BUN
ISZ

7. Write an assembly language program to simulate the machine for following register reference instructions and determine the contents of AC, E, PC, AR and IR registers in decimal after the execution:
   CLA
   CMA
   CME
   HLT
8. Write an assembly language program to simulate the machine for following register reference instructions and determine the contents of AC, E, PC, AR and IR registers in decimal after the execution
   INC
   SPA
   SNA
   SZE

9. Write an assembly language program to simulate the machine for following register reference instructions and determine the contents of AC, E, PC, AR and IR registers in decimal after the execution:
   CIR
   CIL \
10. Write an assembly program that reads in integers and adds them together; until a negative non-zero number is read in. Then it outputs the sum (not including the last number).
11. Write an assembly program that reads in integers and adds them together; until zero is read in. Then it outputs the sum.

Note: Examination scheme and mode shall be as prescribed by the Examination Branch, University of Delhi, from time to time.

DSC07/DSC02/GE4a: DATA STRUCTURES

CREDIT DISTRIBUTION, ELIGIBILITY AND PRE-REQUISITES OF THE COURSE

	Credits	Credit distribution of the course		

Course title & Code		Lecture	Tutorial	Practical /Practice	Eligibility criteria	Pre-requisite of the course (if any)
Data Structures	**4**	**3**	**0**	**1**	**Passed 12th class with Mathematics**	Programming using C++

Course Objectives

The course aims at developing the ability to use basic data structures like arrays, stacks, queues, lists, and trees to solve problems. C++ is chosen as the language to implement the implementation of these data structures.

Learning Outcomes

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

● Compare two functions for their rates of growth.

● Understand abstract specification of data-structures and their implementation.

● Compute time and space complexity of operations on a data-structure.

● Identify the appropriate data structure(s) for a given application and understand the trade-offs involved in terms of time and space complexity.

● Apply recursive techniques to solve problems.

Syllabus

Unit-1 **(9 hours)**

Growth of Functions, Recurrence Relations: Functions used in analysis, asymptotic notations, asymptotic analysis, recurrence, Master Theorem.

Unit-2 **(16 hours)**

Arrays, Linked Lists, Stacks, Queues: Arrays: array operations, applications, two dimensional arrays, dynamic allocation of arrays; Linked Lists: singly linked lists, doubly linked lists, circularly linked lists, Stacks: stack as an ADT, implementing stacks using arrays, implementing stacks using

linked lists, applications of stacks; Queues: queue as an ADT, implementing queues using arrays, implementing queues using linked lists,. Time complexity analysis.

Unit-3 **(5 hours)**

Recursion: Recursive functions, linear recursion, binary recursion.

Unit-4 **(6 hours)**

Trees, Binary Trees: Trees: definition and properties, tree traversal algorithms, and their time complexity analysis; binary trees: definition and properties, traversal of binary trees, and their time complexity analysis.

Unit-5 **(7 hours)**

Binary Search Trees: Binary Search Trees: insert, delete, search operations, time complexity analysis of these operations

Unit-6 **(2 hours)**

Binary Heap: Binary Heaps: heaps, heap operations.

Essential/recommended readings

1. Goodrich, M.T., Tamassia, R., & Mount, D., Data Structures and Algorithms Analysis in C++, 2nd edition, Wiley, 2011. 4 th
2. Cormen, T.H., Leiserson, C.E., Rivest, R. L., Stein C. Introduction to Algorithms, edition, Prentice Hall of India, 2022. Additional references

Additional References

1. Sahni, S. Data Structures, Algorithms and applications in C++, 2nd edition, Universities Press, 2011.
2. Langsam Y., Augenstein, M. J., & Tanenbaum, A. M. Data Structures Using C and C++, Pearson, 2009.

Practical List **(30 hours)**

1. Write a program to implement singly linked list as an ADT that supports the following operations:
   a. Insert an element x at the beginning of the singly linked list
   b. Insert an element x at ith position in the singly linked list
   c. Remove an element from the beginning of the singly linked list
   d. Remove an element from ith position in the singly link

e. Search for an element x in the singly linked list and return its pointer
f. Concatenate two singly linked lists

2. Write a program to implement doubly linked list as an ADT that supports the following operations:
   a. Insert an element x at the beginning of the doubly linked list
   b. Insert an element x at ith position in the doubly linked list
   c. Insert an element x at the end of the doubly linked list
   d. Remove an element from the beginning of the doubly linked list
   e. Remove an element from ith position in the doubly linked list.
   f. Remove an element from the end of the doubly linked list
   g. Search for an element x in the doubly linked list and return its pointer (viii) Concatenate two doubly linked lists
3. Write a program to implement circular linked list as an ADT which supports the following operations:
   a. Insert an element x at the front of the circularly linked list
   b. Insert an element x after an element y in the circularly linked list
   c. Insert an element x at the back of the circularly linked list
   d. Remove an element from the back of the circularly linked list
   e. Remove an element from the front of the circularly linked list
   f. Remove the element x from the circularly linked list
   g. Search for an element x in the circularly linked list and return its pointer
   h. Concatenate two circularly linked lists
4. Implement a stack as an ADT using Arrays.
5. Implement a stack as an ADT using the Linked List ADT.
6. Write a program to evaluate a prefix/postfix expression using stacks.
7. Implement Queue as an ADT using the circular Arrays.
8. Implement Queue as an ADT using the Circular Linked List ADT.
9. Write a program to implement Binary Search Tree as an ADT having the following operations:
   a. Insert an element x
   b. Delete an element x
   c. Search for an element x in the BST and change its value to y and then place the node with value y at its appropriate position in the BST
   d. Display the elements of the BST in preorder, inorder, and postorder traversal
   e. Display the elements of the BST in level-by-level traversal
   f. Display the height of the BST

Note: Examination scheme and mode shall be as prescribed by the Examination Branch, University of Delhi, from time to time.

DSC08/DSC04/GE5a: OPERATING SYSTEMS

CREDIT DISTRIBUTION, ELIGIBILITY AND PRE-REQUISITES OF THE COURSE

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course (if any)
		Lecture	Tutorial	Practical / Practice		
Operating Systems	4	3	0	1	Passed 12th class with Mathematics	Programming using C++ / Python/Java

Course Objectives

The course provides concepts that underlie all operating systems and are not tied to any particular operating system. The emphasis is on explaining the need and structure of an operating system using its common services such as process management (creation, termination etc.), CPU Scheduling, Process Synchronization, Handling Deadlocks, main memory management, virtual memory, secondary memory management. The course also introduces various scheduling algorithms, structures, and techniques used by operating systems to provide these services.

Learning Outcomes

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

● Describe the need of an operating system and define multiprogramming and Multithreading concepts.
● Implement the process synchronization service (Critical Section, Semaphores), CPU scheduling service with various algorithms.
● Implement Main memory Management (Paging, Segmentation) algorithms, Handling of Deadlocks
● Identify and appreciate the File systems Services, Disk Scheduling service

Syllabus

Unit-1 **(6 hours)**

Introduction: Operating Systems (OS) definition and its purpose, OS Structure, OS Operations: Dual and Multi-mode, OS as resource manager.

Unit-2 **(9 hours)**

Operating System Structures: OS Services, System Calls: Process Control, File Management, Device Management, and Information Maintenance, Inter-process Communication, and Protection, System programs, OS structure- Simple, Layered, Microkernel, and Modular.

Unit-3 **(10 hours)**

Process Management: Process Concept, States, Process Control Block, Process Scheduling, Schedulers, Context Switch, Operation on processes, Threads, Multicore Programming, Multithreading Models, Process Scheduling Algorithms: First Come First Served, Shortest-Job-First, Priority & Round-Robin, Process Synchronization: The critical section problem, Deadlock characterization, Deadlock handling.

Unit-4 **(11 hours)**

Memory Management: Physical and Logical address space, Swapping, Contiguous memory allocation strategies - fixed and variable partitions, Segmentation, Paging. Virtual Memory Management: Demand Paging and Page Replacement algorithms: FIFO Page Replacement, Optimal Page replacement, LRU page replacement.

Unit-5 **(9 hours)**

File System: File Concepts, File Attributes, File Access Methods, Directory Structure: Single Level, Two-Level, Tree-Structured, and Acyclic-Graph Directories. Mass Storage Structure: Magnetic Disks, Solid-State Disks, Magnetic Tapes, Disk Scheduling algorithms: FCFS, SSTF, SCAN, C-SCAN, LOOK, and C-LOOk Scheduling.

Essential/recommended readings

1. Silberschatz, A., Galvin, P. B., Gagne G. Operating System Concepts, 9 th edition, John Wiley Publications, 2016.

2. Tanenbaum, A. S. Modern Operating Systems, 3 rd edition, Pearson Education, 2007.
3. Stallings, W. Operating Systems: Internals and Design Principles, 9 th edition, Pearson Education, 2018.

Additional References

1. Dhamdhere, D. M., Operating Systems: A Concept-based Approach, 2nd edition, Tata McGraw-Hill Education, 2017.
2. Kernighan, B. W., Rob Pike, R. The Unix Programming Environment, Englewood Cliffs, NJ: Prentice-Hall, 1984.

Practicals

1. Execute various Linux commands for:
   a. Information Maintenance: wc, clear, cal, who, date, pwd
   b. File Management: cat, cp, rm, mv, cmp, comm, diff, find, grep, awk
   c. Directory Management : cd, mkdir, rmdir, ls
2. Execute various Linux commands for:
   a. Process Control: fork, getpid, ps, kill, sleep
   b. Communication: Input-output redirection, Pipe
   c. Protection Management: chmod, chown, chgrp
3. Write a programme (using fork() and/or exec() commands) where parent and child execute:
   a. same program, same code.
   b. same program, different code.
   c. Before terminating, the parent waits for the child to finish its task.
4. Write a program to to report behaviour of Linux kernel including kernel version, CPU type and model. (CPU information).
5. Write a program to report behaviour of Linux kernel including information on 19 configured memory, amount of free and used memory. (Memory information)
6. Write a program to copy files using system calls.
7. Use an operating system simulator to simulate operating system tasks.
8. Write a program to implement scheduling algorithms FCFS/ SJF/ SRTF/ non preemptive scheduling algorithms.
9. Write a program to calculate the sum of n numbers using Pthreads. A list of n numbers is divided into two smaller lists of equal size, and two separate threads are used to sum the sublists.
10. Write a program to implement first-fit, best-fit and worst-fit allocation strategies.

Note: Examination scheme and mode shall be as prescribed by the Examination Branch, University of Delhi, from time to time.

9. SVM classification

10. K-Means Clustering

11. Hierarchical Clustering

DSC 9 / DSC16/GE6e/DSE: ARTIFICIAL INTELLIGENCE

CREDIT DISTRIBUTION, ELIGIBILITY AND PRE-REQUISITES OF THE COURSE

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course
		Lecture	Tutorial	Practical/ Practice		
Artificial Intelligence	4	3	0	1	Pass in Class XII	Programming using C++/Programming using Python/Object Oriented Programming using Python

Course Objectives

The objectives of this course are to:

- To introduce basic concepts and techniques of Artificial Intelligence (AI).
- To apply informed search techniques for different applications.
- To learn various knowledge representation techniques and writing Prolog programs.
- To learn about the latest techniques for developing AI systems.

Learning outcomes

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

- identify problems that are amenable to solutions by specific AI methods.
- state the utility of different types of AI agents.
- apply different informed search techniques for solving problems.
- use knowledge representation techniques for AI systems.

SYLLABUS

Unit 1 **6 Hours**

Introduction: Introduction to artificial intelligence, background and applications, Turing test, Weak AI, Strong AI, Narrow AI, Artificial General Intelligence, Super AI, rational agent approaches to AI, introduction to intelligent agents, their structure, behavior and task environment.

Unit 2 **12 Hours**

Problem Solving and Searching Techniques: Problem characteristics, production systems, control strategies, breadth-first search, depth-first search, hill climbing and its variations, heuristics search techniques: best-first search, A* algorithm, constraint satisfaction problem, means-end analysis, introduction to game playing, min-max and alpha-beta pruning algorithms.

Unit 3 **16 Hours**

Knowledge Representation: Propositional logic, First-Order Predicate logic, resolution principle, unification, semantic nets, conceptual dependencies, frames, and scripts, production rules, Introduction to Programming in Logic (PROLOG).

Unit 4 **8 Hours**

Understanding Natural Languages: Components and steps of communication, the contrast between formal and natural languages in the context of grammar, Chomsky hierarchy of grammars, parsing, and semantics, Parsing Techniques, Context-Free and Transformational Grammars, Recursive transition nets.

Unit 5 **3 Hours**

AI The Present and the Future: Symbolic AI, Data-driven AI and Machine Learning, Introduction to Machine Learning and Deep Learning based AI, Interpretable and Explainable AI, Ethics of AI: benefits and risks of AI.

Essential/recommended readings

1. Russell, Stuart, J. and Norvig, Peter, *Artificial Intelligence - A Modern Approach*, Pearson, 4th edition, 2020..
2. Bratko, Ivan, *Prolog Programming for Artificial Intelligence*, Addison-Wesley, Pearson Education, 4th edition, 2012.
3. Patterson, DAN,W, *Introduction to A.I. and Expert Systems* – PHI, 2007.
4. Clocksin, W., F. and Mellish, *Programming in PROLOG*, 5th edition, Springer, 2003.

Additional references

1. Kaushik, Saroj, *Artificial Intelligence*, Cengage Learning India, 2011.
2. Rich, Elaine and Knight, Kelvin, *Artificial Intelligence*, 3rd edition, Tata McGraw Hill, 2010

Practical List :

Practical exercises such as

1. Write a program in Prolog to implement TowerOfHanoi(N) where N represents the number of disks.
2. Write a program to implement the Hill climbing search algorithm in Prolog.
3. Write a program to implement the Best first search algorithm in Prolog.
4. Write a program to implement A* search algorithm in Prolog.
5. Write a program to implement the min-max search algorithm in Prolog.
6. Write a program to solve the Water-Jug Problem in Prolog.
7. Implement sudoku problem (minimum 9×9 size) using constraint satisfaction in Prolog.
8. Write a Prolog program to implement the family tree and demonstrate the family relationship.
9. Write a Prolog program to implement knowledge representation using frames with appropriate examples.
10. Write a Prolog program to implement conc(L1, L2, L3) where L2 is the list to be appended with L1 to get the resulted list L3.
11. Write a Prolog program to implement reverse(L, R) where List L is original and List R is reversed list.
12. Write a Prolog program to generate a parse tree of a given sentence in English language assuming the grammar required for parsing.
13. Write a Prolog program to recognize context free grammar $a^n b^n$.

7. Display the data stored in a given graph using the Breadth-First Search algorithm.
8. Display the data stored in a given graph using the Depth-First Search algorithm.
9. Write a program to determine a minimum spanning tree of a graph using the Prim's algorithm.
10. Write a program to implement Dijkstra's algorithm to find the shortest paths from a given source node to all other nodes in a graph.
11. Write a program to solve the weighted interval scheduling problem.
12. Write a program to solve the 0-1 knapsack problem.

For the algorithms at S.No 1, 2 and 3 , test run the algorithm on 100 different input sizes varying from 30 to 1000. For each size find the number of comparisons averaged on 10 different input instances; plot a graph for the average number of comparisons against each input size. Compare it with a graph of nlogn.

DSC-A3/DSE: DATA MINING-I

CREDIT DISTRIBUTION, ELIGIBILITY AND PRE-REQUISITES OF THE COURSE

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course (if any)
		Lecture	Tutorial	Practical/ Practice		
Data Mining - I	4	3	0	1	Passed 12th class with Mathematics	Programming using Python

Course Objectives

This course aims to introduce data mining techniques and their application on real-life datasets. The students will learn to pre-process the dataset and make it ready for application of data mining techniques. The course will focus on three main techniques of data mining i.e. Classification, Clustering and Association Rule Mining. Different algorithms for these techniques

will be discussed along with appropriate evaluation metrics to judge the performance of the results delivered.

Learning outcomes

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

● Pre-process the data for subsequent data mining tasks

● Apply a suitable classification algorithm to train the classifier and evaluate its performance.

● Apply appropriate clustering algorithm to cluster the data and evaluate clustering quality

● Use association rule mining algorithms and generate frequent item-sets and association rules

Syllabus

Unit 1 **(8 hours)**

Introduction to Data Mining:

Motivation and Challenges for data mining, Types of data mining tasks, Applications of data mining, Data measurements, Data quality, Supervised vs. unsupervised techniques

Unit 2 **(9 hours)**

Data Pre-Processing:

Data aggregation, sampling, dimensionality reduction, feature subset selection, feature creation, variable transformation.

Unit 3 **(11 hours)**

Cluster Analysis:

Basic concepts of clustering, measure of similarity, types of clusters and clustering methods, K-means algorithm, measures for cluster validation, determine optimal number of clusters

Unit 4 **(8 hours)**

Association Rule Mining:

Transaction data-set, frequent itemset, support measure, rule generation, confidence of association rule, Apriori algorithm, Apriori principle

Unit 5 **(9 hours)**

Classification:

Naive Bayes classifier, Nearest Neighbour classifier, decision tree, overfitting, confusion matrix, evaluation metrics and model evaluation.

Essential/recommended readings

1. Tan P.N., Steinbach M, Karpatne A. and Kumar V. Introduction to Data Mining,2nd edition, Pearson, 2021.

2. Han J., Kamber M. and Pei J. Data Mining: Concepts and Techniques, 3 rd edition, 2011, Morgan Kaufmann Publishers.

3. Zaki M. J. and Meira J. Jr. Data Mining and Machine Learning: Fundamental Concepts and Algorithms, 2nd edition, Cambridge University Press, 2020.

Additional References

1. Aggarwal C. C. Data Mining: The Textbook, Springer, 2015.
2. Dunham M. Data Mining: Introductory and Advanced Topics, 1st edition, Pearson Education India, 2006.

Recommended Datasets for :

Classification: Abalone, Artificial Characters, Breast Cancer Wisconsin (Diagnostic)

Clustering: Grammatical Facial Expressions, HTRU2, Perfume data

Association Rule Mining: MovieLens, Titanics

Practicals

1. Apply data cleaning techniques on any dataset (e,g, wine dataset). Techniques may include handling missing values, outliers, inconsistent values. A set of validation rules can be prepared based on the dataset and validations can be performed.
2. Apply data pre-processing techniques such as standardization/normalization, transformation, aggregation, discretization/binarization, sampling etc. on any dataset

3. Run Apriori algorithm to find frequent itemsets and association rules on 2 real datasets and use appropriate evaluation measures to compute correctness of obtained patterns a) Use minimum support as 50% and minimum confidence as 75% b) Use minimum support as 60% and minimum confidence as 60 % I.
4. Use Naive bayes, K-nearest, and Decision tree classification algorithms and build classifiers on any two datasets. Divide the data set into training and test set. Compare the accuracy of the different classifiers under the following situations: a) Training set = 75% Test set = 25% b) Training set = 66.6% (2/3rd of total), Test set = 33.3% II. Training set is chosen by i) hold out method ii) Random subsampling iii) Cross-Validation. Compare the accuracy of the classifiers obtained. Data is scaled to standard format.
5. Use Simple K-means algorithm for clustering on any dataset. Compare the performance of clusters by changing the parameters involved in the algorithm. Plot MSE computed after each iteration using a line plot for any set of parameters.

Project: Students should be promoted to take up one project on any UCI/kaggle/data.gov.in or a dataset verified by the teacher. Preprocessing steps and at least one data mining technique should be shown on the selected dataset. This will allow the students to have a practical knowledge of how to apply the various skills learnt in the subject for a single problem/project.

Note: Examination scheme and mode shall be as prescribed by the Examination Branch, University of Delhi, from time to time.

DSE-A4/DSE: DATA MINING-II

CREDIT DISTRIBUTION, ELIGIBILITY AND PRE-REQUISITES OF THE COURSE

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course (if any)
		Lecture	Tutorial	Practical/ Practice		

9. SVM classification

10. K-Means Clustering

11. Hierarchical Clustering

DSC16/GE6e/DSE: ARTIFICIAL INTELLIGENCE

CREDIT DISTRIBUTION, ELIGIBILITY AND PRE-REQUISITES OF THE COURSE

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course
		Lecture	Tutorial	Practical/ Practice		
Artificial Intelligence	**4**	**3**	**0**	**1**	**Pass in Class XII**	**Programming using C++/Programming using Python/Object Oriented Programming using Python**

Course Objectives

The objectives of this course are to:

- To introduce basic concepts and techniques of Artificial Intelligence (AI).
- To apply informed search techniques for different applications.
- To learn various knowledge representation techniques and writing Prolog programs.
- To learn about the latest techniques for developing AI systems.

Learning outcomes

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

- identify problems that are amenable to solutions by specific AI methods.
- state the utility of different types of AI agents.
- apply different informed search techniques for solving problems.
- use knowledge representation techniques for AI systems.

SYLLABUS

Unit 1 **6 Hours**

Introduction: Introduction to artificial intelligence, background and applications, Turing test, Weak AI, Strong AI, Narrow AI, Artificial General Intelligence, Super AI, rational agent approaches to AI, introduction to intelligent agents, their structure, behavior and task environment.

Unit 2 **12 Hours**

Problem Solving and Searching Techniques: Problem characteristics, production systems, control strategies, breadth-first search, depth-first search, hill climbing and its variations, heuristics search techniques: best-first search, A* algorithm, constraint satisfaction problem, means-end analysis, introduction to game playing, min-max and alpha-beta pruning algorithms.

Unit 3 **16 Hours**

Knowledge Representation: Propositional logic, First-Order Predicate logic, resolution principle, unification, semantic nets, conceptual dependencies, frames, and scripts, production rules, Introduction to Programming in Logic (PROLOG).

Unit 4 **8 Hours**

Understanding Natural Languages: Components and steps of communication, the contrast between formal and natural languages in the context of grammar, Chomsky hierarchy of grammars, parsing, and semantics, Parsing Techniques, Context-Free and Transformational Grammars, Recursive transition nets.

Unit 5 **3 Hours**

AI The Present and the Future: Symbolic AI, Data-driven AI and Machine Learning, Introduction to Machine Learning and Deep Learning based AI, Interpretable and Explainable AI, Ethics of AI: benefits and risks of AI.

Essential/recommended readings

1. Russell, Stuart, J. and Norvig, Peter, *Artificial Intelligence - A Modern Approach*, Pearson, 4th edition, 2020..
2. Bratko, Ivan, *Prolog Programming for Artificial Intelligence*, Addison-Wesley, Pearson Education, 4th edition, 2012.
3. Patterson, DAN,W, *Introduction to A.I. and Expert Systems* – PHI, 2007.
4. Clocksin, W., F. and Mellish, *Programming in PROLOG*, 5th edition, Springer, 2003.

Additional references

1. Kaushik, Saroj, *Artificial Intelligence*, Cengage Learning India, 2011.
2. Rich, Elaine and Knight, Kelvin, *Artificial Intelligence*, 3rd edition, Tata McGraw Hill, 2010

Practical List :

Practical exercises such as

1. Write a program in Prolog to implement TowerOfHanoi(N) where N represents the number of disks.
2. Write a program to implement the Hill climbing search algorithm in Prolog.
3. Write a program to implement the Best first search algorithm in Prolog.
4. Write a program to implement A* search algorithm in Prolog.
5. Write a program to implement the min-max search algorithm in Prolog.
6. Write a program to solve the Water-Jug Problem in Prolog.
7. Implement sudoku problem (minimum 9×9 size) using constraint satisfaction in Prolog.
8. Write a Prolog program to implement the family tree and demonstrate the family relationship.
9. Write a Prolog program to implement knowledge representation using frames with appropriate examples.
10. Write a Prolog program to implement conc(L1, L2, L3) where L2 is the list to be appended with L1 to get the resulted list L3.
11. Write a Prolog program to implement reverse(L, R) where List L is original and List R is reversed list.
12. Write a Prolog program to generate a parse tree of a given sentence in English language assuming the grammar required for parsing.
13. Write a Prolog program to recognize context free grammar $a^n b^n$.

CREDIT DISTRIBUTION, ELIGIBILITY AND PRE-REQUISITES OF THE COURSE

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course (if any)
		Lecture	Tutorial	Practical/ Practice		
Android Programming using Java	**4**	**3**	**0**	**1**	Pass in Class XII	NIL

Learning Objective

The course enables the students to understand Android architecture and its key features, making them competent to develop Android applications using Java.

Learning outcomes

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

- logically organize Java classes and interfaces using packages.
- understand the design of the Android operating system.
- design user interfaces using various dialog boxes, menus, etc.
- design Android applications with interaction among various activities/applications.

SYLLABUS OF DSE 01b

Unit 1 (15 hours)
Review of Object Oriented Programming and Java Fundamentals: Structure of Java programs, classes and objects, data types, type casting, looping constructs, inheritance.

Unit 2 (2 hours)
Interfaces: Interface basics, defining, implementing and extending interfaces.

Unit 3 (4 hours)
Packages: Basics of packages, creating and accessing packages.

Unit 4 (7 hours)
GUI Programming: AWT classes, event handling.

Unit 5 (5 hours)

Introduction to Android Programming: Introduction to Android Operating System, Android SDK, AVD, components of an Android Application, parcels, and bundles.

Unit 6 (6 hours)

User Interface Architecture: Android Architecture, Contexts in Android, Intents and Intent Filters, Activity Life Cycle, Activity Stack, Fragments, and Fragments Life Cycle.

Unit 7 (6 hours)

User Interface Design: Android Layouts, Views, Spinner, Menu, Toggle Buttons, Radio Buttons, Check Boxes, Alert Box, and Toasts.

Essential/recommended readings

1. Schildt H. Java: The Complete Reference. 12th edition. McGraw-Hill Education, 2021
2. Griffiths D. & Griffiths D. Head First Android Development. O'Reilly, 2017
3. Meier R. Professional AndroidTM 4 Application Development. John Wiley & Sons, Inc., 2012

Additional Resources:

1. Horstmann, C. S. Core Java - Vol. I – Fundamentals. 12th edition. Pearson Education, 2021
2. Murphy M. L. The Busy Coder's Guide to Android Development. CommonsWare, 2018
3. Phillips B., Stewart C., Hardy B. & Marsicano K. Android Programming: The Big Nerd Ranch Guide. Big Nerd Ranch, LLC, 2015
4. Sheusi J. C. Android Application Development for Java Programmers. Cengage Learning, 2013

Suggested Practical List (If any): (30 Hours)

1. Write a function to find whether a number is prime or not. Use this function to determine the nth prime number. Read n from the user.

2. Design a class Complex having a real part (x) and an imaginary part (y). Provide methods to perform the following on complex numbers:

    a. Add two complex numbers.

b. Multiply two complex numbers.

c. toString() method to display complex numbers in the form: x + i y

3. 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.

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

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

6. 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.

7. Create a “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.

8. Create an Android application with a login module. (Check username and password).

9. Create a Spinner with strings taken from resource folder (res >> value folder) and on changing the spinner value, Image will change.

10. Create a Menu with 5 options and a selected option should appear in the text box.

11. Create an application with three option buttons, on selecting a button colour of the screen will change.

12. Create an Application to display various Activity and Fragment Life Cycle Methods.

13. Create an application with 2 fragments, one to set the background and other to set the fore-color of the text.

Note: Examination scheme and mode shall be as prescribed by the Examination Branch, University of Delhi, from time to time.

Revised Syllabus

DISCIPLINE SPECIFIC ELECTIVE COURSE : Cyber Security

Credit Contribution, Eligibility, Pre-requisites of the Course

Course title and Code	Credits	Credit distribution of the course			Eligibility Criteria	Pre-requisite of the course(if any)
		Lecture	Tutorial	Practical/ practice		
Cyber Security	4	3	0	1	Pass in Class XII	None

Learning Objectives

The course is designed to develop awareness and foundational knowledge in Cyber Security, enabling students to understand the basic architecture of cyber space and the techniques used to protect digital environments. It aims to equip students with the skills necessary to recognize threats, analyze vulnerabilities and implement basic protective measures.

Learning Outcomes

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

- Understand the fundamentals of cyber space and cybersecurity.
- Identify the vulnerabilities and potential security gaps across the various components of the digital ecosystem.
- Analyze and classify cyber threats like malware and intrusion attempts, using appropriate tools.
- Utilize cybersecurity tools and best practices to safeguard personal and institutional digital assets.

Unit 1 Introduction (12 Hours) Ref 1,2

Importance of Cyber Security, what security is and isn't, roles and responsibilities of cybersecurity professionals, types of hackers (black hats, white hats, and their types). Understanding threats and vulnerabilities, Advanced Persistent Threats(APT), humans as the point of vulnerability. Social engineering attacks (Phishing, Vishing and other non-email Phishing attacks).

Unit 2 Attack targets on the internet (9 Hours) ref 1

Working on the Internet, Attack methodology: Reconnaissance, Weaponization, Delivery, Exploitation and Installation, Command and control, Attack on objectives. Black Hats methodologies to find victims, how to hide from attackers

Unit 3 Phishing Tactics and Malware Infections (12 Hours) ref 1

Protection against Phishing, How Black Hats trick you with URLs, Typosquatting, Complex URLs and redirects, Modifying DNS records, Hoaxes, Why Black Hats love phishing and how to avoid phishing. Definition and types of Malware (Viruses, Worms, Trojans, Ransomware, Spyware and Adware), Rootkits and Bootkits, Polymorphic, Deployment and defense against Malware.

Unit 4 Password Thefts and Account Access Tricks (12 Hours) ref 1

Authentication, Types of Authentication, Multi-Factor Authentication, Authorization, Mandatory Access Control, Rule-Based Access Control, Role-Based Access Control, Attribute-Based Access Control, Discretionary Access Control, Auditing, Indicators of attack.

Essential/ Recommended Readings

1. How Cybersecurity Really Works. A Hands-on Guide for Total Beginners by Sam Grubb. No Starch Press, San Francisco, 2021.
2. Cybersecurity and Cyberwar: what everyone needs to know by P.W. Singer and Allan Friedman, Oxford University Press, United States of America, 2014.

Additional References:

1. Information Security Education & Awareness https://isea.gov.in/
2. Indian Computer Emergency Response Team https://www.cert-in.org.in/
3. Cyber Swachhta Kendra, https://www.csk.gov.in/
4. A Handbook for Preventing Computer Frauds and Cyber Crimes by Gaurav Gupta and Garima Gupta, Vilvam Publications Pvt Ltd., New Delhi, 2021.

Practical list:

1. Install and set up VMware on your machines. (4 hours)
2. Execute network commands: ipconfig, ifconfig, ping, tracert to find the IP address of your machine, to test if a machine is connected and is reachable and find the route to the destination. (2 hours)
3. Use Shodan tool to find vulnerabilities in the websites and detect their open ports. (4 hours)
4. Use VirusTotal to scan and analyse malicious files and URLs to detect malware. (4 hours)
5. Use MX toolbox to analyse phishing emails and email header analysis. (2 hours)

6. Do account protection settings in any operating system (Windows 10 /MacOS/Linux) (6 hours)
7. Use USBDeview to track USB usage on your machine for currently connected USB devices as well as previously connected devices. (2 hours)
8. Explore the following three government-backed portals for various purposes: (6 hours)
    a. https://cybercrime.gov.in - for reporting cybercrime
    b. https://stopncii.org/ - if someone is threatening to share intimate images
    c. https://sancharsaathi.gov.in/ - Block your lost/stolen mobile handset, know mobile connections in your name, **chakshu**—report suspected fraudulent communication, and know the genuineness of your mobile handset.

c. Compare density distribution for features age and passenger fare
d. Use a pair plot to show pairwise bivariate distribution

4. Using Titanic dataset, do the following

   a. Find total number of passengers with age less than 30
   b. Find total fare paid by passengers of first class
   c. Compare number of survivors of each passenger class

5. Download any dataset and do the following

   a. Count number of categorical and numeric features
   b. Remove one correlated attribute (if any)

   c. Display five-number summary of each attribute and show it visually

Project: Students are encouraged to work on a good dataset in consultation with their faculty and apply the concepts learned in the course.

Note: Examination scheme and mode shall be as prescribed by the Examination Branch, University of Delhi, from time to time.

DSC11/DSC05/GE3a: DATABASE MANAGEMENT SYSTEMS

Credit distribution, Eligibility and Prerequisites of the Course

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course (if any)
		Lecture	Tutorial	Practical/ Practice		
Database Management Systems	**4**	**3**	**0**	**1**	Pass in Class XII	NIL

Course Objectives

The course introduces the students to the fundamentals of database management system and its architecture. Emphasis is given on the popular relational database system including data

models and data manipulation. Students will learn about the importance of database structure and its designing using conceptual approach using Entity Relationship Model and formal approach using Normalization. The importance of file indexing and controlled execution of transactions will be taught. The course would give students hands-on practice of structured query language in a relational database management system and glimpse of basic database administration commands.

Learning outcomes

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

● Use database management system software to create and manipulate the database.

● Create conceptual data models using entity relationship diagrams for modeling real-life situations and designing the database schema.

● Use the concept of functional dependencies to remove redundancy and update anomalies.

● Apply normalization theory to get a normalized database scheme.

● Write queries using relational algebra, a procedural language.

Syllabus

Unit 1 **(5 hours)**

Introduction to Database: Purpose of database system, Characteristics of database approach, data models, database management system, database system architecture, three-schema architecture, components of DBMS, data independence, and file system approach vs database system approach.

Unit 2 **(7 hours)**

Entity Relationship Modeling: Conceptual data modeling - motivation, entities, entity types, attributes, relationships, relationship types, constraints on relationship, Entity Relationship diagram notation.

Unit 3 **(7 hours)**

Relational Data Model: Update anomalies, Relational Data Model - Concept of relations, schema-instance distinction, keys, relational integrity constraints, referential integrity and foreign keys, relational algebra operators and queries.

Unit 4 **(12 hours)**

Structured Query Language (SQL): Querying in SQL, DDL to create database and tables, table constraints, update database-update behaviors, DML, aggregation functions group by and having clauses, retrieve data from the database, generate and query views. Access and manipulate databases using ODBC. Basic Database administration SQL commands.

Unit 5 **(10 hours)**

Database Design: Mapping an Entity Relationship model to relational database, functional dependencies and Normal forms, 1NF, 2NF, 3NF and BCNF decompositions and desirable properties of them.

Unit 6 **(4 hours)**

Data Storage and Indexes: Need of file indexes, file organizations, index structures, single- and multi-level indexing, concurrent execution of transactions, ACID properties,.

Essential/recommended readings

1. Elmasri, R., Navathe, B. S. Fundamentals of Database Systems, 7th Edition, Pearson Education, 2015.

2. Krogh, J. W. MySQL Connector/Python Revealed: SQL and NoSQL Data Storage Using MySQL for Python Programmers, Apress, 2018.

3. Murach J. Murach's MySQL, 3rd edition, Pearson, 2019.

Additional References

1. Ramakrishnan, R., Gehrke J. Database Management Systems, 3rd Edition, McGraw Hill, 2014.

2. Silberschatz, A., Korth, H. F., Sudarshan S. Database System Concepts, 7th Edition, McGraw Hill, 2019.

3. Connolly, T. M., Begg, C. E. Database Systems: A Practical Approach to Design, Implementation, and Management, 6th edition, Pearson, 2019.

Practicals (30 hours)

Create and use the following student-course database schema for a college to answer the given queries using the standalone SQL editor.

Here, Rollno (ADMISSION) and SID (ADMISSION) are foreign keys. Note that course type may have two values viz. Fulltime and Parttime and a student may enroll in any number of courses

1. Retrieve names of students enrolled in any course.
2. Retrieve names of students enrolled in at least one part time course.
3. Retrieve students' names starting with letter 'A'.
4. Retrieve students' details studying in courses 'computer science' or 'chemistry'.
5. Retrieve students' names whose roll no either starts with 'X' or 'Z' and ends with '9'
6. Find course details with more than N students enrolled where N is to be input by the user.
7. Update student table for modifying a student name.
8. Find course names in which more than five students have enrolled
9. Find the name of youngest student enrolled in course 'BSc(P)CS'
10. Find the name of most popular society (on the basis of enrolled students)
11. Find the name of two popular part time courses (on the basis of enrolled students)
12. Find the student names who are admitted to full time courses only.
13. Find course names in which more than 30 students took admission
14. Find names of all students who took admission to any course and course names in which at least one student has enrolled
15. Find course names such that its teacher-in-charge has a name with 'Gupta' in it and the course is full time.
16. Find the course names in which the number of enrolled students is only 10% of its total seats.
17. Display the vacant seats for each course
18. Increment Total Seats of each course by 10%
19. Add enrollment fees paid ('yes'/'No') field in the enrollment table.
20. Update the date of admission for all the courses by 1 year.
21. Create a view to keep track of course names with the total number of students enrolled in it.
22. Count the number of courses with more than 5 students enrolled for each type of course.
23. Add column Mobile number in student table with default value '9999999999'
24. Find the total number of students whose age is > 18 years.
25. Find names of students who are born in 2001 and are admitted to at least one part time course.

Create and use the following student-society database schema for a college to answer the given (sample) queries using the standalone SQL editor.

II. Do the following database administration commands:

Create user, create role, grant privileges to a role, revoke privileges from a role, create index

22. Count the number of courses with more than 5 students enrolled for each type of course.
23. Add column Mobile number in student table with default value '9999999999'
24. Find the total number of students whose age is > 18 years.
25. Find names of students who are born in 2001 and are admitted to at least one part time course.
26. Count all courses having 'science' in the name and starting with the word 'BSc'.

Note: Examination scheme and mode shall be as prescribed by the Examination Branch, University of Delhi, from time to time.

GENERIC ELECTIVES : Java Programming

Credit distribution, Eligibility and Pre-requisites of the Course

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course	Department offering the course
		Lecture	Tutorial	Practical/ Practice			
GE: Java Programming	**4**	**3**	**0**	**1**	**Pass in class XII**	NIL	**Computer Science**

Learning Objectives

This course is designed to develop understanding of object-oriented programming concepts like Classes, Objects, Inheritance and Polymorphism using Java. The course provides understanding of multithreading and exception handling in Java. It also introduces how to create Java applications with graphical user interface (GUI).

Learning outcomes

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

- Understand the object-oriented concepts – Classes, Objects, Inheritance, Polymorphism– for problem solving.
- Create and handle multithreading.
- Handle program exceptions.
- Handle input/output through files.
- Create Java applications with a graphical user interface (GUI).

SYLLABUS OF GE

Unit 1 (6 hours)

Introductory Concepts: program, identifiers, variables, constants, primitive data types, expressions, Naming Conventions, Type casting, operators, control statements, structured data types, arrays, functions.

Unit 2 (13 hours)

Object Oriented Concepts: Abstraction, encapsulation, objects, classes, methods, constructors, inheritance, polymorphism, static and dynamic binding, Anonymous block, Static Data members, overloading and overriding, Usage of super and this keyword, Abstract classes, Interfaces and Packages, Access modifiers, Object class

Unit 3 (11 hours)

Multithreading: Creating Threads, Thread Priority, Blocked States, Extending Thread Class, Runnable Interface, Starting Threads, Thread Synchronization, Sync Code Block, Overriding Synced Methods, Thread Communication, wait, notify and notify all.

Unit 4 (8 hours)

Introduction to Exception handling: Exception and Error, Throw, try and catch Blocks, Exception handlers, java.lang Exceptions, Built-InExceptions.

Unit 5 (7 hours)

Introduction to File Handling: Byte Stream, Character Stream, File I/O Basics, File Operations, Serialization.

Essential/recommended readings

1. Cay S. Horstmann, *Core Java - Vol. I – Fundamentals*, 10th edition, Pearson, 2017.
2. James Gosling, Bill Joy, Guy L. Steele Jr, Gilad Bracha, Alex Buckley, *The Java Language Specification*, *Java SE* 7th edition, Addison-Wesley, 2011

Additional References

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

2 Richard Johnson, *An Introduction to Java Programming and Object-Oriented Application Development*, Thomson Learning, 2006.

3. Kathy Sierra and Bert Bates, *Head First Java*, 3rd edition, O'Reilly, 2022.

Suggested Practical List (If any): (30 Hours)

Practical exercises such as

1. Create a java program to implement stack and queue concepts.
2. Write a program to take input from command line arguments.
3. Write a java program to show static and dynamic polymorphism.
4. Write a java program to show multiple inheritance using interfaces.
5. Write a program in java to show the chaining of execution of construction.
6. Write a java program to show multithreaded producer and consumer applications.
7. write a program in java to synchronize the multithreaded application
8. Create a customized exception and also make use of all the exception keywords.
9. Write a program to show different ways to get input from user
10. Design a form using AWT components and the Frame container.

Note: Examination scheme and mode shall be as prescribed by the Examination Branch, University of Delhi, from time to time.