CS601AT: Data Warehousing and Data Mining

Objective: Capabilities of both generating and collecting data have been increasing rapidly in the last several decades due to the use of bar codes, computerizations of many products, advances of data collection tools etc. This growth in stored data has generated an urgent need of the subject like Data Mining. The paper aims to give the concept and various techniques of data mining to the students. Students will also learn the feasibility, usefulness, efficiency and the scalability of the techniques for discovery of patterns hidden in large databases.

Outline of the course

Minimum Class hours	Exam Time (Hours)	Marks		
		External	Internal	Total
100	3	75	25	100

Unit	Topic	Class Hours	Marks
I	Data Warehousing	20	18
II	Data Mining Introduction	15	10
III	Clustering Approaches	25	20
IV	Rule Mining	20	15
V	Classification, Prediction and Applications	20	12
Total		100	75

UNIT I — 20 Hours

Data Warehousing: Overview and Concepts: Need for Data Warehousing, Basic elements of Data Warehousing, differences between Database Systems and Data Warehouse.
Planning and Requirements: Project planning and management, collecting the requirements.
Architecture and Infrastructure: Data Warehouse Architecture and its components, Infrastructure and metadata.
Data Design and Data Representation: Principles of dimensional modeling, advanced topics- data extraction, transformation and loading, data quality.
Information Access and Delivery: Matching information to classes of users, OLAP in Data Warehouse, Data warehousing and the web.
Implementation and Maintenance: Physical design process, Data Warehouse deployment, growth and maintenance.

UNIT II — 15 Hours

Data Mining Introduction: Basics of data mining, Different definitions of Data Mining and related concepts, Data mining process- Data preparation, data cleaning and data visualization. KDD process. Data mining techniques: Clustering, Association rules and Decision trees.

UNIT III — 25 Hours

Cluster Analysis: What is cluster analysis? *Types of Data*: Interval-scaled variables, Binary variables, Categorical, Ordinal, Ratio-scaled, Variables of mixed types and vector objects.
Clustering: Partitional versus Hierarchical Clustering. Partitional clustering methods - k-means, k-medoids, PAM, CLARA, CLARANS. Hierarchical clustering methods - BIRCH, CURE. Density based clustering methods- DBSCAN. Categorical clustering - DBSCAN.

UNIT IV

20 Hours

Rule Mining: Market basket analysis; Frequent itemset, Closed itemsets. Mining association rules, frequent sets and border sets, algorithms for mining association rules - Apriori algorithm, Pincer-Search algorithm, Border algorithm. Generalized association rule, quantitative association rule, association rule with item constraint.

UNIT V

20 Hours

Classification and Prediction: Introduction, Classification by Decision Tree Induction, Tree construction principle, decision tree generation algorithms - CART, ID3.

Applications of Data Mining: Data mining for financial data analysis, Data mining for retail industry, Data mining for telecommunication industry, Data mining biological data analysis, Data mining for scientific applications, Data mining for intrusion-detection.

Instruction to paper setter

Unit	To be set	To be answered	Marks
I	3	2	18
II	2	1	10
III	3	2	20
IV	2	1	15
V	2	1	12
Total	12	7	75

Recommended Books

Text

1. A.K. Puzari, Data Mining Techniques, University Press.
2. J. Han and M. Kamber. Data Mining: Concepts and Techniques. Morgan Kaufman. 2010.

Reference:

1. P. Tan, M. Steinbach and V. Kumar; Introduction to Data Mining; Pearson Education (LPE); 2009.

CS601BT: Design and Analysis of Algorithms and Theory of Computation

Objective: The study of algorithms is at the heart of computer science and at the same time it is vast computer science domains. In recent years, a number of advances have been made in the field of designing of the algorithms. This paper is meant to give the students an in-depth knowledge to analyse and design a better algorithm before its actual implementation.
The objective of the theory of computing is to introduce and study abstract, mathematical models of computation (such as finite state, push down & Turing machines), and to use the abstract machines models to study the ability to solve computational problems. At the complete course students will be able to use regular expression effectively and appropriately, construct derivations and parse trees, understand the equivalence of grammars, languages & automata and translate between grammars, languages and automata.

Outline of the Course

Minimum Class Hours	Exam Time (Hours)	Marks		
		External	Internal	Total
100	3	75	25	100

Unit	Topic	Minimum Class Hours	Marks
I	Computational Models + Memory Management	12	10
II	DAA + Algorithm Analysis	22	15
III	Algorithm Design + External Storage	22	15
IV	Theory of Automata	20	15
V	Formal Languages + Regular Grammars	24	20
	Total	**100**	**75**

Unit I: **12 Hours**

Computational Models: RAM(only definition and characteristics), Turing Machine(only definition and characteristics), PRAM(only definition and characteristics),Programming Concepts.

Memory Management: The issues in memory management, Managing equal-sized blocks, Garbage collection algorithms for equal-sized blocks, Storage compaction.

Unit II **22 Hours**

Design and Analysis of Algorithms: The running time of a program, Calculating the running time of a program, Good programming practice.

Algorithm Analysis Techniques: Efficiency of algorithms, Analysis of recursive programs, Solving recurrence equations, A general solution for large class of recurrences.

Unit III **22 Hours**

Algorithm Design Techniques: Divide and conquer algorithms, Dynamic programming, Greedy algorithms, Backtracking, Local search algorithms.

Data Structures and Algorithms for External Storage: A model for External computation, External sorting, Storing information in files, External search trees.

Unit IV **20 hours**

Theory of Automata: Definition of an Automaton, Description of a Finite Automaton, Transition Systems, Properties of Transition Functions, Acceptability of a String by a Finite Automaton, Nondeterministic Finite State Machines, The Equivalence of DFA and NDFA, Mealy and Moore Models, Minimization of Finite Automata

Unit V **24 Hours**

Formal Languages, Regular Sets & Regular Grammars: Definition of Formal Languages, Chomsky Classification of Languages, Languages and Their Relation, Recursive and Recursively Enumerable Sets, Operations on Languages, Languages and Automata; Regular Expressions, Finite Automata and Regular Expressions, Pumping Lemma for Regular Sets, Application of Pumping Lemma, Regular Sets and Regular Grammars, Exercises.

Instructions for Paper Setter

Unit	Questions		Marks
	To Be Set	To Be Answered	
I	2	1	10
II	2	1	15
III	2	1	15
IV	2	1	15
V	2	1	20
Total	**10**	**5**	**75**

Recommended Books:

Text Books

Design And Analysis of Algorithms (Units I, II and III):

1. Aho, A. V.; J. E. Hopcroft; J. D. Ullman, Data Structures and Algorithms, Addison Wesley/Pearson, New Delhi, 2000.
2. S K Basu. Design and Analysis of Algorithms. PHI, India, 2005.

Further reading

1. Aho, A. V.; J. E. Hopcroft; J. D. Ullman, The Design and Analysis of Computer Algorithms, New Delhi: Addison Wesley, 2001.
2. Cormen, T. H.; C. E. Leiserson; R. L. Rivest; C. Stein, Introduction to Algorithms (Second Edition), New Delhi: Prentice-Hall India, 2004

3. Manbar, V., Introduction to Algorithms- A Creative Approach. New Delhi: Addison Wesley, 2000
4. Horwitz, E.; S. Sahani, Fundamentals of Computer Algorithms, Computer Science Press, 2000

Theory of Computation: (Units IV and V)

1. Mishra, K. L. P.; N. Chandrasekaran, Theory of Computer Science (Third Edition), New Delhi: PHI Learning Private Limited, 2011
2. Lewis, H. R.; C. H. Papadimitriou, Elements of the Theory of Computation (Second Edition), New Delhi: Prentice-Hall India, 2003
3. Hopcroft, H. E.; J. D. Ullman, Introduction to Automata Theory, Languages and Computation, New Delhi: Narosa Publications, 2001

Further readings:

1. Martin, J. C., Introduction to Languages and the Theory of Automata, New Delhi: Tata McGraw-Hill
2. Papadimitriou, C. H., Computation Complexity, New Delhi: Addison Wesley

CS601CT and CS601CP: Object Oriented Programming through Java

Objective

The course is designed to impart knowledge and skill required to solve the real world problem using object-oriented approach utilizing Java language constructs. This course covers the two main part of Java i.e. Java Language and Java Library

After completion of the course students are expected to understand the following-

- Java tokens for creating expressions and creating Datatypes.
- The way various expressions and data types are assembled in packages.
- Implementation of Inheritance, Exception handling and Multithreading in Java.
- Java I/O basics and Applets.
- Setting up GUI using AWT/Swing.
- Network Programming in Java.
- Accessing relational databases from Java program.
- Java Servlets.

Outline of the Course

Minimum Class Hours			Exam time (Hours)		Marks				
					Theory		Practical		
Theory	Practical	Total	Theory	Practical	External	Internal	External	Internal	Total
50	50	100	2	3	45	15	30	10	100

Unit	Topic	Minimum Class Hours			Marks
		Theory	Practical	Total	Theory
I	Introduction to Java Programming, Classes and Methods	10	10	20	09
II	Inheritance, Exception handling, Multithreading Enumerations and Autoboxing	10	10	20	09
III	java.util, String handling, java.lang and Input/Output	10	10	20	09
IV	Applet class & Handling events	10	10	20	09
V	Networking, JDBC, Java Servlets	10	10	20	09
	Total	**50**	**50**	**100**	**45**

Detailed Syllabus

Unit I: Introduction to Java Programming, Classes and Methods **10+10 Hours**

Introduction to Java: Genesis and Overview, Java & Internet, Object-Oriented Programming features (Abstraction, Encapsulation, Inheritance and Polymorphism); Difference between (Java Script and Java, Java and C++, Java applet and Application), Java Development Kit (JDK) Java Virtual Machine (JVM), The Bytecodes, Compile & run a simple program

Constant, Variable, Data types & Arrays: Java Token & Keywords, Primitive Types, Integer literal, Floating point literal, Character literal, Boolean literal, String literal, declaring a variable, Dynamic initialization, The scope and lifetime of variable, Type conversion and casting, Automatic type promotion in expression, Arrays (One-dimension, Multidimension), Alternative array declaration syntax

Operators: Arithmetic operators, Bitwise operators, Relational operators, Boolean logical operators, The assignment operator, Conditional operator, Operator precedence
Control statement: Decision making and Branching (*if*, Nested *if*, *if-else-if* ladder, *switch*, Nested *switch*, The?: operator), Decision making and Looping (*while*, *do-while*, *for*) , Jump (*break*, *continue* and *return*)
Introduction to classes, methods and objects: The general form of a class, declaring objects, Assigning object reference variable, Introducing methods (Adding methods to a class, returning a value, Adding methods that takes parameters), Constructors, Parameterized constructor, The *this* keyword, Instance variable hiding, Garbage collection, the finalize() method, A stack class-an example, Overloading(methods, constructors), Using object as parameters, Argument passing, Returning objects, Recursion, Introducing Access control (public, private and protected), *static*, *final*, nested and inner classes, String class, Command-line argument, Variable-Length arguments, Scanner (Constructors, Basics, setting delimiters)

Unit II: Inheritance, Exception handling, Multithreading, Enumerations and Autoboxing **10+10 Hours**

Inheritance: Extending a class, Basics of Inheritance, Member access and inheritance, using **super**, creating a multilevel hierarchy, when constructors are called, method overriding, dynamic method dispatch, using abstract classes, using *final* with inheritance, the Object class

Packages and Interface: Packages (Defining a package, Finding packages and classpath, Access protection and importing packages), Interfaces (Defining, implementing and Applying Interfaces, Variables in interface, Interfaces can be extended)
Exception handling: Exception handling fundamentals, Exception types (uncaught exceptions, using *try* and *catch*) Nested try statement, multiple *catch* clauses, throw, *throws* and *finally*, Java's built-in exceptions, user defined throwable, user defined exception subclasses, using Exception
Multithreaded Programming: The Java thread model (thread priorities, synchronization and inter-thread communication); The main thread, Creating a thread. isAlive( ), join( ), suspend( ) and resume( ), Deadlock
Enumerations, Type Wrappers and Autoboxing: Enumeration fundamentals, Java Enumerations are Class Types, Enumerations Inherit Enum, Type Wrappers, autoboxing and methods, autoboxing/unboxing occurs in expressions, autoboxing/unboxing Boolean and Character values

Unit III: java.util, String handling, java.lang and Input/Output **10+10 Hours**

java.util package – The Collection Framework – Collection Overview, Collection Interfaces (*Collection, List, Set, SortedSet),* Accessing Collection via an Iterator, Using a Comparator, Arrays, Legacy Classes and Interfaces (*Enumeration, Vector, Stack, Dictionary, Hashtable, Properties)*
String handling: The string constructor, Special string operations, Character extraction, String searching & comparison, Data conversion using valueOf (), StringBuffer
Exploring java.lang: Math functions (transcendental, exponential, rounding)
Input/Output-: Streams (Byte Streams and Character streams- class InputStream, OutputStream, Reader, Writer, Predefined streams, InputStreamReader, BufferedReader, Reading console input, writing console output, Reading and writing files, File, FileNameFilter & Directories, FileInputStream, FileOutputStream, PrintStream, FileReader, FileWriter, Serialization (Serializable, Externalizable), ObjectOutputStream, ObjectInputStream

Unit IV: Applet class and Handling Events **10+10 Hours**

The applet class: Applet fundamentals, The applet class, Applet architecture, Applet skeleton (initialization and termination, overriding update()),Applet Display Methods, Requesting repainting, Using the Status Window, HTML applet tag, Passing parameters to applets

Handling events:, The Delegation Event Model, Event Classes, Sources of events, Event Listener Interfaces, Processing mouse events, Handling keyboard events, Adapter classes, Anonymous Inner classes

Unit V: Networking, JDBC, Java Servlets **10+10 Hours**

Networking: Networking basics, InetAddress, Factory methods, URL, URLConnection, HttpURLConnection, Establishing a simple server using Stream Sockets, Establishing a simple client using Stream Sockets, Connectionless Client/Server Interaction with Datagrams

Java database connectivity (JDBC): Introduction to JDBC, type of JDBC connectivity, Accessing relational database from Java programs, Establishing database connections

Java Servlets: Background, The life cycle of a Servlet, The javax.servlet.http package (HttpServletRequest, HttpServletResponse, HttpSession Interfaces, Cookie class, HttpServlet class, HttpSessionEvent class), Handling Http requests and Response (HTTP GET & HTTP POST), Using Cookies, Session Tracking

Recommended Books

1. **Herbert Schildt,** *The Complete Reference Java* , Tata McGraw Hill Eighth Edition 2011.

References:

1. **Deitel Deitel,** *Java How to program*, Prentice Hall India Ltd Ninth Edition, 2012.
2. **Kathy Sierra & Bert Bates,** *Head First Java*, O'Reilly Second Edition.
3. **E Balagurusamy,** *Programming with Java A Primer*, Tata McGraw Hill Fourth Edition 2006.

Theory Questions				Practical Questions		
Unit	To be set	To be Answered	Marks	To be set	To be Answered	Marks
I	2	1	09	2	1	10
II	2	1	09			
III	2	1	09	3	2	20
IV	2	1	09			
V	2	1	09			
Total	**10**	**5**	**45**	**5**	**3**	**30**

Practical Assignments
(Questions need not be restricted to this list)

1. Write a program to create a class called Box with a parameterized constructor, along with a method to calculate the volume of the box. Use the class to find the volume of two boxes whose height, width and depth are 10,20,30 and 20,30,40 respectively.
2. Define a class called stack that can hold 10 integer values, then initialize top of the stack, with push and pop methods. Write a program to push the elements into the stack and pop out from the stack.
3. Write a Java program using a class to multiply two matrixes of 3*3 order. Allow the user to input the values through the keyboard.
4. Write a program to multiply two numbers using a method in a class and pass the values using call by value (pass by value and pass by reference) techniques.
5. Write a Java program to find factorial of positive integer using recursion.
6. Write a Java program to accept the command line arguments and display the arguments along with the positions.

7. Write a Java program to demonstrate method overriding where the program creates a superclass called figure that stores the dimensions of various two-dimensional objects. It also defines a method called area() that computes the area of an object. The program derives two subclasses from figure. The first is Rectangle and the second is Triangle. Each of these subclass overrides area() so that it returns the area of a rectangle and a triangle respectively.
8. Write a Java program to create a thread and start running it using runnable interface. Allow the thread to display a message five times with a gap of 500ms.
9. Write a Java program to demonstrate the synchronization of two threads using the synchronized statement.
10. Write a Java program to demonstrate interthread communication considering the producer and consumer problem. There must be two classes one for producer to produce data and another is consumer to consume data [Hint: Use wait() and nothing() to signal in both directions].
11. Write a Java program to copy the content of one file to another using java.io
12. Write an applet program to accept a message from the keyboard and then to display it on the console.
13. Write an applet to find the biggest of three numbers from the keyboard and display it on the console.
14. Design a Calculator System using Java, The applet should have all the digit buttons along with buttons for operations +,-,*,/ and =. There is a designated panel to show the current results. If a digital button is clicked, the number is displayed on the panel. If an operator button is clicked the operation is to be performed. You may assume the expression to be infix. The calculator can operate in two modes
    i. When the operator buttons are pressed the intermediate results should be displayed
    ii. The operations can take in any number of arguments and the final result is displayed only when the = button is pressed. [Hint: Use Overloading]
15. Write a program to input integers into an array and sort them using methods. Display the sorted numbers.
16. Write a program to copy the contents of one file to another file using command line arguments. Give appropriate error messages if any I/O error occurs.
17. Write a socket based Java application program to create a connection between two machines such that whatever text one machine is sending to the other will be displayed at the latter's screen and vice-versa
18. Create a Java application in which a particular machine is configured as the time server which continually listens for requests for time from clients. Clients request the server for time as a result of which the server sends the current time of the clients. The clients make a correction of the received time by adding a very small positive constant to the value and display the corrected time.
19. Create an editor applet in Java using which the users can enter some text and set the font and color of the text according to their choice. The text will be displayed appropriately when the applet is run.
20. Develop an applet to display four push buttons and a text field. Each button displays an icon that represents the flag of a country. When a button is pressed, the name of the country is displayed in the text field. The applet begins by getting its context page and setting the layout manager of that pane. Next, the applet is registered to receive action events that are generated by the buttons. A text filed is then created and added to the applet. Finally, a handle for action event displays the command string that is associated with the button. The text field is used to present this string.
21. Create an applet that displays four check boxes and a text field. When a check box pressed. Its text is displayed in the text field. The context pane for the JApplet object is obtained, and a flow layout is assigned as its layout manager. Next, four check boxes are added to the

context pane, and icons are assigned for the normal, roll over and selected status. The applet is then registered to receive item events. Finally, a text field is added to the context pane.

22. Create an applet that displays four radio buttons and one text field. When a radio button is selected, its text should be displayed in the text field.
23. Develop a servlet allowing you to read the names and values of parameters that are included in a client request using ServletRequest class. Develop the web page corresponding to the servlet.
24. Develop a servlet that handles an HTTP GET request. The servlet is involved when a form on a web page is submitted. The HTML web page defines a form that contains a select elements and a submit buttons. The select element name is color and the options are Red, Green and Blue. The servlet responses according to the option submitted and display the message "you have selected color".
25. Develop a servlet that handles an HTTP POST request. The servlet is involved when a form on a web page is submitted. [Hint: The HTML source code is same as the above problem. Except that the method parameter for the form tag explicitly specifies that the POST method should be used and the action parameter for the form tag specifies a different servlet].
26. Write a Java program that prints the addresses and names of the local machine and two well known explored Internet web sites.
27. Implement a simple networked communications clients and server. Message are typed into the window at the server and written across the network to the client side, then they are displayed to demonstrate datagrams.

CS602T: Software Engineering

Objective: Software Engineering is a fast developing field. We can view Software Engineering as the engineering approach to developing software. The objective of this paper is to provide a broad understanding of system development concepts. It provides the students with a sense of confidence to develop new systems.

Outline of the Course

Minimum Class Hours	Exam Time (Hours)	Marks		
		External	Internal	Total
50	2	45	15	60

Unit	Topic	Minimum Class Hours	Marks
I	System Analysis and Design	10	09
II	Introduction To Software Engineering Software life cycle models	8	08
III	Software project management, requirements and design	14	10
IV	Function Oriented Software Design, and User Interface Design	10	09
V	Testing, software reliability and maintenance	8	09
TOTAL		50	45

UNIT I **10 hours**

Introduction: System definition and concepts: Characteristics and types of system, Manual and automated systems

Systems models: Types of models, Systems environment and boundaries

Systems analyst: Role and need of systems analyst ,Qualifications and responsibilities, Change agent, Investigator and monitor, Architect, Psychologist, Salesperson, Motivator, politician, The analyst /User interface-behavioural issues, conflict resolution, MIS organization

System Planning: Data and fact gathering techniques: Interviews, Group communication, Presentations, Site visits.

Feasibility study and its importance: Types of feasibility reports, System, Selection plan and proposal

UNIT II **8 hours**

Introduction: - Evolution of an art to an engineering disciple, Solution to the software crisis, Computer systems engineering.

Software Life cycle models: - Importance of a life cycle model, waterfall model (feasibility study, requirement analysis and specification, design, coding and unit testing, integration and system testing, maintenance), prototyping model, evolutionary model, spiral model, Comparison of different life cycle models.

UNIT III **14 hours**

Software project management: Responsibilities of a Software Project Manager, Project Planning, Project Estimation Techniques (only Basic COCOMO), Scheduling (work breakdown, Activity Networks and Critical Path Method, Gantt Charts, PERT Charts, Project Monitoring and Control), Organization and Team Structures (Organization Structure, Team Structure), Risk management (Definition), Software Configuration Management (Definition).

Requirement Analysis and specification: Requirement gathering and analysis, Software Requirements Specification (Content of the SRS document, characteristics of a good SRS document, techniques for representing complex logic – Decision Tree and Decision Table).
Software Design: Characteristics of a good software design, cohesion and coupling (classification of cohesiveness and coupling), Software designs approaches (function-oriented design, Object-oriented Design).

UNIT IV **10 hours**

Function Oriented Software Design: Overview of SA/SD methodology, Structured Analysis, data Flow Diagrams (DFDs)(primitive symbols used for constructing DFDs, important concepts associated with designing DFDs, developing the DFD Model of a system, Shortcomings of the DFD Model), Structured Design (flow chart vs. structure chart, transformation of a DFD model into a structure chart).
User Interface Design: characteristics of a good user interface, basic concepts (user guidance and online help, mode-based vs. Modeless Interface, Graphical User Interface (GUI) vs. Text-based User Interface), Types of user interfaces (command language-based Interface, Menu-based Interface, direct manipulation Interface), Component-Based GUI Development (Window system, Types of widgets, Visual programming)

UNIT V **8 hours**

Coding and Testing: coding standards and guidelines, code review (code walkthroughs, code inspection), Software Documentation(Internal and External), Testing (testing, verification vs. validation, design of test cases), Testing in the large, Testing in the small, unit testing, Black-box testing, White-box testing (Statement, Branch and Condition coverage), debugging, integration testing, system testing.
Software Reliability: Software reliability, software quality, and software quality management.
Software Maintenance: Characteristics of software maintenance (types of software maintenance, special problems associated with software maintenance), software reverse engineering.

Instruction to Paper Setter

Unit	Questions		Marks
	To be Set	To be Answered	
I	2	1	09
II	2	1	08
III	2	1	10
IV	2	1	09
V	2	1	09
Total	**10**	**5**	**45**

Recommended Books

Text:
1. **Elias M. Awad,** ***System Analysis and Design,*** Galgotia Publications (P) Ltd, New Delhi
2. **Rajib Mall,** ***Fundamentals of Software Engineering,*** Pearson Education/Prentice Hall of India, New Delhi.

References:
1. **Carlo Ghezzi, Mehdi Jazayeri, Dino Mandrioli,** ***Fundamentals Of Software Engineering,*** Second Edition, Prentice Hall of India Private Limited, New Delhi, 2002.
2. **Richard E Fairley,** ***Software Engineering Concepts,*** Tata McGraw Hill Publishing Company Limited, New Delhi, 1997.

CS602P: Project

Objective

The objective of the project is to consolidate the concepts and practices that were learned during the course and to serve as a record of competence. It should enable a student to apply concretely in a small package the concepts gained from Software Engineering.

Outline of the Course

Minimum Hours	Marks		
	External	Internal	Total
80	30	10	40

Guidelines

- ***Overview***: The project will be carried out over a duration of three months, involving minimum 100 hours for General students and minimum 180 hours for Honours students. Every student should do a project individually and not in a group. The selected project can be either of type Model 1 or Model 2 described below.
- ***Platform***: The project can be in any platform e.g., DOS, WINDOWS, UNIX, LINUX, Mac OS, etc.
- ***Language and package***: The project can be done using any language or package learned within or outside the course such as C, C++, Java, VB, C#, Director, tcl, VC++, Visual FoxPro, Flash, etc.
- ***Venue***: The project can be done in the College itself or in a reputed organization.
- ***Guides***: Internal Guides from within the college should be assigned to each student. If the project is to be done in a reputed organization, an External Guide from that organization is also required as Co-Guide, and the qualification of the External Guide should not be less than that of the Internal Guide.
- ***Monitoring of Projects***: The progress of the project should be monitored through seminars, and each of the seminars should be evaluated, a record of which should be maintained. Every student will have to maintain a log book where the coding of the project is kept. This will have to be periodically signed by the internal guide. The number of seminars should not be less than three (e.g. Analysis, Design, Implementation).
- ***Final Examination***: For the final external evaluation a brief summary of the project should be submitted to the university at least one week prior to the date of the examination for the benefit of the external examiner(s).

Types of Project

Model 1

1. The topic for the project can be any subsystem of a system software or tool or any scientific or a fairly complex algorithmic situation.
2. The aim of this type is to highlight the abilities of algorithmic formulation, program and data flow representation, modular programming or object oriented programming, optimized code preparation, systematic documentation and other associated aspects of software engineering.
3. The assessment would be through the Project Report, Viva and the following criteria for this model:
   - Programming style, structured design, minimum coupling and high cohesion, abstraction, encapsulation, inheritance and polymorphism, as relevant.
   - Good commenting and annotating of the code and flow of representation, such that meaningful code, with good readability and ease of maintenance, results.
   - Design specifications, depicting the method adopted and giving a simple data dictionary for each data, to cover name, type and validity aspects.

» Test case samples, enough in number, to adequately cover the possible chances of common errors

Model 2

1. This model can be of a typical business application. The aim of this type is to highlight the stages involved in a typical business oriented project development, though on a miniature scale, in a real or simulated environment. The appropriate use of DBMS/RDBMS towards any business application, along with adequate system analysis and structured design and development of specific tools/products, would be the underlying activity in preparing this project.
2. The emphasis should be on selecting a system/subsystem that shows the DBMS/ RDBMS and System Analysis aspects to a greater degree. Any small and simple business system may be selected, although candidates are advised to use their knowledge and creativity, to select typical and intelligent applications, rather than run-of-the-mill themes, such as simple Pay roll calculation or Issue-Return portion of an inventory scheme. The Evaluation stage would give due weightage for theme selection, problem analysis, fact finding techniques and initial design, which is as close to real-life business situations as possible.
3. The code can be generated out of 4 GL Interface, like Screen Builder and Report Generator, Application Generator/Program Code Generators, or can be totally hand-coded or a combination of both. The documentation need not contain the code generated by these applications, but only that written by the candidate.
4. The assessment would be through the Project Report, Viva and the following criteria for this model:
   - » Requirements leading to the project, those which were the result of System Analysis
   - » The design aspect of DBMS/RDBMS oriented documentation which describes the structure and organization of the database, well annotated source code, supplemental documentation, which can serve as Data Analysis and Data Flow description
   - » A simple Data Dictionary of the elements which form the structure
   - » Details about I/O Screens and facilities for on-screen querying, print oriented Reports and built in house-keeping routines which help disk management and file integrity, are to be included to the extent possible.
   - » Details of Acceptance Tests which, should be in adequate number and should include error messages

Content of the Project Report

1. Acknowledgement
2. Certificate, stating it to be a bonafide work of the student, and that it has not been submitted for any other examination, and counter-signed by the project guide(s).
3. Synopsis of the project
4. Description of the existing system
5. Proposed system
6. User requirements
7. Hardware and software requirements
8. Costs and benefits estimation
9. Gantt Chart (Project Control)
10. System Flow Charts, Algorithms
11. DFD, Decision Tables, Decision Trees
12. Data Dictionary
13. Module Design
14. Database Design
15. File Description

16. Source Code
17. Input and Output Screen Design
18. Testing used and Test Results
19. Need for review : deficiencies and future enhancements
20. User/Operational Manual (including menu design, security aspects, access rights, backup, controls etc.)

Data Dictionary

1. This should give a catalogue of the data elements used in the system/subsystem developed.
2. The following are the details required. Write NA where NOT applicable
   - » Data Name
   - » Aliases, if any
   - » Length (size)
   - » Type (Numeric, alpha, binary, etc)
   - » Validity criterion (Minimum, maximum, etc)
   - » Default value, if any
   - » Whether related to other data items
   - » Where used in the program: Reference to data structure/file/procedures/modules

User Manual

It may include chapters like the ones suggested below:

- » Installation
- » Hardware requirements
- » System requirements
- » Installation procedure, including security aspects like password, protection, backups, controls, etc
- » Menu choices and their actions - screen formats
- » Error messages
- » Output
- » A Sample test case

Viva-Voce

The viva-voce will be conducted by external examiner(s) appointed by the University and internal examiner(s) from the College. Other members of the faculty and students may be present. It will be of duration of about 15 to 20 minutes. The analysis, design aspects and quality of implementation of the project would be the main subject matter for the viva. However the general proficiency of the candidate in the selected software platform should also be tested.

Distribution of Marks

Sl. No	Criteria	Marks
1.	Analysis	06
2.	Design	06
3.	Implementation	06
4.	Project Report	06
5.	Viva	06
6.	Internal Assessment	10
	Total	**40**