University of Engineering & Management, Jaipur

Syllabus for BSc Computer Science Admission Batch 2026

SEMESTER 2

S. No.	Subject Code	Subject Name	L	T	P	Total Contact Hrs	Credit Points
Theory							
1	BSCCS201	Programming in Java	3	1	0	4	4
2	BSCCS202	Discrete Structure	3	1	0	4	4
3	BSCCS203	General Studies & Current Affairs - II	3	1	0	4	4
4	BSCCS204	Universal Human Values	0	2	0	2	2
Practical							
5	BSCCS291	Programming in Java Lab	0	0	2	2	2
6	BSCCS292	Discrete Structure Tutorial	0	0	2	2	2
Sessional							
7	BSCCS293	Programming in Python	0	2	0	2	2
		Total	9	7	4	20	20

Subject Code: BSCCS201

Subject Name: Programming in Java

Credit Points: 04

Course Objective

This course adds to the basic programming language skills acquired by the student in earlier semesters. 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, the student will:

1. Implement Exception Handling and File Handling.
2. Implement multiple inheritance using Interfaces.
3. Logically organize classes and interfaces using packages.
4. Use AWT and Swing to design GUI applications.

Detailed Syllabus

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 Packages Basics of packages, Creating and accessing packages, System packages, Creating user defined packages

Unit 3

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 4

File Handling Byte Stream, Character Stream, File I/O Basics, File Operations

Unit 5

AWT and Event Handling: The AWT class hierarchy, Events, Event sources, Event classes, Event Listeners, Relationship between Event sources and Listeners, Delegation event model, Creating GUI applications using AWT.

Unit 6

Swing Introduction to Swing, Swing vs. AWT, Hierarchy for Swing components, Creating GUI applications using Swing.

References

1. Schildt, H. (2018). *Java: The Complete Reference*. 10th edition. McGraw-Hill Education.

Additional Resources:

1. Balaguruswamy E. (2014). *Programming with JAVA: A Primer*. 5th edition. India: McGraw Hill Education
2. Horstmann, C. S. (2017). *Core Java - Vol. I – Fundamentals* (Vol. 10). Pearson Education
3. Schildt, H., & Skrien, D. (2012). *Java Fundamentals - A Comprehensive Introduction*. India: McGraw Hill Education.

Subject Code: BSCCS291

Subject Name: Programming in Java Lab

Credit Points: 02

Practical

1. Design a class Complex having a real part (x) and an imaginary part (y). Provide methods to perform the following on complex numbers:
    1. Add two complex numbers.
    2. Multiply two complex numbers.
    3. toString() method to display complex numbers in the form: x + iy

2. Create a class TwoDim which contains private members as x and y coordinates in package P1. Define the default constructor, a parameterized constructor and override toString() method to display the co-ordinates. Now reuse this class and in package P2 create another class ThreeDim, adding a new dimension as z as its private member. Define the constructors for the subclass and override toString() method in the subclass also. Write appropriate methods to show dynamic method dispatch. The main() function should be in a package P.

3. Define an abstract class Shape in package P1. Inherit two more classes: Rectangle in package P2 and Circle in package P3. Write a program to ask the user for the type of shape and then using the concept of dynamic method dispatch, display the area of the appropriate subclass. Also write appropriate methods to read the data. The main() function should not be in any package.

4. Create an exception subclass UnderAge, which prints “Under Age” along with the age value when an object of UnderAge class is printed in the catch statement. Write a class exceptionDemo in which the method test() throws UnderAge exception if the variable age

passed to it as argument is less than 18. Write main() method also to show working of the program.

5. Write a program to implement stack. Use exception handling to manage underflow and overflow conditions.

6. Write a program that copies content of one file to another. Pass the names of the files through command-line arguments.

7. Write a program to read a file and display only those lines that have the first two characters as '//' (Use try with resources).

8. Write a program to create an Applet. Create a frame as a child of applet. Implement mouseClicked( ), mouseEntered( ) and mouseExited( ) events for applet. Frame is visible when mouse enters applet window and hidden when mouse exits from the applet window.

9. Write a program to display a string in frame window with pink color as background.

10. Write a program to create an Applet that has two buttons named “Red” and “Blue”. When a button is pressed the background color of the applet is set to the color named by the button’s label.

11. Create an applet which responds to KEY_TYPED event and updates the status window with message (“Typed character is: X”). Use adapter class for other two events.

12. Create an applet with two buttons labelled ‘A’ and ‘B’. When button ‘A’ is pressed, it displays your personal information (Name, Course, Roll No, College) and when button ‘B’ is pressed, it displays your CGPA in the previous semester.

13. Write a program that creates a Banner and then creates a thread to scrolls the message in the banner from left to right across the applet's window.

14. Rewrite the applet programs using Swing.

Subject Code: BSCCS202

Subject Name: Discrete Structure

Credit Points: 04

Course Objective

The course aims to introduce the students to Boolean algebra, sets, relations, functions, principles of counting, and growth functions so that these concepts may be used effectively in other courses.

Course Learning Outcomes

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

1. Define mathematical structures (relations, functions, sequences, series, and graphs) and use them to model real-life situations.
2. Understand (trace) and construct simple mathematical proofs using logical arguments.
3. Solve classroom puzzles based on counting principles.
4. Compare functions and relations with respect to their growth for large values of the input.

Detailed Syllabus

Unit 1

Introduction: Sets - finite and infinite sets, uncountable infinite sets; functions, relations, properties of binary relations, closure, partial ordering relations; counting - Pigeonhole Principle, permutation and combination; mathematical induction, Principle of Inclusion and Exclusion.

Unit 2

Growth of Functions: asymptotic notations, summation formulas and properties, bounding summations, approximation by integrals.

Unit 3

Recurrence: recurrence relations, generating functions, linear recurrence relations with constant coefficients and their solution, recursion trees, Master Theorem

Unit 4

Graph Theory: basic terminology, models and types, multi-graphs and weighted graphs, graph representation, graph isomorphism, connectivity, Euler and Hamiltonian Paths and Circuits, planar graphs, graph coloring, Trees, basic terminology and properties of Trees, introduction to spanning trees.

Unit 5

Propositional Logic: logical connectives, well-formed formulas, tautologies, equivalences, Inference Theory

References

1. Mohapatra, & Liu, C. L. (2012). *Elements of Discrete mathematics*. 4th edition. McGraw Hill Education.

2. Rosen, K. H. (2011). *Discrete Mathematics and Its Applications*. 7th edition. Tata McGraw Hill Education.

Additional Resources

1. Albertson, M. O., & Hutchinson, J.P., (1988). *Discrete Mathematics with Algorithms.* John Wiley and Sons.
2. Cormen, T. H., Leiserson, C. E., & Rivest, R. L. (2009). *Introduction to algorithms*. 3rd edition. MIT Press.
3. Hein, J. L. (2015). *Discrete Structures, Logic, and Computability.* 4th edition. Jones and Bartlett Learning.
4. Hunter, D. J. (2011). *Essentials of Discrete Mathematics.* 2nd edition. Jones and Bartlett Learning

Subject Code: BSCCS202

Subject Name: Discrete Structure Tutorial

Credit Points: 02

Practical

1. Write a Program to create a SET A and determine the cardinality of SET for an input array of elements (repetition allowed) and perform the following operations on the SET:
    a) ismember (a, A): check whether an element belongs to set or not and return value as true/false.
    b) powerset(A): list all the elements of power set of A.
2. Create a class SET and take two sets as input from user to perform following SET Operations:
    a) Subset: Check whether one set is a subset of other or not.
    b) Union and Intersection of two Sets.
    c) Complement: Assume Universal Set as per the input elements from the user.
    d) Set Difference and Symmetric Difference between two SETS
    e) Cartesian Product of Sets.
3. Create a class RELATION, use Matrix notation to represent a relation. Include functions to check if the relation is Reflexive, Symmetric, Antisymmetric, and Transitive. Write a Program to use this class.
4. Use the functions defined in Ques 3 to check whether the given relation is:
    a) Equivalent, or
    b) Partial Order relation, or
    c) None
5. Write a Program to implement Bubble Sort. Find the number of comparisons during each pass and display the intermediate result.

Use the observed values to plot a graph to analyse the complexity of algorithm.

6. Write a Program to implement Insertion Sort. Find the number of comparisons during each pass and display the intermediate result. Use the observed values to plot a graph to analyse the complexity of algorithm.
7. Write a Program that generates all the permutations of a given set of digits, with or without repetition. (For example, if the given set is {1,2}, the permutations are 12 and 21). (One method is given in Liu)
8. Write a Program to calculate Permutation and Combination for an input value n and r using recursive formula of ${}^{n}Cr$ and ${}^{n}Pr$.
9. For any number n, write a program to list all the solutions of the equation x1 + x2 + x3 + …+ xn = C, where C is a constant (C<=10) and x1, x2, x3, …, xn are nonnegative integers using brute force strategy.
10. Write a Program to accept the truth values of variables x and y, and print the truth table of the following logical operations:

    a) Conjunction b) Disjunction c) Exclusive OR d) Conditional e) Bi-conditional

    f) Exclusive NOR g) Negation h) NAND i) NOR
11. Write a Program to store a function (polynomial/exponential), and then evaluate the polynomial. (For example, store f(x) = 4n3 + 2n + 9 in an array and for a given value of n, say n = 5, evaluate (i.e. compute the value of f(5)).
12. Write a Program to represent Graphs using the Adjacency Matrices and check if it is a complete graph.
13. Write a Program to accept a directed graph G and compute the in-degree and out-degree of each vertex.
14. Given a graph G, write a Program to find the number of paths of length n between the source and destination entered by the user.
15. Given an adjacency matrix of a graph, write a program to check whether a given set of vertices {v1, v2, v3, ..., vk} forms an Euler Path / Euler Circuit (for circuit assume vk=v1).
16. Given a full m-ary tree with i internal vertices, write a Program to find the number of leaf nodes.

Subject Code: BSCCS293

Subject Name: Programming in Python

Credit Points: 02

Course Objective

This course is designed to introduce the student to the basics of programming using Python. The course covers the topics essential for developing well-documented modular programs using different instructions and built-in data structures available in Python.

Course Learning Outcomes

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

1. Develop, document, and debug modular Python programs to solve computational problems.
2. Select a suitable programming construct and data structure for a situation.
3. Use built-in strings, lists, sets, tuples and dictionaries in applications. 4. Define classes and use them in applications.
4. Use files for I/O operations.

Detailed Syllabus

Unit 1

Introduction to Programming using Python: Structure of a Python Program, Functions, Interpreter shell, Indentation. Identifiers and keywords, Literals, Strings, Basic operators (Arithmetic operator, Relational operator, Logical or Boolean operator, Assignment Operator, Bit wise operator).

Unit 2

Building blocks of Python: Standard libraries in Python, notion of class, object and method.

Unit 3

Creating Python Programs: Input and Output Statements, Control statements: branching, looping, Exit function, break, continue and pass, mutable and immutable structures. Testing and debugging a program

Unit 4

Built-in data structures: Strings, lists, Sets, Tuples and Dictionary and associated operations. Basic searching and sorting methods using iteration and recursion.

Unit 5

Visualization using 2D and 3D graphics: Visualization using graphical objects like Point, Line, Histogram, Sine and Cosine Curve, 3D objects

Unit 6

Exception Handling and File Handling: Reading and writing text and structured files, Errors and Exceptions.

References

1. Downey, A.B., (2015), Think Python–How to think like a Computer Scientist, 3rd edition. O'Reilly Media.

2. Taneja, S. & Kumar, N., (2017), Python Programming- A Modular Approach. Pearson Education.

Additional Resources

1. Brown, M. C. (2001). The Complete Reference: Python, McGraw Hill Education.
2. Dromey, R. G. (2006), How to Solve it by Computer, Pearson Education.
3. Guttag, J.V.(2016), Introduction to computation and programming using Python. MIT Press.
4. Liang, Y.D. (2013), Introduction to programming using Python. Pearson Education.