Object Oriented Programming  
COT–202

L  T  P  Theory:  75  
3  1  -  Sessional: 25

1. Object Oriented Programming and Design  

2. C++ Programming Basics  
Fundamentals: Variables and assignments, Input and Output, Data Types and Expressions, Flow of control, Subprograms: Top down design, Predefined functions, Programmer defined functions, Procedural abstractions, Local variables, Overloading function names, Operator overloading, Parameter passing, this pointer, Destructors, Copy constructors, Overloading the assignment operator, Virtual functions, Function Calling functions, Friend functions, Recursive function, Recursive member function.

3. C++ Object Oriented Concepts  
Objects and Classes: Use of file for I/O, Formatting output with stream functions, Character I/O, Inheritance, Structures for diverse data, Structures as function arguments, Initializing structures, Defining classes and member functions, Public and private members, Constructors for initializations, Standard C++ classes, Derived classes, Flow of Control, Use of Boolean expressions, Multiway branches, Use and design of loops.

4. C++ Data Structures and Advanced Topics  
Arrays – Programming with arrays, arrays of classes, arrays as function arguments, Strings, Multidimensional arrays, Arrays of strings, Pointers, Dynamic arrays, Classes and dynamic arrays, Base classes, access control, Templates – generic classes and functions, namespaces.

5. Introduction to Java  
Data types, Variables and Assignment, String and Characters, Arrays, Control statements, Loops, Operators. Introduction to Classes, Constructors, this keyword, Static, Local and Instance variables, Methods, Method overloading, Method overriding, subclasses, inheritance, modifiers, polymorphism.

Note:- At least one question will be set from each section.

BOOKS  
1. Herb Schildt: C++ - The Complete Reference, TMH, Delhi  
Programming Languages Concepts
COT-204

L T P Theory: 75
3 1 - Sessional: 25

1. Describing Syntax and Semantics
Introduction, general problem of describing syntax, formal methods of describing syntax, attribute grammar dynamic semantics.

2. Data Types
Name, variables, binding, types of binding, type checking, strong typing, type conversion, named constant, principle data types, character string, user defined data types, pointer and reference. Discuss with reference to C, C++, and JAVA.

3. Subprogram
Fundamentals of subprogram, referencing, environment – local and non local parameter passing, subprogram name as parameter, overloaded subprogram, generic subprograms, coroutine, CALL-RETURN structure, recession, implementing non-local referencing environment, scope-static and dynamic, implementation of scopes. Discuss with reference to C, C++, and JAVA.

4. Concurrency and Exception Handling
Introduction, subprogram level concurrency, and synchronization, through semaphores, monitors and message passing, Introduction to Exception handling. Discuss with reference to C, C++, and JAVA.

5. Storage Management
Major routine storage elements, programmer and system controlled storage management, storage management phases, static storage management, stock based storage management, heap storage management –fixed size and variable size.

6. Functional Programming & Logical Programming
Fundamentals of functional programming, LISP: data types, functions, control flow, applications, Overview of logical programming, basic elements of Prolog, deficiencies and application of Prolog.

Note:- Question paper will contain at least one question from each unit and two questions from unit 3.

BOOKS
2. T.W. Pratt, Programming Languages: Design & Implementation, PHI, 3rd Ed.
1. **Introduction**
   Introduction to Software crisis & Software processes; Software life cycle models – Build & Fix, waterfall prototype evolutionary, spiral model.

2. **Requirement Analysis & Specifications**
   Problem Analysis – DFD, Data dictionaries, ER diagrams, object diagrams; approaches to problems analysis; SRS; specifying behavioral & non-behavioral requirements.

3. **Software Design**
   What is design? Modularity, strategy of design, function oriented design, object oriented design.

4. **Software Metrics**
   Introduction, size metrics, data structure metrics, information flow metrics, entropy-based measures, metric analysis.

5. **Software Reliability**
   Importance, Software reliability & Hardware reliability, failures & faults, reliability concepts, reliability models – macro, basic, logarithmic Poisson, calendar time component, micro models; estimating number of residual errors; reliability allocation.

6. **Software Testing**
   Introduction, Functional testing, structural testing, activities during testing, debugging, testing tools.

7. **Software Maintenance**
   Introduction, types of maintenance, maintenance process, maintenance models, reverse engineering, re-engineering.

**Note:** There will be 8 question in all. Attempt any 5 questions. There will be max 1 question from unit 1 & unit 6 each and minimum 1 question from each of remaining units.

**BOOKS**
Microprocessors I  
COT-208

L  T  P  Theory:  75
3  1  -  Sessional:  50

1.  **8086 Architecture**  
   CPU architecture, internal operation, machine language instruction, instruction execution time.

2.  **Assembly Language Programming**  
   Assembler, instruction format, data transfer instruction, arithmetic instructions, branch instruction, NOP & HLT instructions, flag manipulation instruction, logical instruction, shift and rotate instruction, directions and operators.

3.  **Modular Programming**  
   Stacks, Procedures, Interrupt and interrupt routines, macros – local labels and nested macros.

4.  **Strings and I/O Programming**  
   String instruction, I/O consideration, programmed I/O block transfer and DMA.

5.  **I/O Interface**  
   Serial communication, asynchronous, synchronous, physical, 8251A; Parallel communication: 8255 A, DMA controllers; maximum mode, 16-bit bus interface.

6.  **80286 Architecture**  
   Salient features of 80286, Internal architecture, real and protected virtual addressing modes, protection, 80286 minimum system configuration, instruction set features.

**Note:** At least one question will be set from each unit.

**BOOKS**
2. D.V. Hall, Microprocessors and Interfacing, TMH.
1. **Spectral Analysis**
Fourier series, Response of linear system Power spectral density, Fourier Transform, Convolution, Parseval’s Theorem, correlation between waveforms, Impulse Function, ideal low pass filter, Hilbert transform, Pre-envelope.

2. **Random Variables & Noise**

3. **Analog Modulation Techniques**
Introduction, Amplitude Modulation, AM demodulators, Spectrum of AM signal, Double sideband suppressed carrier modulation, single side band modulation, Methods of generating SSB signals, vertical side band modulation, frequency division multiplexing; Angle modulation, Phase and frequency modulation, spectrum of F.M. signal bandwidth of FM signal; Effect of modulation index on bandwidth; NBFM & WBFM FM generation methods, FM Demodulation methods.

4. **Pulse Modulation**
Sampling theorem for low pass and band pass signals, time division multiplexing, concept of pulse amplitude modulation and pulse width modulation, demodulation of signals, pulse code modulation, Noise computation in PCM systems, Delta modulation and adaptive delta modulation.

5. **Digital Modulation Techniques**
Binary phase shift keying, differential phase shift keying, quadrature phase shift keying, M-ary PSK, QASK, Binary FSK, M-ary FSK, Minimum shift keying.

6. **Spread Spectrum Signals**
Spread spectrum model, direct sequence spread signals, generation of PN sequences, CDMA system based on frequency hopped spread spectrum signal.

**Note:** At least one question will be set from each unit.

**BOOKS**
3. J. G. Proakis: Digital Communications, MGH.
Object Oriented Programming (Pr)
COT-212

L   T   P       Practical:  25
-   -   2       Sessional:  50

Note:- Write programs first in C++ and later in Java.

1. (a) Model a geometric point to find distance between two points.
   (b) Model complex numbers and their operations.

2. Describe a class called TOLL-BOOTH with the following data items
   unsigned int - to hold the number of cars passing through the booth,
   double       - to hold the total amount collected.

   Include the following member functions:
   * a constructor that sets both the data fields to zero.
   * PAYINGCAR( ) that increases the numbers of cars by one and increase
     the total amount by 2.50.
   * NOPAYING( ) that increases the number of cars but keeps the total amount
     unchanged.
   * DISPLAY( ) that displays both the total number of cars passing and the
     total number of amount collected.

   Write main( ) to test the class thoroughly.

3. 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 numbers.
   * 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 .

4. 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<<endl;}

Derive the two classes son and daughter from the above class and for each, define iam() to write out 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.

5. A thermostat is a device that keeps a system at a constant temperature. It behaves like a temperature gauge that is capable of getting the current temperature from the system. It is also a switch that can be turned "on" and "off". The thermostat monitors the temp. in the following manner:
   if the current temp. falls below 95% of the required temp., it turns itself "on". On the other hand, if the current temp. exceeds 1.05 of the required temp. ,it turns itself "off" .In all other cases ,its on-off status remain unchanged.

Implement classes for temp. gauge and switch(named switch) with suitable data and member functions. The temp. gauge class must have a member function get_temp() that will pretend to get the current temp. of the system by actually reading it from the keyboard.

Now, implement thermostat class in both the following ways:
   a) Develop a class called thermostat that include objects of temp. gauge and switch as its member(aggregation).  
   b) Develop a class called thermostat that inherits the data methods of temp. gauge and switch(multiple inheritance).

Write main() to test all the features of above mentioned classes.

6. 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.

7. Using the file created in problem 6, write a program to display the roll no. and names of the students who have passed (has obtained 50 or more).

8. You are to create a file containing n records. Each record relates to a historical event and the year in which the event took place
   Some examples are:
   India Wins Freedom 1947
   Amartya Sen Gets Nobel 1998
   First World War Begins 1914
   The data should be read from terminal while creating the file.

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 databases. Create a derived class to store the age of the patients. List the information about all the pediatric patients (less than twelve years in age).

10. Define a class to store the time at a point. The data members should include hr., min., and sec. to store hours, minutes and seconds. The member functions should include functions for reading the time and displaying the same. Add a friend function to add two times. Write a program, using the above declaration, to read two times and add them.

11. Write a program to read two matrices and find their product. Use operator overloading so that the statement for multiplying the matrices may be written as $Z = X * Y$ where $X$, $Y$ and $Z$ are matrices.

12. Write a program to read a number and display its square, square root, cube and cube root. Use a virtual function to display any one of the above.

13. 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 toString that prints the manager’s name, department and salary. Make a class Executive inherit from Manager. Supply a method toString 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.

14. Write a superclass Worker and subclass HourlyWorker and SalariedWorker. Every worker has a name and a salary rate. Write a method computePay(int hours) that computes the weekly pay for every worker. An hourly worker gets paid the hourly wage for the actual number of hours worked, if hours is at most 40. If the hourly worker worked more than 40 hours, the excess is paid at time and a half. The salaried worker gets paid the hourly wage for 40 hours, no matter what the actual number of hours is. Write a static method that uses polymorphism to compute the pay of any Worker. Supply a test program that tests these classes and methods.

BOOKS
5. Teach Yourself Java by Joseph O’Neil (TMH, Delhi).
1. Write a program to print the alphabets.
2. Write a program to read a integer number of max (16 bit), store that number in a register and display it digit by digit.
3. Repeat exercise 2 for 32 bit number.
4. Write a program to find factorial of a number, where result does not exceed 32 bit. Use procedure to calculate factorial and pass parameters
5. Write modular program to perform addition, subtraction, multiplication and division of two 16-bit numbers.
6. Repeat exercise 5 for two 32-bit numbers
7. Sort n numbers using modular program.
8. Check whether a given string is palindrome or not.
9. Reverse an input string.
10. Merge two sorted list of integers
11. Using int 10h, change the size of cursor, change the position of the cursor based on user’s choice.
12. Write some programs, which use multiple data segments and multiple code segments. Do these programs by defining different segments in different files and link all of them to get the desired output.
Software Engineering (Pr)
COT-216

L   T   P
-   -   3

Practical: 50
Sessional: 50

Note:- Implement the following programs using C.

1. Implement Halstead’s equation to compute various science metrics like volume etc., language level, estimated program length, effort and time in a program.
2. Compute average number of live variables per statement in a program.
3. Compute average life of variables in a program.
4. Compute psychological complexity of a program.
5. Compute McCabe’s cyclomatic complexity of a program and generate its control graph.
6. Use some CASE tool for identifying various phases of software engineering, generate SRS document, design document like DFD and ER diagram, test cases generation for result automation, engineering admission automation (seat allocation during counseling).
Communication (Pr)
ECT-234

L      T      P      Practical: 25
-      -      2      Sessional: 25

List of Experiments:

2. Study of PAM. (Modulation/Demodulation).
5. Study of PCM. (Modulation/Demodulation).
7. Study of ADM. (Modulation/Demodulation).
9. Measurement of bit error rate BER of PSK.
10. Measurement of bit error rate BER of FSF.