Semi Design

find you way in #VLSI with us

C++

- About C++
- Characteristics of Object Oriented languages
    - Objects
    - Classes
    - Inheritance
    - Reusability
    - Creating new data types
    - Polymorphism and Overloading
    - C++ v/s C
    - Output using cout
    - Input using cin
    - The stew manipulator
- Library Functions
    - Header files
    - Library files
    - Two ways to use #include
- Enumerations
- A simple class
    - Classes and objects
    - Declaring the class
    - Using the class
    - Calling member Functions
- A C++ objects as physical objects
- A C++ objects as Data types
- Nesting of member functions
- Private member functions
- Array within a class

Static data member **C++ Module**

- What is c++
- Characteristics of Object Oriented languages
    - Objects
    - Classes
    - Inheritance
    - Reusability
    - Creating new data types
    - Polymorphism and Overloading
- C++ v/s C

- Output using cout
- Input using cin
- The setw manipulator
- Library Functions
  - Header files
  - Library files
  - Two ways to use #include
- Enumerations
- A simple class
  - Classes and objects
  - Declaring the class
  - Using the class
  - Calling member Functions
- A C++ objects as physical objects
- A C++ objects as Data types
- Nesting of member functions
- Private member functions
- Array within a class
- Static data members & member functions
- Arrays of objects
- Local classes
- Constructors
  - A counter example
  - A graphics example
  - A parameterized constructor
  - Multiple constructor in a class
  - Dynamic initialization of objects
  - Copy constructor
  - Dynamic constructor
- Destructor
- Returning objects from functions
- Structures and classes
- Classes, objects and memory
- Static class data
  - Uses of static class data
  - An example of static class data
  - Separate declaration and definition
- Const member function and const objects
- Overloaded Functions
- Inline functions
- Variable and storage classes
- Const functions arguments
- Virtual Functions
  - Normal Member Functions Accessed with Pointers
  - Virtual Member Functions Accessed with Pointers

- Late Binding
- Abstract Classes and Pure Virtual Functions
- Virtual Functions and the person Class
- Virtual Functions in a Graphics Example
- Virtual Destructors
- Virtual Base Classes

- Friend Functions
  - Friends as Bridges
  - English Distance Example
  - friends for Functional Notation
  - friend Classes
- Static Functions
  - Accessing static Functions
  - Numbering the Objects
  - Investigating Destructors
- The this Pointer
  - Accessing Member Data with this
  - Using this for Returning Values
- Dynamic Type Information
  - Checking the Type of a Class with dynamic_cast
  - Changing Pointer Types with dynamic_cast
- Overloading Unary Operator
  - The operator keyword
  - Operator arguments
  - Operator return values
  - Nameless temporary Objects
  - Postfix notations
- Overloading Binary Operators
  - Arithmetic operators
  - Concatenating strings
  - Multiples overloading
  - Comparison Operators
  - Arithmetic assignment operators
  - The Subscript notation []
- Data Conversion
  - Conversion between basic types
  - Conversion between objects and basic types
  - Conversion between objects of different class
- Keywords explicit and mutable
  - Preventing conversions with explicit
  - Changing const objects Data using mutable
- Derived Class and Base Class
  - Specifying the Derived Class
  - Accessing Base Class Members
  - The protected Access Specifier

- Derived Class Constructors
- Overriding Member Functions
- Inheritance in the English Distance Class
  - Operation of ENGLEN
  - Constructors in DistSign
  - Member Functions in DistSign
  - Abetting Inheritance
- Class Hierarchies
  - “Abstract” Base Class
  - Constructors and Member Functions
- Public and Private Inheritance
- Levels of Inheritance
- Multiple Inheritance
- Member Functions in Multiple Inheritance
- Ambiguity in Multiple Inheritance
- Containership: Classes Within Classes
- Inheritance and Program Developments
- Function Templates
  - A Simple Function Template
  - Function Templates with Multiple Arguments
- Class Templates
  - Class Name Depends on Context
  - A Linked List Class Using Templates
  - Storing User-Defined Data Types
- Exceptions
  - Why Do We Need Exceptions?
  - Exception Syntax
  - A Simple Exception Example
  - Multiple Exceptions
  - Exceptions with the Distance Class
  - Exceptions with Arguments
  - Extracting Data from the Exception Object
  - The bad_alloc Class

- s & member functions
- Arrays of objects
- Local classes
- Constructors
  - A counter example
  - A graphics example
  - A parameterized constructor
  - Multiple constructor in a class
  - Dynamic initialization of objects
  - Copy constructor
  - Dynamic constructor

- Destructor
- Returning objects from functions
- Structures and classes
- Classes, objects and memory
- Static class data
    - Uses of static class data
    - An example of static class data
    - Separate declaration and definition
- Const member function and const objects
- Overloaded Functions
- Inline functions
- Variable and storage classes
- Const functions arguments
- Virtual Functions
    - Normal Member Functions Accessed with Pointers
    - Virtual Member Functions Accessed with Pointers
    - Late Binding
    - Abstract Classes and Pure Virtual Functions
    - Virtual Functions and the person Class
    - Virtual Functions in a Graphics Example
    - Virtual Destructors
    - Virtual Base Classes
- Friend Functions
    - Friends as Bridges
    - English Distance Example
    - friends for Functional Notation
    - friend Classes
- Static Functions
    - Accessing static Functions
    - Numbering the Objects
    - Investigating Destructors
- The this Pointer
    - Accessing Member Data with this
    - Using this for Returning Values
- Dynamic Type Information
    - Checking the Type of a Class with dynamic_cast
    - Changing Pointer Types with dynamic_cast
- Overloading Unary Operator
    - The operator keyword
    - Operator arguments
    - Operator return values
    - Nameless temporary Objects
    - Postfix notations

- Overloading Binary Operators
  - Arithmetic operators
  - Concatenating strings
  - Multiples overloading
  - Comparison Operators
  - Arithmetic assignment operators
  - The Subscript notation []
- Data Conversion
  - Conversion between basic types
  - Conversion between objects and basic types
  - Conversion between objects of different class
- Keywords explicit and mutable
  - Preventing conversions with explicit
  - Changing const objects Data using mutable
- Derived Class and Base Class
  - Specifying the Derived Class
  - Accessing Base Class Members
  - The protected Access Specifier
- Derived Class Constructors
- Overriding Member Functions
- Inheritance in the English Distance Class
  - Operation of ENGLEN
  - Constructors in DistSign
  - Member Functions in DistSign
  - Abetting Inheritance
- Class Hierarchies
  - “Abstract” Base Class
  - Constructors and Member Functions
- Public and Private Inheritance
- Levels of Inheritance
- Multiple Inheritance
- Member Functions in Multiple Inheritance
- Ambiguity in Multiple Inheritance
- Containership: Classes Within Classes
- Inheritance and Program Developments
- Function Templates
  - A Simple Function Template
  - Function Templates with Multiple Arguments
- Class Templates
  - Class Name Depends on Context
  - A Linked List Class Using Templates
  - Storing User-Defined Data Types
- Exceptions

- Why Do We Need Exceptions?
- Exception Syntax
- A Simple Exception Example
- Multiple Exceptions
- Exceptions with the Distance Class
- Exceptions with Arguments
- Extracting Data from the Exception Object
- The bad_alloc Class