B. Tech Computer Science and Engineering (Artificial Intelligence and Data Science)
Scheme of Studies/Examination
Semester III

S. No.	**Course No.**	**Subject**	**L:T:P**	**Hours/ Week**	**Credits**	**Examination Schedule**				**Duration of Exam (Hrs.)**
						Major Test	**Minor Test**	**Practical**	**Total**	
1	BS-CS-AIDS-201A	Mathematics for Big Data & Optimization	3:0:0	3	3	75	25	0	100	3
2	PC-CS-AIDS-203A	Object Oriented Programming	3:0:0	3	3	75	25	0	100	3
3	PC-CS-AIDS-205A	Data Structures & Algorithms	3:0:0	3	3	75	25	0	100	3
4	PC-CS-AIDS-207A	Introduction to Artificial Intelligence	3:0:0	3	3	75	25	0	100	3
5	PC-CS-AIDS-209A	Programming Language	3:0:0	3	3	75	25	0	100	3
6	HM-902	Business Intelligence and Entrepreneurship	3:0:0	3	3	75	25	0	100	3
7	PC-CS-AIDS-213LA	Data Structures & Algorithms Lab	0:0:2	2	1	0	40	60	100	3
8	PC-CS-AIDS-215LA	Artificial Intelligence Lab	0:0:2	2	1	0	40	60	100	3
9	PC-CS-AIDS-217LA	Object Oriented Programming Lab	0:0:2	2	1	0	40	60	100	3
		Total		**24**	**21**	**450**	**270**	**180**	**900**	
11	SIM-201A*	Seminar on Summer Internship	2:0:0	2	0	0	50	0	50	

***Note: SIM-201A* is a mandatory credit-less course in which the students will be evaluated for the Summer Internship (training) undergone after 2nd semester and students will be required to get passing marks to qualify.**

BS-CS-AIDS-201A	**Mathematics for Big Data & Optimization**						
Lecture	**Tutorial**	**Practical**	**Credit**	**Major Test**	**Minor Test**	**Total**	**Time**
3	**0**	**0**	**3**	**75**	**25**	**100**	**3 Hour**
Purpose	To introduce the concepts of mathematics in Data Science.						
Course Outcomes (CO)							
CO1	Demonstrate understanding of basic mathematical concepts in data science, relating to linear algebra, probability, and calculus						
CO2	Employ methods related to these concepts in a variety of data science applications						
CO3	Apply logical thinking to problem-solving in context.						
CO4	Use appropriate technology to aid problem-solving and data analysis						

UNIT-I

Fourier series: Introduction, Fourier-Euler Formula, Dirichlet's conditions, Change of intervals, Fourier series for even and odd functions, Half range sine and cosine series.

Fourier Transform: Fourier Integral theorem, Fourier sine and cosine transforms and its properties, Convolution, Parseval's identity for fourier transforms, Fourier Transform of derivative of a function.

UNIT-II

First order ordinary differential equations: Exact, linear and Bernoulli's equations, Euler's equations, Equations not of first degree: equations solvable for p, equations solvable for y, equations solvable for x and Clairaut's type.

Differential equations of higher orders: Second order linear differential equations with constant coefficients, method of variation of parameters, Cauchy and Legendre's linear differential equations.

UNIT-III

Solution of polynomial and transcendental equations: Bisection method, Newton-Raphson method and Regula-Falsi method, Finite differences, Interpolation using Newton's forward and backward difference formulae. Interpolation with unequal intervals: Lagrange's formulae. Numerical differentiation using forward and backward difference.
Numerical Integration: Trapezoidal rule and Simpson's 1/3rd and 3/8 rules, Ordinary differential equations: Euler and modified Euler's methods. Runge-Kutta method of fourth order for solving first order equations.

UNIT-IV

Introduction – Formulation and classification of optimization problems, overview of analytical solution of unconstrained optimization problems, constrained optimization, convex set, convex functions, convex optimization problem, Kuhn Tucker condition, Search methods: overview of single variable search methods, search methods for multivariable unconstrained problems: optimality criteria, unidirectional search-direct search methods-evolutionary search methods.

Suggested Books:

- Veerarajan T., Engineering Mathematics, Tata McGraw-Hill, New Delhi, 2008.
- P. Kandasamy, K. Thilagavathy, K. Gunavathi, Numerical Methods, S. Chand & Company, 2nd Edition, Reprint 2012.
- S.S. Sastry, Introductory methods of numerical analysis, PHI, 4th Edition, 2005.
- Ramana B.V., Higher Engineering Mathematics, Tata McGraw Hill New Delhi, 11th Reprint, 2010.
- Erwin Kreyszig, Advanced Engineering Mathematics, 9th Edition, John Wiley & Sons, 2006.
- N.P. Bali and Manish Goyal, A text book of Engineering Mathematics, Laxmi Publications, Reprint, 2010.

- B.S. Grewal, Higher Engineering Mathematics, Khanna Publishers, 35th Edition, 2000.
- W. E. Boyce and R. C. DiPrima, Elementary Differential Equations and Boundary Value Problems, 9th Edition, Wiley India, 2009.
- E. A. Coddington, An Introduction to Ordinary Differential Equations, Prentice Hall India, 1995.
- E. L. Ince, Ordinary Differential Equations, Dover Publications, 1958.
- G.F. Simmons and S.G. Krantz, Differential Equations, Tata McGraw Hill, 2007.
- Optimization Methods for Engineers, N.V.S. Raju, PHI Learning Pvt. Ltd.

PC-CS-AIDS- 203A	**Object-Oriented Programming**						
Lecture	**Tutorial**	**Practical**	**Credit**	**Major Test**	**Minor Test**	**Total**	**Time**
3	**0**	**0**	**3**	**75**	**25**	**100**	**3 Hour**
Purpose	To introduce the principles and paradigms of Object Oriented Programming Language for design and implement the Object Oriented System.						
Course Outcomes (CO)							
CO1	To introduce the basic concepts of object oriented programming language and the its representation.						
CO2	To allocate dynamic memory, access private members of class and the behavior of inheritance and its implementation.						
CO3	To introduce polymorphism, interface design and overloading of operator.						
CO4	To handle backup system using file, general purpose template and handling of raised exception during programming.						

UNIT – I

Introduction to C++, C++ Standard Library, Illustrative Simple C++ Programs. Header Files, Namespaces, Application of object oriented programming.

Object Oriented Concepts, Introduction to Objects and Object Oriented Programming, Encapsulation, Polymorphism, Overloading, Inheritance, Abstract Classes, Accessifier (public/ protected/ private), Class Scope and Accessing Class Members, Controlling Access Function, Constant, Class Member, Structure and Class

UNIT – II

Friend Function and Friend Classes, This Pointer, Dynamic Memory Allocation and Deallocation (New and Delete), Static Class Members, Constructors, parameter Constructors and Copy Constructors, Deconstructors,

Introduction of inheritance, Types of Inheritance, Overriding Base Class Members in a Derived Class, Public, Protected and Private Inheritance, Effect of Constructors and Deconstructors of Base Class in Derived Classes.

UNIT – III

Polymorphism, Pointer to Derived class, Virtual Functions, Pure Virtual Function, Abstract Base Classes, Static and Dynamic Binding, Virtual Deconstructors.

Fundamentals of Operator Overloading, Rules for Operators Overloading, Implementation of Operator Overloading Like <<,>> Unary Operators, Binary Operators.

UNIT – IV

Text Streams and binary stream, Sequential and Random Access File, Stream Input/ Output Classes, Stream Manipulators.

Basics of C++ Exception Handling, Try, Throw, Catch, multiple catch, Re-throwing an Exception, Exception specifications.

Templates: Function Templates, Overloading Template Functions, Class Template, Class Templates and Non- Type Template arguments.

Suggested Books:

1. The complete reference C ++ by Herbert shieldt Tata McGraw Hill.
2. Object Oriented Programming in Turbo C++ by Robert Lafore, 1994, The WAITE Group Press.
3. Shukla, Object Oriented Programming in c++, Wiley India.
4. C++ How to Program by H M Deitel and P J Deitel, 1998, Prentice Hall.
5. Programming with C++ By D Ravichandran, 2003, T.M.H.

Note: The Examiner will be given the question paper template and will have to set the question paper according to the template provided along with the syllabus.

PC-CS-AIDS-205A	**Data Structures & Algorithms**						
Lecture	**Tutorial**	**Practical**	**Credit**	**Major Test**	**Minor Test**	**Total**	**Time**
3	**0**	**0**	**3**	**75**	**25**	**100**	**3 Hour**
Purpose	To introduce the principles and paradigms of Data Structures for design and implement the software systems logically and physically.						
Course Outcomes (CO)							
CO 1	To introduce the basic concepts of Data structure , basic data types ,searching and sorting based on array data types.						
CO 2	To introduce the structured data types like Stacks and Queue and its basic operations' implementation.						
CO 3	To introduce dynamic implementation of linked list.						
CO 4	To introduce the concepts of Tree and graph and implementation of traversal algorithms.						

UNIT – I

Introduction to Data Structures, Data Types, Built in and User Defined Data Structures, Applications of Data Structure, Algorithm Analysis, Worst, Best and Average Case Analysis, Notations of Space and Time Complexity, Basics of Recursion.

Arrays, One Dimensional Arrays, Two Dimensional Arrays and Multi-Dimensional Arrays, Sparse Matrices, Searching from array using Linear and Binary Searching Algorithm, Sorting of array using Selection, Insertion, Bubble, Radix Algorithm

UNIT – II

Stacks: Definition, Implementation of Stacks and Its Operations, Evaluation of Infix, prefix and Postfix Expression, Inter-conversion of Infix, Prefix and Post-Fix Expression, Implementation of Merge Sort and Quick Sort Algorithm.

Queues: Definition, Sequential Implementation of Linear Queues and Its Operations, Circular Queue and Its Implementation, Priority Queues and Its Implementation, Applications of queues.

UNIT – III

Linked Lists: Need of Dynamic Data Structures, Single Link List and Its Dynamic Implementation, Traversing, Insertion, Deletion Operations on Single Link Lists. Comparison between Static and Dynamic, Implementation of Linked List. Circular Link Lists and Doubly Link List, Dynamic Implementation of Primitive Operations on Doubly Linked Lists and Circular Link List. Dynamic Implementation of Stacks and Queues.

UNIT – IV

Trees: Definition, Basic Terminology, Binary Tree, External and Internal Nodes, Static and Dynamic Implementation of a Binary Tree, Primitive Operations on Binary Trees, Binary Tree Traversals: Pre-Order, In-Order and Post-Order Traversals. Representation of Infix, Post-Fix and Prefix Expressions using Trees.

Introduction to Binary Search Trees: B+ trees, AVL Trees, Threaded Binary trees, Balanced Multi-way search trees, Implementation of Heap Sort Algorithm.

Graphs: Basic Terminology, Definition of Undirected and Directed Graphs, Memory Representation of Graphs, Minimum-Spanning Trees, Warshal Algorithm, Graph Traversals Algorithms: Breadth First and Depth First.

Suggested Books:

- Theory and Problems of Data Structures by Jr. Symour Lipschetz, Schaum's outline, TMH.
- Data Structures and Algorithms by PAI, TMH.
- Fundamentals of Data structures by Ellis Horowitz and Sartaj Sahni, Pub, 1983, AW.
- Data Structures and Algorithms by A.V. Aho, J.E. Hopcroft and T.D. Ullman, Original edition, Addison-Wesley, 1999, Low Priced Edition.
- Data Structures and Program Design in C by Robert Kruse, PHI,
- Shukla, Data Structures using C++, Wiley India
- Introduction to Computers Science -An Algorithms Approach, Jean Paul Tremblay, Richard B. Bunt, 2002, T.M.H.
- Data Structure and the Standard Template library – Willam J. Collins, 2003, T.M.H.

Note: The Examiner will be given the question paper template and will have to set the question paper according to the template provided along with the syllabus.

PC-CS-AIDS-207A	Introduction to Artificial Intelligence						
Lecture	**Tutorial**	**Practical**	**Credit**	**Major Test**	**Minor Test**	**Total**	**Time**
3	**0**	**0**	**3**	**75**	**25**	**100**	**3 Hour**
Purpose	To gain a broad understanding of the discipline of Artificial Intelligence and its scope in various emerging areas.						
Course Outcomes(CO)							
CO1	Demonstrate fundamental understanding of Artificial Intelligence (AI) and its foundation						
CO2	Apply basic principles of AI in solutions that require problem solving, inference, perception, knowledge representation, and learning						
CO3	Demonstrate proficiency in applying scientific method to models of machine learning						
CO4	Demonstrate an ability to share in discussions of AI, its current scope and limitations, and societal implications						

UNIT – I

Scope of AI: Games, theorem proving, natural language processing, vision and speech processing, robotics, expert systems, AI techniques-search knowledge, abstraction.

Problem Solving (Blind): State space search; production systems, search space control; depth first search , breadth-first search. Heuristic Based Search: Heuristic search, Hill climbing, best-first search, branch and bound, Problem Reduction, Constraint Satisfaction End, Means-End Analysis.

UNIT – II

Game Playing: Game Tree, Minimax Algorithm, Alpha Beta Cutoff, Modified Minimax Algorithm, Horizon Effect, Futility Cut-off.

Knowledge Representation: Predicate Logic: Unificatioin, Modus Ponens, Modus Tolens, Resolution in Predicate Logic, Conflict Resolution Forward Chaining, Backward Chaining, Declarative and Procedural Representation, Rule based Systems.

Structured Knowledge Representation: Semantic Nets: Slots, exceptions and default frames, conceptual dependency, scripts.

UNIT – III

Knowledge Engineering: First order logic, Syntax and semantics for first order logic, Inference in First order logic – prepositional versus first order logic, unification and lifting, forward chaining, backward chaining , Resolution, Knowledge representation

Handling Uncertainty: Non-Monotonic Reasoning, Probabilistic reasoning, use of certainty factors, fuzzy logic.

Natural Language Processing: Introduction, Syntactic Processing, Semantic Processing, Pragmatic Processing.

UNIT – IV

LEARNING PRINCIPLES : Learning from observations , forms of learning , Inductive learning , Learning decision trees, Ensemble learning, Knowledge in learning,Logical formulation of learning ,Explanation base learning,Learning using relevant information, Inductive logic programming,Statistical learning methods,Learning with complete data ,Learning with hidden variable**,** genetic algorithm, learning by inductions, neural networks.

Expert Systems: Need and justification for expert systems, knowledge acquisition, Case Studies: MYCIN, RI.

Suggested Books:

1. E. Rich and K. Knight, “Artificial Intelligence”, TMH, 2nd Ed., 1992.
2. N. J. Nilsson, “Principles of AI”, Narosa Publ. House, 1990.
3. M. N. Hoda, “Foundation Course in Artificial Intelligence”, Vikas Pub., 2004.
4. P. H. Winston, "Artificial Intelligence", Pearson Education, 3rd Edition, 2002. Artificial Intelligence.
5. D. W. Patterson, “Introduction to AI and Expert Systems”, PHI, 1992.
6. R. J. Schalkoff, “Artificial Intelligence – An Engineering Approach”, McGraw Hill Int. Ed. Singapore, 1992.
7. M. Sasikumar, S. Ramani, “Rule Based Expert Systems”, Narosa Publishing House, 1994. 5. Tim Johns, “Artificial Intelligence, Application Programming, Wiley Dreamtech, 2005.

PC-CS-AIDS- 209A	**Programming Languages**						
Lecture	**Tutorial**	**Practical**	**Credit**	**Major Test**	**Minor Test**	**Total**	**Time**
3	**0**	**0**	**3**	**75**	**25**	**100**	**3 Hour**
Purpose	To introduce the principles and paradigms of programming languages for design and implement the software intensive systems.						
Course Outcomes (CO)							
CO 1	To introduce the basic concepts of programming language, the general problems and methods related to syntax and semantics.						
CO 2	To introduce the structured data objects, subprograms and programmer defined data types.						
CO 3	To outline the sequence control and data control.						
CO 4	To introduce the concepts of storage management using programming languages.						

UNIT – I

Introduction: A brief history, Characteristics of a good programming language, Programming language translators-compiler and interpreters, Elementary data types – data objects, variable and constants, data types. Specification and implementation of elementary data types, Declarations, type checking and type conversions, Assignment and initialization, Numeric data types, enumerations, Booleans and characters.

Syntax and Semantics: Introduction, general problem of describing syntax, Formal method of describing Syntax, attribute grammar dynamic semantic.

UNIT – II

Structured data objects: Structured data objects and data types, specification and implementation of structured data types, Declaration and type checking of data structure, vector and arrays, records Character strings, variable size data structures, Union, pointer and programmer defined data objects, sets, files.

Subprograms and Programmer Defined Data Types: Evolution of data type concept abstraction, encapsulation and information hiding, Subprograms, type definitions, abstract data types, over loaded subprograms, generic subprograms.

UNIT – III

Sequence Control: Implicit and explicit sequence control, sequence control within expressions, sequence control within statement, Subprogram sequence control: simple call return, recursive subprograms, Exception and exception handlers, co routines, sequence control. Concurrency – subprogram level concurrency, synchronization through semaphores, monitors and message passing

Data Control: Names and referencing environment, static and dynamic scope, block structure, Local data and local referencing environment, Shared data: dynamic and static scope, Parameter and parameter transmission schemes.

UNIT – IV

Storage Management: Major run time elements requiring storage, programmer and system controlled storage management and phases, Static storage management, Stack based storage management, Heap storage management, variable and fixed size elements.

Programming Languages: Introduction to procedural, non-procedural, structured, logical, functional and object oriented programming language, Comparison of C and C++ programming languages.

Suggested Books:

- Terrence W. Pratt, Marvin V. Zelkowitz, Programming Languages Design and Implementation, Pearson.
- Allen Tucker and Robert Noonan, Programming Languages–Principles and Paradigms, Tata McGraw-Hill, 2009.
- Ellis Horowitz, Fundamentals of Programming Languages, Galgotia Publications, 2010.
- C. Ghezzi, Programming Languages Concepts, Wiley Publications, 2010.

Note: The Examiner will be given the question paper template and will have to set the question paper according to the template provided along with the syllabus.

HM-902	Business Intelligence and Entrepreneurship						
Lecture	**Tutorial**	**Practical**	**Credit**	**Major Test**	**Minor Test**	**Total**	**Time**
3	**0**	**0**	**3**	**75**	**25**	**100**	**3**
Purpose	To make the students conversant with the basics concepts in management thereby leading to nurturing their managerial skills.						
Course Outcomes (CO)							
CO1	Students will be able understand who the entrepreneurs are and what competences needed to become an Entrepreneur.						
CO2	Students will be able understand insights into the management, opportunity search, identification of a Product; market feasibility studies; project finalization etc. required for small business enterprises.						
CO3	Students can be able to write a report and do oral presentation on the topics such as product identification, business idea, export marketing etc.						
CO4	Students will be able to know the different financial and other assistance available for the small industrial units.						

UNIT – I

Entrepreneurship : Concept and Definitions; Entrepreneurship and Economic Development; Classification and Types of Entrepreneurs; Entrepreneurial Competencies; Factor Affecting Entrepreneurial Growth – Economic, Non-Economic Factors; EDP Programmes; Entrepreneurial Training; Traits/Qualities of an Entrepreneurs; Manager Vs. Entrepreneur, Entrepreneurial challenges.

UNIT – II

Opportunity / Identification and Product Selection: Entrepreneurial Opportunity Search and Identification; Criteria to Select a Product; Conducting Feasibility Studies; Sources of business ideas, Marketing Plan : Conducting of Marketing Research, Industry Analysis, Competitor analysis, market segmentation and positioning, building a marketing plan, marketing mix, launching a new product; export marketing, Methods of Project Appraisal, Project Report Preparation; Specimen of Project Report; Project Planning and Scheduling using Networking Techniques of PERT / CPM.

UNIT – III

Small Enterprises and Enterprise Launching Formalities : Definition of Small Scale; Rationale; Objective; Scope; SSI; Registration; NOC from Pollution Board; Machinery and Equipment Selection , Role of SSI in Economic Development of India; major problem faced by SSI,MSMEs – Definition and Significance in Indian Economy; MSME Schemes, Challenges and Difficulties in availing MSME Schemes.

UNIT – IV

Role of Support Institutions and Management of Small Business : DIC; SIDO; SIDBI; Small Industries Development Corporation (SIDC); SISI; NSIC; NISBUD; State Financial Corporation SIC; Venture Capital : Concept, venture capital financing schemes offered by various financial institutions in India.

Special Issues for Entrepreneurs: Legal issues – Forming business entity, requirements for formation of a Private/Public Limited Company, Entrepreneurship and Intellectual Property Rights: IPR and their importance. (Patent, Copy Right, Trademarks) , Case Studies-At least one in whole course.

Note:

• Case studies of Entrepreneurs – successful, failed, turnaround ventures should be discussed in the class.
• Exercises / activities should be conducted on ‘generating business ideas’ and identifying problems and opportunities.
• Interactive sessions with Entrepreneurs, authorities of financial institutions, Government officials should be organized

Suggested Readings:

1. "Entrepreneurship development small business enterprises", Pearson, Poornima M Charantimath,2013.
2. Roy Rajiv, "Entrepreneurship", Oxford University Press, 2011.
3. "Innovation and Entrepreneurship",Harper business- Drucker.F, Peter, 2006.
4. "Entrepreneurship", Tata Mc-graw Hill Publishing Co.ltd new Delhi- Robert D. Hisrich, Mathew J. Manimala, Michael P Peters and Dean A. Shepherd, 8th Edition, 2012
5. Enterpreneurship Development- S.Chand and Co.,Delhi- S.S.Khanka 1999
6. Small-Scale Industries and Entrepreneurship. Himalaya Publishing House, Delhi –Vasant Desai 2003.
7. Entrepreneurship Management -Cynthia, Kaulgud, Aruna, Vikas Publishing House, Delhi, 2003.

Note: The Examiner will be given the question paper template and will have to set the question paper according to the template provided along with the syllabus.

PC-CS-AIDS-213LA	**Data Structure & Algorithms Lab**						
Lecture	**Tutorial**	**Practical**	**Credit**	**Minor Test**	**Practical**	**Total**	**Time**
0	**0**	**2**	**1**	**40**	**60**	**100**	**3**
Purpose	To introduce the principles and paradigms of Data Structures for design and implement the software systems logically and physically.						
Course Outcomes (CO)							
CO1	Implement linear and non linear data structures using linked list.						
CO2	Apply various data structures such as stack, queue and tree to solve the problems.						
CO3	Implement various searching and sorting techniques.						
CO4	Choose appropriate data structure while designing the applications and analyze the complexity of the algorithms.						

LIST OF PRACTICALS

1. Write a program for Binary search methods.
2. Write a program for insertion sort, selection sort and bubble sort.
3. Write a program to implement Stack and its operation.
4. Write a program for quick sort.
5. Write a program for merge sort.
6. Write a program to implement Queue and its operation.
7. Write a program to implement Circular Queue and its operation.
8. Write a program to implement singly linked list for the following operations: Create, Display, searching, traversing and deletion.
9. Write a program to implement doubly linked list for the following operations: Create, Display, inserting, counting, searching, traversing and deletion.

10 Write a program to implement circular linked list for the following operations: Create, Display, inserting, counting, searching, traversing and deletion.

11. Write a program to implement insertion, deletion and traversing in B tree

NOTE: A student has to perform at least ten experiments. Seven experiments should be performed from the above list. Remaining three experiments may either be performed from the above list or designed & set by the concerned institution as per the scope of the syllabus.

PC-CS-AIDS-215LA	Artificial Intelligence Lab						
Lecture	**Tutorial**	**Practical**	**Credit**	**Minor Test**	**Practical**	**Total**	**Time**
0	**0**	**2**	**1**	**40**	**60**	**100**	**3**
Purpose	To gain a broad understanding of the discipline of Artificial Intelligence						
Course Outcomes							
CO1	To understand the basic concepts of Artificial Intelligence.						
CO2	To apply various AI Search algorithms.						
CO3	To understand the fundamentals of knowledge representation and theorem proving using AI tools.						
CO4	Ability to apply knowledge representation and machine learning techniques to real life problems.						

LIST OF PRACTICALS:

1. Introduction to PYTHON & study of basic commands.
2. Write a program to implement Factorial, Fibonacci of a given number.
3. Write a program to solve 8 queens problem.
4. Solve a problem using Depth First Search.
5. Solve a problem using Breadth First Search.
6. Solve 8 puzzle problem using Best First Search
7. Solve Robot Traversal Problem using Means End Analysis
8. Solve Travelling Salesman Problem.

Suggested Books:

1. Artificial Intelligence: A Modern Approach,. Russell & Norvig, Prentice Hall.
2. Artificial Intelligence, Elain Rich and Kevin Knight, TMH.
3. Artificial Intelligence-A modern approach, Staurt Russel and peter norvig, 1998, PHI.
4. Artificial intelligence, Patrick Henry Winston:, 1992, Addition Wesley 3 Ed.

PC-CS-AIDS-217LA	**Object Oriented Programming Lab**						
Lecture	**Tutorial**	**Practical**	**Credit**	**Minor Test**	**Practical**	**Total**	**Time**
0	**0**	**2**	**1**	**40**	**60**	**100**	**3 Hour**
Purpose	To introduce the principles and paradigms of Object Oriented Programming Language for design and implement the Object Oriented System.						
Course Outcomes (CO)							
CO1	Implement object oriented concepts such as objects,classes abstraction and message passing.						
CO2	Implement the friend function ,function overloading and virtual function						
CO3	Implement Operator overloading, Inheritance and method overriding.						
CO4	Implement the various functions on String and apply I/O operation to handle file system						

LIST OF PRACTICALS

1. Raising a number n to a power p is the same as multiplying n by itself p times. Write a function called power

( ) that takes a double value for n and an int value for p, and returns the result as double value. Use a default argument of 2 for p, so that if this argument is omitted, the number will be squared. Write a main ( ) function that gets values from the user to test this function.

2. A point on the two dimensional plane can be represented by two numbers: an X coordinate and a Y coordinate. For example, (4,5) represents a point 4 units to the right of the origin along the X axis and 5 units up the Y axis. The sum of two points can be defined as a new point whose X coordinate is the sum of the X coordinates of the points and whose Y coordinate is the sum of their Y coordinates. Write a program that uses a structure called point to model a point. Define three points, and have the user input values to two of them. Then set the third point equal to the sum of the other two, and display the value of the new point. Interaction with the program might look like this:

```
Enter coordinates for P1: 3 4

Enter coordinates for P2: 5 7

Coordinates of P1 + P2 are : 8, 11
```

3. Create the equivalent of a four function calculator. The program should request the user to enter a number, an operator, and another number. It should then carry out the specified arithmetical operation: adding, subtracting, multiplying, or dividing the two numbers. (It should use a switch statement to select the operation). Finally it should display the result. When it finishes the calculation, the program should ask if the user wants to do another calculation. The response can be ‘Y’ or ‘N’. Some sample interaction with the program might look like this.

```
Enter first number, operator, and second number: 10/ 3

    Answer = 3.333333
    Do another (Y/ N)? Y
    Enter first number, operator, second number 12 + 100
    Answer = 112
    Do another (Y/ N) ? N
```

4. A phone number, such as (212) 767-8900, can be thought of as having three parts: the area code (212), the exchange (767) and the number (8900). Write a program that uses a structure to store these three parts of a phone number separately. Call the structure phone. Create two structure variables of type phone. Initialize one, and have the user input a number for the other one. Then display both numbers. The interchange might look like this:

- Enter your area code, exchange, and number: 415 555 1212
- My number is (212) 767-8900
- Your number is (415) 555-1212

5. Create two classes DM and DB which store the value of distances. DM stores distances in metres and centimeters and DB in feet and inches. Write a program that can read values for the class objects and add one object of DM with another object of DB. Use a friend function to carry out the addition operation. The object that stores the results maybe a DM object or DB objects, depending on the units in which the results are required. The display should be in the format of feet and inches or metres and cenitmetres depending on the object on display.

6. Create a class rational which represents a numerical value by two double values- NUMERATOR and DENOMINATOR. Include the following public member Functions:

- constructor with no arguments (default).
- constructor with two arguments.
- void reduce( ) that reduces the rational number by eliminating the highest common factor between the numerator and denominator.
- Overload + operator to add two rational number.
- Overload >> operator to enable input through cin.
- Overload << operator to enable output through cout.

Write a main ( ) to test all the functions in the class.

7. Consider the following class definition

```
class father {

protected : int age;

public;

father (int x) {age = x;}

virtual void iam ( )

{ cout < < “I AM THE FATHER, my age is : ”<< age<< end1:}
```

Derive the two classes son and daughter from the above class and for each, define iam ( ) to write our similar but appropriate messages. You should also define suitable constructors for these classes. Now, write a main ( ) that creates objects of the three classes and then calls iam ( ) for them. Declare pointer to father. Successively, assign addresses of objects of the two derived classes to this pointer and in each case, call iam ( ) through the pointer to demonstrate polymorphism in action.

8. Write a program that creates a binary file by reading the data for the students from the terminal. The data of each student consist of roll no., name (a string of 30 or lesser no. of characters) and marks.

9. A hospital wants to create a database regarding its indoor patients. The information to store include

a) Name of the patient

b) Date of admission

c) Disease

d) Date of discharge

Create a structure to store the date (year, month and date as its members). Create a base class to store the above information. The member function should include functions to enter information and display a list of all the patients in the database. Create a derived class to store the age of the patients. List the information about all the to store the age of the patients. List the information about all the pediatric patients (less than twelve years in age).

10. Make a class **Employee** with a name and salary. Make a class **Manager** inherit from **Employee**. Add an instance variable, named department, of type string. Supply a method to **to String** that prints the manager's name, department and salary. Make a class **Executive** inherits from **Manager**. Supply a method **to String** that prints the string **"Executive"** followed by the information stored in the **Manager** superclass object. Supply a test program that tests these classes and methods.

11. Imagine a tollbooth with a class called toll Booth. The two data items are a type unsigned int to hold the total number of cars, and a type double to hold the total amount of money collected. A constructor initializes both these to 0. A member function called payingCar ( ) increments the car total and adds 0.50 to the cash total. Another function, called nopayCar ( ), increments the car total but adds nothing to the cash total. Finally, a member function called displays the two totals. Include a program to test this class. This program should allow the user to push one key to count a paying car, and another to count a nonpaying car. Pushing the ESC kay should cause the program to print out the total cars and total cash and then exit.

12. Write a function called reversit ( ) that reverses a string (an array of char). Use a for loop that swaps the first and last characters, then the second and next to last characters and so on. The string should be passed to reversit ( ) as an argument. Write a program to exercise reversit ( ). The program should get a string from the user, call reversit ( ), and print out the result. Use an input method that allows embedded blanks. Test the program with Napoleon's famous phrase, "Able was I ere I saw Elba)".

13. Create some objects of the string class, and put them in a Deque-some at the head of the Deque and some at the tail. Display the contents of the Deque using the forEach ( ) function and a user written display function. Then search the Deque for a particular string, using the first That ( ) function and display any strings that match. Finally remove all the items from the Deque using the getLeft ( ) function and display each item. Notice the order in which the items are displayed: Using getLeft ( ), those inserted on the left (head) of the Deque are removed in "last in first out" order while those put on the right side are removed in "first in first out" order. The opposite would be true if getRight ( ) were used.

14. Assume that a bank maintains two kinds of accounts for customers, one called as savings account and the other as current account. The savings account provides compound interest and withdrawal facilities but no cheque book facility. The current account provides cheque book facility but no interest. Current account holders should also maintain a minimum balance and if the balance falls below this level, a service charge is imposed.

Create a class account that stores customer name, account number and type of account. From this derive the classes

cur_acct and sav_acct to make them more specific to their requirements. Include necessary member functions in order to achieve the following tasks:

a) Accept deposit from a customer and update the balance.

b) Display the balance.

c) Compute and deposit interest.

d) Permit withdrawal and update the balance.

e) Check for the minimum balance, impose penalty, necessary and update the balance.

f) Do not use any constructors. Use member functions to initialize the class members.

15. Create a base class called shape. Use this class to store two double type values that could be used to compute the area of figures. Derive two specific classes called triangle and rectangle from the base shape. Add to the base class, a member function get_data( ) to initialize baseclass data members and another member function display_area( ) to compute and display the area of figures. Make display_area ( ) as a virtual function and redefine this function in the derived classes to suit their requirements. Using these three classes, design a program that will accept dimensions of a triangle or a rectangle interactively and display the area.

Remember the two values given as input will be treated as lengths of two sides in the case of rectangles and as base and height in the case of triangles and used as follows:

Area of rectangle = $x * y$

Area of triangle = $\frac{1}{2} * x * y$

NOTE: A student has to perform at least ten experiments. Seven experiments should be performed from the above list. Remaining three experiments may either be performed from the above list or designed & set by the concerned institution as per the scope of the syllabus.