *Object Oriented Programming in C++ (4th Edition).* SAMS Publishing.

Additional Resources

1. Balaguruswamy, E. (2017). *Object Oriented Programming with C++ (7th edition).* McGraw-Hill Education.
2. Kanetkar, Y. P. (2015). *Let us C++ (2nd Edition).* BPB Publishers.
3. Stroustrup, B. (2013). *The C++ Programming Language (4th Edition).* Pearson Education.

Teaching Learning Process

- Talk and chalk method
- Computer based presentations by teachers.
- Group Discussions
- Assignments
- Offline and online Quiz
- Presentations by group of students for enhanced learning.

Tentative weekly teaching plan is as follows:

Week 1 - 2	**Unit 1- Introduction to C++ :** Need and characteristics of Object-Oriented Programming, Structure of a C++ Program (main () function, header files, Output, Input, comments), compile and execute a simple program
Week 3 - 4	**Unit 2-Data types and Expression:** Keywords, built in data types, variables and constants, naming convention, Input-Output statements, expressions and operators, precedence of operators, typecasting, library functions.
Week 5 - 7	**Unit 3-Control Constructs in C++ :** Decision making using selection constructs, looping constructs , control constructs.
Week 7 - 9	**Unit 4 -User defined Data types and functions:** User defined data types, defining and initializing structures, derived data types, defining and initializing single and multi dimensional arrays, and user defined functions, passing arguments to functions, returning values from functions, inline functions, default arguments.
Week 10 - 13	**Unit 5 -Classes and Objects:** Need of abstraction, encapsulation, inheritance and polymorphism, creating classes, objects as function arguments, modifiers and access control, constructors and destructors, Implementation of single level inheritance, implementation of polymorphism, function overloading.
Week 14 - 15	**Unit 6-File Handling:** File I/O basics, read and write operations.

Assessment Methods

- Unit-wise assignments, presentations, viva, quiz,as announced by the instructor in the class
- End semester exam
- Internal assessment

Keywords

Abstraction, Encapsulation, Inheritance and Polymorphism

Programming in Java
(BSCS08B)
Skill-Enhancement Elective Course - (SEC) Credit:4

Course Objective

This course introduces fundamental concepts of Object Oriented Programming using Java. Basic concepts such as data types, expressions, control structures, functions and arrays are covered. Students are exposed to extensive Java programming to solve practical programming problems.

Course Learning Outcomes

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

1. develop and execute Java programs using iteration and selection.
2. create classes and their objects.
3. implement OOPS concepts to solve problems using JAVA

Unit 1

Introduction to Java: Features of Java, JDK environment, structure of Java programs

Unit 2

Programming Fundamentals: Data types, variables, operators, expressions, arrays, keywords, naming convention, decision making constructs , iteration, type casting, methods.

Unit 3

Object Oriented Programming Overview: Abstraction, encapsulation, inheritance, polymorphism.

Unit 4

Classes and Objects: Creating classes and objects, modifiers and access control, constructors, implementation of single and multilevel inheritance, implementation of polymorphism using overloading, overriding and dynamic method dispatch.

Unit 5

Strings: String class methods, string buffer methods.

Practical

9. Write a program to find the largest of n natural numbers.
10. Write a program to find whether a given number is prime or not.
11. Write a menu driven program for following:
    a) display a Fibonacci series
    b) compute Factorial of a number
    c) to check whether a given number is odd or even.
    d) to check whether a given string is palindrome or not.
12. Write a program to print the sum and product of digits of an Integer and reverse the Integer.
13. Write a program to create an array of 10 integers. Accept values from the user in that array. Input another number from the user and find out how many numbers are equal to the number passed, how many are greater and how many are less than the number passed.
14. Write a program that will prompt the user for a list of 5 prices. Compute the average of the prices and find out all the prices that are higher than the calculated average.
15. Design a class named Car, having registration number, model and engine as its private members. Here engine is an object of a class called Engine with the private members: Chassis number and make. Define a suitable constructor of Car and override toString() Method to print the details of a car. Assume appropriate data types for the instance Members of the classes. Write a Java program to test the above class.
16. Write a program that computes the area of a circle, rectangle and a Cylinder using function overloading.

References

1. Horstmann, C. S. (2017). *Core Java - Vol. I – Fundamentals (10th Edition).* Pearson.

Additional Resources

1. Balagurusamy, E. (2014). *Programming with JAVA: A Primer (5th Edition).* McGraw Hill Education (India) Private Limited.
2. Schildt, H. (2018). *Java: The Complete Reference (10th Edition).* McGraw-Hill Education.
3. Schildt, H. & Skrien, D. (2013). *Java Fundamentals - A comprehensive Introduction.* TMH.

Teaching Learning Process

- Talk and chalk method
- Computer based presentations by teachers.
- Group Discussions
- Assignments

- Offline and online Quiz
- Presentations by group of students for enhanced learning.

Tentative weekly teaching plan is as follows:

Week 1 - 2	**Unit 1 - Introduction to Java** Features of Java, JDK environment, structure of Java programs
Week 3 - 5	**Unit 2 - Programming Fundamentals** Data types, variables, operators, expressions, arrays, keywords, naming convention, decision making constructs , iteration, type casting, methods.
Week 6 - 7	**Unit 3 - Object Oriented Programming Overview** Abstraction, encapsulation, inheritance, polymorphism
Week 8 - 12	**Unit 4 - Classes and Objects** Creating classes and objects, modifiers and access control, constructors, implementation of single and multilevel inheritance, implementation of polymorphism using overloading, overriding and dynamic method dispatch
Week 13 - 15	**Unit 5 - Strings** String class methods, string buffer methods

Assessment Methods

- Unit-wise assignments, presentations, viva, quiz,as announced by the instructor in the class
- End semester exam
- Internal assessment

Keywords

Abstraction, encapsulation, inheritance, polymorphism.

Advanced Programming in Java
(BSCS09A)
Skill-Enhancement Elective Course - (SEC) Credit:4

Course Objective

This course builds over basic Java language skills acquired by the student in earlier semester. The students are exposed to the advanced features available in Java such as exception handling, file handling, interfaces, packages and GUI programming.

Course Learning Outcomes

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

1. implement Exception Handling and File Handling.
2. implement multiple inheritance using Interfaces.
3. logically organize classes and interfaces using packages
4. use AWT classes to design GUI applications.

Unit 1

Review of Object Oriented Programming and Java Fundamentals : Structure of Java programs, classes and objects, data types, type casting, looping constructs.

Unit 2

Interfaces: Interface basics, defining, implementing and extending interfaces; implementing multiple inheritance using interfaces.

Unit 3

Packages: Basics of packages, creating and accessing packages, system packages, creating user defined packages.

Unit 4

Exception Handling : Using the main keywords of exception handling: try, catch, throw, throws and finally, nested try, multiple catch statements, creating user defined exceptions.

Unit 5

File Handling : Byte stream, character stream, file I/O basics, file read/write operations.

Unit 6

GUI Programming : AWT classes, event handling.

Practical

1) Implement Abstraction, Encapsulation, Inheritance, Polymorphism
2) Implement multiple inheritance using interfaces
3) Create user defined packages
4) Create user defined exceptions
5) Implement file operations
6) Create Applets

More exercises as announced by instructor in the laboratory.

References

1.Horstmann, C. S. (2017). *Core Java - Vol. I – Fundamentals (10th Edition)*. Pearson.

Additional Resources

1. Balagurusamy, E. (2014). *Programming with JAVA: A Primer (5th Edition)*. McGraw Hill Education (India) Private Limited.
2. Schildt, H. (2018). *Java: The Complete Reference (10th Edition).* McGraw-Hill Education.

Teaching Learning Process

- Talk and chalk method
- Computer based presentations by teachers.
- Group Discussions
- Assignments
- Offline and online Quiz
- Presentations by group of students for enhanced learning.

Tentative weekly teaching plan is as follows:

Week 1 - 2	**Unit 1 - Review of Object Oriented Programming and Java Fundamentals** Structure of Java programs, classes and objects, data types, type casting, looping constructs.
Week 3 - 6	**Unit 2- Interfaces** Interface basics, defining, implementing and extending interfaces; implementing multiple inheritance using interfaces
Week 7 - 8	**Unit 3 - Packages** Basics of packages, creating and accessing packages, system packages, creating user defined packages
Week 9 - 10	**Unit 4- Exception Handling** Using the main keywords of exception handling: try, catch, throw, throws and finally, nested try, multiple catch statements, creating user defined exceptions
Week 10 - 12	**Unit 5-File Handling :** Byte stream, character stream, file I/O basics, file read/write operations.
Week 13 - 15	**Unit 6 - GUI Programming :** AWT classes, event handling.

Assessment Methods

- Unit-wise assignments, presentations, viva, quiz,as announced by the instructor in the class
- End semester exam
- Internal assessment

Keywords

Interface, Byte stream,Exception Handling, AWT, Event handling.

Web Design using HTML5
(BSCS09B)
Skill-Enhancement Elective Course - (SEC) Credit:4

Course Objective

The course introduces the basics of HTML5 including CSS styling. It helps students learn how to plan and design effective web pages and producing effective websites.

Course Learning Outcomes

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

1. define the principles and basics of Web page design.
2. recognize the elements of HTML.
3. apply basic concepts of CSS.
4. publish web pages.

Unit 1

Introduction: Introduction to HTML: What is HTML, HTML Documents, Basic structure of an HTML document, creating an HTML document, markup tags, heading-paragraphs, line breaks, HTML tags.

Unit 2

Elements of HTML: Introduction to elements of HTML, Working with Text, Working with Lists, Tables and Frames, Working with Hyperlinks, Images and Multimedia, Working with Forms and controls.

Unit 3

Introduction to Cascading Style Sheets: Concept of CSS, Creating Style Sheet, CSS Properties, CSS Styling(Background, Text Format, Controlling Fonts), Working with block elements and objects, Working with Lists and Tables, CSS Id and Class, Box Model(Introduction, Border properties, Padding Properties, Margin properties).

Unit 4

CSS Advanced: CSS Advanced(Grouping, Dimension, Display, Positioning, Floating, Align, Pseudo class, Navigation Bar, Image Sprites, Attribute sector), CSS Color.

Unit 5

Web Designs: Creating page Layout and Site Designs, Creating the Web Site, Saving the web site, Working on the web site, Creating web site structure, Creating Titles for web pages, Themes-Publishing web sites.

Practical

1. Creating HTML documents with various elements
2. Implementing Cascading style sheets
3. Creating and hosting websites

More exercises as announced by instructor in the laboratory.
Mini project for creating and hosting websites.

References

1. Boehm, A., & Ruvalcaba, Z. (2018). *Munarch's HTML5 and CCS3 (4th Edition).* Mike Murach & Associates.

Additional Resources

1. Minnick, J. (2015). *Web Design with HTML5 and CSS3 (8th Edition).* Cengage Learning.

Teaching Learning Process

- Talk and chalk method
- Computer based presentations by teachers.
- Group Discussions
- Assignments
- Offline and online Quiz
- Presentations by group of students for enhanced learning.

Tentative weekly teaching plan is as follows:

Week 1 - 3	**Unit I - Introduction:** Introduction to HTML: What is HTML, HTML Documents, Basic structure of an HTML document, creating an HTML document, markup tags, heading-paragraphs, line breaks, HTML tags.
Week 4 - 6	**Unit 2 - Elements of HTML:** Introduction to elements of HTML, working with text, lists, tables, frames, hyperlinks, images, multimedia, forms and controls.
Week 7-10	**Unit 3-Introduction to Cascading Style Sheets:** Concept of CSS, Creating Style Sheet, CSS Properties, CSS Styling (Background, Text Format, Controlling Fonts), Working with block elements and objects, Working with Lists and Tables, CSS Id and Class, Box Model(Introduction, Border properties, Padding Properties, Margin properties).
Week 11 - 13	**Unit 4-CSS Advanced:** CSS Advanced(Grouping, Dimension, Display, Positioning, Floating, Align, Pseudo class, Navigation Bar, Image Sprites, Attribute sector), CSS Color.
Week 14 - 15	**Unit 5-Web Designs:** Creating page Layout and Site Designs.

Assessment Methods

- Unit-wise assignments, presentations, viva, quiz,as announced by the instructor in the class
- End semester exam
- Internal assessment

Keywords

Web Design, HTML, CSS, Web Publishing

Android Programming
(BSCS10A)
Skill-Enhancement Elective Course - (SEC) Credit:4

Course Objective

The course is designed for students to learn to develop android applications. They will learn android architecture and key principles underlying its design.

Course Learning Outcome

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

1. describe the design of Android operating system.
2. describe various components of Android applications.
3. design user interfaces using various widgets, dialog boxes, menus.
4. design application with interaction among various activities/applications using intents.
5. develop application(s) with database handling.

Unit 1

Introduction: Review to JAVA & OOPS Concepts, introduction to Android operating systems and its development tools, android architecture along with components including activities, view and view group, services, content providers, broadcast receivers, intents, parcels, instance state. Android virtual device manager, Android SDK manager, Android emulator, Dalvik debug monitor service and debug bridge.

Unit 2

User Interface Architecture: Application context, intents, explicit intents, returning results from activities, implicit intents, intent filter and intent resolution, and applications of implicit intents, activity life cycle, activity stack, application's priority and its process' states, fragments and its life cycle.

Unit 3

User Interface Design: Layouts, optimizing layout hierarchies, form widgets, text fields, button control, toggle buttons, spinners, auto complete textview, edittext, images, image buttons, menu, dialog.

Unit 4

Broadcast receivers: Broadcast sender, receiver, broadcasting events with intents, listening for broadcasts with broadcast receivers, broadcasting ordered intents, broadcasting sticky intents, pending intents.

Unit 5

Database using SQLite: SQLite, content values and cursors, creating SQLite databases, querying a database, adding, updating, and removing rows.

Practical

1. Create "Hello World" application. That will display "Hello World" in the middle of the screen in the emulator. Also display "Hello World" in the middle of the screen in the Android Phone.
2. Create an application with login module. (Check username and password).
3. Create spinner with strings taken from resource folder (res >> value folder) and on changing the spinner value, Image will change.

4. Create a menu with 5 options and and selected option should appear in text box.
5. Create a list of all courses in your college and on selecting a particular course teacher-in-charge of that course should appear at the bottom of the screen.
6. Create an application with three option buttons, on selecting a button colour of the screen will change.
7. Create and Login application as above. On successful login, pop up the message.
8. Create an application to Create, Insert, update, Delete and retrieve operation on the database.

References

1. Griffiths, D., & Griffiths, D. (2015). *Head First Android Development*. O'reilly.
2. Meier, R. (2012). *Professional Android™ 4 Application Development.* John Wiley & Sons, Inc..

Additional Resources

1. Murphy, M. L. (2018). *The Busy Coder's Guide to Android Development.* CommonsWare.
2. Phillips, B., Stewart, C., Hardy, B. & Marsicano, K. (2015). *Android Programming: The Big Nerd Ranch Guide*. Big Nerd Ranch, LLC.
3. Sheusi, J. C. (2013). *Android Application Development for Java Programmers.* Cengage Learning.

Teaching Learning Process

- Talk and chalk method
- Computer based presentations by teachers.
- Group Discussions
- Assignments
- Offline and online Quiz
- Presentations by group of students for enhanced learning.

Tentative weekly teaching plan is as follows:

Week 1 - 3	**Unit 1 - Introduction:** Review to JAVA & OOPS Concepts, introduction to Android operating systems and its development tools, android architecture along with components including activities, view and view group, services, content providers, broadcast receivers, intents, parcels, instance state. Android virtual device manager, Android SDK manager, Android emulator, Dalvik debug monitor service and debug bridge
Week 4 – 6	**Unit 2 - User Interface Architecture:** Application context, intents, explicit intents, returning results from activities, implicit intents, intent filter and intent resolution, and applications of implicit intents, activity life cycle, activity stack, application's priority and its process' states, fragments and its life cycle.

Week 7 – 8	**Unit 3 - User Interface Design:** Layouts, optimizing layout hierarchies, form widgets, text fields, button control, toggle buttons, spinners, auto complete textview, edittext, images, image buttons, menu, dialog.
Week 9 – 10	**Unit 4 - Broadcast receivers:** Broadcast sender, receiver, broadcasting events with intents, listening for broadcasts with broadcast receivers, broadcasting ordered intents, broadcasting sticky intents, pending intents.
Week 11 – 15	**Unit 5 - Database using SQLite:** SQLite, content values and cursors, creating SQLite databases, querying a database, adding, updating, and removing rows

Assessment Methods

- Unit-wise assignments, presentations, viva, quiz,as announced by the instructor in the class
- End semester exam
- Internal assessment

Keywords

Android architecture, Android virtual device manager, Android SDK manager, Android emulator, Broadcast senders & receivers

---

PHP Programming
(BSCS10B)
Skill-Enhancement Elective Course - (SEC) Credit:4

Course Objective

This course will introduce server side scripting to students through PHP programming language. They will learn to design web applications with a specific functionality, and dynamic websites requiring handling/processing data input by users.

Course Learning Outcomes

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

1. use different data types and control structures in PHP.
2. handle arrays and strings in PHP.
3. create dynamic interactive web pages with PHP.
4. use PHP built-in functions as well as define custom functions.
5. perform data validation in PHP.
6. manipulate and manage a database using PHP.

Unit 1

Introduction: Introduction to three tier web application development, front end, business layer and back end connectivity, role of PHP in web application development, software requirements.

Unit 2

Starting PHP Programming: Basics of PHP programming, variables, scope of a variable, expressions, operators, operator precedence, simple procedural scripts, decision making based on conditions, case structure, loops.

Unit 3

Modular Programming: Functions and objects, Passing parameters.

Unit 4

Strings and Arrays: Creating and accessing strings, built-in functions for string and string formatting, creating index based and associative array, accessing array elements.

Unit 5

Forms and form processing: Capturing form data, GET and POST form methods, processing of form data, and use of regular expressions.

Unit 6

Integrating PHP & DBMS: Connecting PHP and DBMS, creating database, defining database structure and accessing data stored in tables using PHP.

Practical

1) Write a PHP script to input three numbers and print the largest number.

2) Write a function to calculate the factorial of a number (non-negative integer), which accepts the number as an argument. Use this function to compute C(n,r) as:
3) WAP to check whether the given number is prime or not.
4) Write a PHP script to accept a string from user, and print its reverse as output.
5) Write a script to check if the input string consists of lowercase characters only.
6) Write a PHP script to check whether a string is palindrome or not? (A palindrome is word, phrase, or sequence that reads the same backward as forward, e.g., madam or nurses run)
7) WAP to sort an array of numbers.
8) Write a PHP script to remove whitespaces from a string.

Sample string : The quick brown fox

Expected Output : Thequickbrownfox

9) Write a PHP script to find the sum of first n odd numbers.
10) Create a login page, which asks the user for a username and password. On clicking submit, a welcome message should be displayed if the user is already registered (i.e.name is present in the database) otherwise error message should be displayed.
11) Write a PHP script that checks if a string contains another string.
12) Create a simple 'birthday countdown' script, the script will count the number of days between current day and birth day.
13) Create a script to construct the following pattern, using nested for loop:

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

14) Write a simple PHP program to check that emails are valid.
15) WAP to print first n even numbers.
16) $color = array('white', 'green', 'red') Write a PHP script which will display the colors in the following way : Output : white, green, red, • green • red • white
17) Using switch case and dropdown list display a "Hello" message depending on the language selected in drop down list.
18) Write a PHP program to print Fibonacci series using recursion.
19) Write a PHP script to replace the first 'the' of the following string with 'That'. Sample : 'the quick brown fox jumps over the lazy dog'.

References

1. Nixon, R. (2014). Learning PHP, MySQL, JavaScript, CSS & HTML5. 3rd Edition, O'reilly.

Additional Resources

1. Boronczyk, T., & Psinas, M. E. (2008). *PHP and MYSQL (Create-Modify-Reuse).* Wiley

India Private Limited.
2. Holzner, S. (2007). *PHP: The Complete Reference.* McGraw Hill Education (India).
3. Sklar, D., & Trachtenberg, A. (2014). *PHP Cookbook: Solutions & Examples for PHP Programmers*. O'Reilly Media.
4. Welling, L., Thompson, L. (2008). *PHP and MySQL Web Development. 4th Edition*, Addison-Wesley Professional.

Teaching Learning Process

- Talk and chalk method
- Computer based presentations by teachers.
- Group Discussions
- Assignments
- Offline and online Quiz
- Presentations by group of students for enhanced learning.

Tentative weekly teaching plan is as follows:

Week 1 – 2	**Unit 1 - Introduction:** Introduction to three tier web application development, front end, business layer and back end connectivity, role of PHP in web application development, software requirements
Week 3 – 4	**Unit 2 - Starting PHP Programming:** Basics of PHP programming, variables, scope of a variable, expressions, operators, operator precedence, simple procedural scripts, decision making based on conditions, case structure, loops
Week 5 – 6	**Unit 3 - Modular Programming:** Functions and objects, passing parameters
Week 7 – 10	**Unit 4 - Strings and Arrays:** Creating and accessing strings, built-in functions for string and string formatting, creating index based and associative array, accessing array elements
Week 11 – 12	**Unit 5 - Forms and form processing:** Capturing form data, GET and POST form methods, processing of form data, and use of regular expressions
Week 13 – 15	**Unit 6 - Integrating PHP & DBMS:** Connecting PHP and DBMS, creating database, defining

	database structure and accessing data stored in tables using PHP

Assessment Methods

- Unit-wise assignments, presentations, viva, quiz,as announced by the instructor in the class
- End semester exam
- Internal assessment

Keywords

Server-side scripting, Web applications, Dynamic Websites, Database integration.

Acknowledgement (in alphabetical order)

Alka Khurana	Hansraj College
Anamika Gupta	Shaheed Sukhdev College of Business Studies
Anuja Soni	Deen Dayal Upadhyay College
Aparna Datt	P.G.D.A.V College
Archna Singhal	Indraprastha College for Women
Arpita Aggarwal	P.G.D.A.V College
Arpita Sharma	Deen Dayal Upadhyay College
Aruna Jain	Bharati College
Bharti	Hans Raj College
Bharti Kumar	Shyam Lal College (Eve.)
Bhavna Gupta	Keshav Mahavidyalaya
Divya Kawatra	Hansraj College
Geetanjali Kher	Kirori Mal Colleage
Harita Ahuja	Acharya Narendra Dev College
Harmeet Kaur,	Hans Raj College
Hema Banati	Dayal Singh College
Manisha Bansal	Indraprastha College for Women
Manju Bhardwaj	Maitreyi College
Megha Khandelwal	Deptt. of Computer Science
Nagendra	Bhagini Nivedita College
Naveen Kumar	Deptt. of Computer Science
Neelima Gupta	Deptt. of Computer Science
Nidhi Arora	Kalindi College
Nisha	Deptt. of Computer Science
PK Hazra	Deptt. of Computer Science
Preeti Marwaha	Acharya Narendra Dev College
Priti Sehgal	Keshav Mahavidyalaya
Priyanka Rathi	Deptt. of Computer Science
Priyanka Sharma	Acharya Narendra Dev College
Punam Bedi	Deptt. of Computer Science
Rajni Bala	Deen Dayal Upadhyay College
Rakhi Saxena	Deshbandhu College
Rampal Rana	Deen Dayal Upadhyay College
Reena Kasana	Deptt. of Computer Science
Roli Bansal	Keshav Mahavidyalaya
Ronnie Chakre	Deptt. of Computer Science
Rupali Ahuja	Maitreyi College
S K Muttoo	Deptt. of Computer Science
Sahil Pathak	Ramanujan College
Sapna Varshney	Deptt. of Computer Science
Sarabjeet Kochhar	Indraprastha College for Women
Seema Aggarwal	Miranda House
Shalini Sharma	Kalindi College
Sharanjit Kaur	Acharya Narendra Dev College
Shikha Gupta	Shaheed Sukhdev College of Business Studies

Sonika Thakral	Shaheed Sukhdev College of Business Studies
Sujata Khatri	Deen Dayal Upadhyay College
Sunita Narang	Acharya Narendra Dev College
Vandana Kalra	SGGSCC
Vasudha Bhatnagar	Deptt. of Computer Science
Vinita Jindal	Keshav Mahavidyalaya
Vipin Kumar Rathi	Ramanujan College