CS304
Object Oriented Programming

Important mcqs

Lec 1 - INTRODUCTION

1. **What is the purpose of an introduction?**
   A) To summarize the entire work
   B) To provide a roadmap for the reader
   C) To include all the details of the content
   D) To present the conclusion first
   Answer: B

   Which of the following is a characteristic of a good introduction?
   A) Lengthy and repetitive
   B) Unclear and confusing
   C) Concise and informative
   D) Irrelevant to the topic
   Answer: C

   What does an introduction provide for the reader?
   A) A summary of the conclusion
   B) An overview of the entire work
   C) The main body of the content
   D) None of the above
   Answer: B

   What should be the tone of an introduction?
   A) Informative
   B) Persuasive
   C) Formal
   D) All of the above
   Answer: A

   Which of the following is NOT an objective of an introduction?
   A) To provide context
   B) To grab the reader's attention
   C) To present the conclusion first
   D) To introduce the topic
   Answer: C

   What is the ideal length of an introduction?
   A) Half of the total work
   B) One third of the total work
   C) One quarter of the total work

D) It depends on the length of the work
Answer: C

Which of the following should be included in an introduction?
A) All the details of the content
B) A thesis statement
C) The entire conclusion
D) None of the above
Answer: B

What should be the structure of an introduction?
A) Chronological
B) Random
C) Logical
D) None of the above
Answer: C

What is the main function of an introduction?
A) To provide a conclusion
B) To grab the reader's attention
C) To provide context
D) None of the above
Answer: C

Which of the following is a common mistake to avoid in an introduction?
A) Being too general
B) Being too specific
C) Being too formal
D) All of the above
Answer: A

Lec 2 - INFORMATION HIDING

1. **What is a model?**
   A. A simplified representation of a complex system
   B. A complicated representation of a simple system
   C. A physical replica of a real-world phenomenon
   D. A mathematical equation used in statistics
   Answer: A

   What is the main purpose of a model?
   A. To perfectly capture the complexity of reality
   B. To simulate real-world phenomena
   C. To create a physical replica of a system
   D. To replace the need for experimentation
   Answer: B

   Which fields use models?
   A. Science and engineering
   B. Economics and finance
   C. Computer science and technology
   D. All of the above
   Answer: D

   What are the limitations of a model?
   A. It can make assumptions that may affect its accuracy
   B. It can only capture the complexity of reality perfectly
   C. It is always expensive to develop
   D. It cannot be used to inform decision-making
   Answer: A

   What is an example of a model?
   A. A physical replica of a car
   B. A computer program simulating traffic flow
   C. A mathematical equation representing the weather
   D. All of the above
   Answer: D

   Why are models useful in science?
   A. They allow for the testing of hypotheses
   B. They replace the need for experimentation
   C. They can perfectly capture the complexity of reality
   D. They are always more accurate than real-world data
   Answer: A

   What is the purpose of a mathematical model?
   A. To simulate real-world phenomena
   B. To create a physical replica of a system
   C. To make assumptions about a system
   D. To predict outcomes and inform decision-making
   Answer: D

   What is a disadvantage of using a physical model?
   A. It is always cheaper to develop than other types of models

B. It can be difficult to accurately replicate a real-world phenomenon
C. It cannot be used to test hypotheses
D. It is not suitable for informing decision-making
Answer: B

What is the difference between a model and a theory?
A. A theory is a type of model
B. A model is a type of theory
C. A theory is a well-established explanation for a phenomenon, while a model is a simplified representation of a system
D. A model is more accurate than a theory
Answer: C

How can models be improved?
A. By incorporating more complex variables
B. By reducing the number of assumptions made
C. By including real-world data
D. All of the above
Answer: D

Lec 3 - ABSTRACTION

1. **What is abstraction?**
   A) A process of creating detailed representations of an object
   B) A process of simplifying complex ideas by removing unnecessary details
   C) A process of defining precise specifications of a system
   D) A process of optimizing code for performance
   Answer: B) A process of simplifying complex ideas by removing unnecessary details

   Which field does abstraction play a crucial role in?
   A) Mathematics
   B) Computer science
   C) Art
   D) All of the above
   Answer: D) All of the above

   In computer science, what does abstraction refer to?
   A) A process of hiding implementation details while exposing essential functionality
   B) A process of creating complex algorithms
   C) A process of testing software for bugs
   D) A process of designing user interfaces
   Answer: A) A process of hiding implementation details while exposing essential functionality

   What is the benefit of abstraction in computer science?
   A) More efficient and maintainable software systems
   B) More complex and difficult to understand software systems
   C) Faster software development
   D) Higher quality software systems
   Answer: A) More efficient and maintainable software systems

   Which art form utilizes abstraction to represent the essence of a subject?
   A) Realism
   B) Impressionism
   C) Abstract expressionism
   D) Surrealism
   Answer: C) Abstract expressionism

   What is the cognitive process involved in abstraction?
   A) Creating detailed representations of an object
   B) Simplifying complex ideas by removing unnecessary details
   C) Analyzing complex systems
   D) Memorizing information
   Answer: B) Simplifying complex ideas by removing unnecessary details

   Which of the following is an example of abstraction in mathematics?
   A) Using variables to represent unknown values in equations
   B) Solving complex equations without simplification
   C) Graphing equations without labeling the axes
   D) Using only whole numbers in calculations
   Answer: A) Using variables to represent unknown values in equations

   What is the purpose of abstraction in philosophy?
   A) To understand the nature of reality

B) To create complex arguments
C) To study the history of philosophy
D) To memorize philosophical theories
Answer: A) To understand the nature of reality

Which of the following is NOT a benefit of abstraction?
A) More efficient and maintainable software systems
B) Simplified understanding of complex systems
C) Increased complexity of systems
D) Improved problem-solving ability
Answer: C) Increased complexity of systems

Which term refers to the level of abstraction that focuses on the essential features of a system?
A) High-level abstraction
B) Low-level abstraction
C) Mid-level abstraction
D) No abstraction
Answer: A) High-level abstraction

Lec 4 - CONCEPTS RELATED WITH INHERITANCE

1. **In object-oriented programming, what is inheritance?**
   a) A way to create new objects based on existing ones
   b) A way to create new classes based on existing ones
   c) A way to create new methods based on existing ones
   d) A way to create new properties based on existing ones
   Answer: b) A way to create new classes based on existing ones

   What is a superclass in inheritance?
   a) A class that inherits from another class
   b) A class that is inherited by another class
   c) A class that has no parent class
   d) A class that is created from scratch
   Answer: b) A class that is inherited by another class

   What is a subclass in inheritance?
   a) A class that inherits from another class
   b) A class that is inherited by another class
   c) A class that has no parent class
   d) A class that is created from scratch
   Answer: a) A class that inherits from another class

   What is the purpose of inheritance in object-oriented programming?
   a) To create new objects
   b) To reduce code redundancy
   c) To create new methods
   d) To create new properties
   Answer: b) To reduce code redundancy

   Which of the following keywords is used to denote inheritance in Java?
   a) extends
   b) implements
   c) interface
   d) abstract
   Answer: a) extends

   Which type of inheritance allows a subclass to inherit from multiple parent classes?
   a) Single inheritance
   b) Multiple inheritance
   c) Multilevel inheritance
   d) Hierarchical inheritance
   Answer: b) Multiple inheritance

   Inheritance promotes which principle of object-oriented programming?
   a) Encapsulation
   b) Polymorphism
   c) Abstraction
   d) All of the above
   Answer: d) All of the above

   Which access modifier in Java allows a subclass to access a protected member of its

parent class?
a) public
b) private
c) protected
d) default
Answer: c) protected

What is method overriding in inheritance?
a) Creating a new method in the subclass with the same name as a method in the parent class
b) Creating a new method in the parent class with the same name as a method in the subclass
c) Renaming a method in the subclass to match a method in the parent class
d) Hiding a method in the parent class from the subclass
Answer: a) Creating a new method in the subclass with the same name as a method in the parent class

What is method overloading in inheritance?
a) Creating a new method in the subclass with the same name as a method in the parent class
b) Creating a new method in the parent class with the same name as a method in the subclass
c) Renaming a method in the subclass to match a method in the parent class
d) Creating a new method in the subclass with the same name but different parameters as a method in the parent class
Answer: d) Creating a new method in the subclass with the same name but different parameters as a method in the parent class

Lec 5 - SIMPLE ASSOCIATION

1. **In simple association, how many classes are involved?**
   A) One
   B) Two
   C) Three
   D) Four
   Answer: B) Two

   How is simple association represented in UML diagrams?
   A) As a solid line
   B) As a dotted line
   C) As a dashed line
   D) As a double line
   Answer: A) As a solid line

   In simple association, which class is usually responsible for initiating the interaction?
   A) The class on the left side of the association
   B) The class on the right side of the association
   C) Both classes can initiate the interaction
   D) Neither class can initiate the interaction
   Answer: A) The class on the left side of the association

   Which of the following is an example of simple association?
   A) A car has an engine
   B) A car is a vehicle
   C) A car belongs to a person
   D) A car drives on a road
   Answer: A) A car has an engine

   Which of the following is true about simple association?
   A) It is always one-way
   B) It is always bidirectional
   C) It can be one-way or bidirectional
   D) It is always represented by an arrow
   Answer: C) It can be one-way or bidirectional

   What is the role of the class on the right side of the association in simple association?
   A) To provide functionality to the class on the left side
   B) To receive functionality from the class on the left side
   C) To initiate the interaction with the class on the left side
   D) To define the type of the association
   Answer: B) To receive functionality from the class on the left side

   Which of the following is not an example of simple association?
   A) A dog has a tail
   B) A book belongs to a library
   C) A bird can fly
   D) A student attends a class
   Answer: C) A bird can fly

   Which of the following is not true about simple association?
   A) It is a type of relationship between classes

B) It enables communication and collaboration between classes
C) It is always one-to-one
D) It can be one-way or bidirectional
Answer: C) It is always one-to-one

What is the difference between simple association and inheritance?
A) Inheritance involves a parent-child relationship, while simple association does not
B) Simple association involves a parent-child relationship, while inheritance does not
C) Simple association enables communication and collaboration between classes, while inheritance does not
D) Inheritance and simple association are the same thing
Answer: A) Inheritance involves a parent-child relationship, while simple association does not

Which of the following is an example of bidirectional simple association?
A) A person has a car
B) A car belongs to a person
C) A teacher teaches a class
D) A student attends a class
Answer: A) A person has a car and a car belongs to a person can both be examples of bidirectional simple association.

Lec 6 - CLASS COMPATIBILITY

1. **What is class compatibility in object-oriented programming?**
   a) The ability of classes to be used interchangeably
   b) The ability of one class to use objects of another class without errors
   c) The ability of classes to be inherited from each other
   d) The ability of classes to have the same name
   Answer: b) The ability of one class to use objects of another class without errors

   Which of the following can affect class compatibility?
   a) Inheritance
   b) Interfaces
   c) Method signatures
   d) All of the above
   Answer: d) All of the above

   What is the difference between static and dynamic class compatibility?
   a) Static compatibility is checked at compile time, while dynamic compatibility is checked at runtime
   b) Static compatibility is checked at runtime, while dynamic compatibility is checked at compile time
   c) There is no difference between static and dynamic class compatibility
   d) Static compatibility is a type of inheritance, while dynamic compatibility is a type of polymorphism
   Answer: a) Static compatibility is checked at compile time, while dynamic compatibility is checked at runtime

   Which of the following is an example of static class compatibility?
   a) An object of a subclass being passed to a method that expects an object of the superclass
   b) An object of a class implementing an interface being passed to a method that expects an object of the interface
   c) An object of a class with a compatible method signature being passed to a method
   d) None of the above
   Answer: c) An object of a class with a compatible method signature being passed to a method

   Which of the following is an example of dynamic class compatibility?
   a) A superclass reference being used to refer to an object of a subclass
   b) An interface reference being used to refer to an object of a class implementing the interface
   c) A method being overridden in a subclass
   d) All of the above
   Answer: a) A superclass reference being used to refer to an object of a subclass

   What is type checking in relation to class compatibility?
   a) The process of checking if a variable or object is of a specific type
   b) The process of checking if two classes are compatible
   c) The process of checking if an object can be cast to a specific type
   d) The process of checking if a method signature is compatible with a class
   Answer: a) The process of checking if a variable or object is of a specific type

   What is casting in relation to class compatibility?
   a) The process of checking if a variable or object is of a specific type

b) The process of checking if two classes are compatible
c) The process of converting an object to a different type
d) The process of checking if a method signature is compatible with a class
Answer: c) The process of converting an object to a different type

What happens if an object is cast to an incompatible type?
a) An exception is thrown at runtime
b) The object is automatically converted to the compatible type
c) The object remains unchanged
d) None of the above
Answer: a) An exception is thrown at runtime

Which of the following is an example of a type casting error?
a) Casting an object of a subclass to a superclass type
b) Casting an object of an interface implementation to an interface type
c) Casting an object to a different type than it was created as
d) All of the above
Answer: d) All of the above

Why is class compatibility important in software development?
a) It ensures that classes can work together effectively
b) It makes software systems easier to maintain
c) It reduces the likelihood of errors and bugs
d) All

Lec 7 - CLASS

1. **What does CLASS stand for?**
   a) Classroom Assessment and Scoring System
   b) Children's Learning and Support System
   c) Classroom Analysis and Skills System
   d) Creative Learning and Assessment Strategies
   Answer: a) Classroom Assessment and Scoring System

   How many domains does CLASS evaluate?
   a) 2
   b) 3
   c) 4
   d) 5
   Answer: c) 4

   Which of the following is not a domain evaluated by CLASS?
   a) Emotional Support
   b) Classroom Organization
   c) Instructional Support
   d) Student Performance
   Answer: d) Student Performance

   What is the purpose of CLASS?
   a) To measure and improve the quality of teacher-student interactions in early childhood education settings
   b) To assess student academic achievement
   c) To evaluate teacher performance
   d) To promote student engagement in the classroom
   Answer: a) To measure and improve the quality of teacher-student interactions in early childhood education settings

   Who developed the CLASS tool?
   a) The National Institute for Early Education Research
   b) The National Association for the Education of Young Children
   c) The University of Virginia
   d) The American Educational Research Association
   Answer: c) The University of Virginia

   How many dimensions are evaluated within the Emotional Support domain of CLASS?
   a) 2
   b) 3
   c) 4
   d) 5
   Answer: b) 3

   Which dimension within the Classroom Organization domain of CLASS evaluates how well teachers manage student behavior?
   a) Behavior Management
   b) Productivity

c) Instructional Learning Formats
d) Concept Development
Answer: a) Behavior Management

Which dimension within the Instructional Support domain of CLASS evaluates how well teachers promote higher-order thinking skills?
a) Quality of Feedback
b) Cognitive Complexity
c) Language Modeling
d) Concept Development
Answer: b) Cognitive Complexity

How is CLASS typically used in early childhood education settings?
a) To evaluate student academic achievement
b) To assess teacher performance
c) To identify areas for improvement in teacher-student interactions
d) To promote student engagement in the classroom
Answer: c) To identify areas for improvement in teacher-student interactions

Which of the following is not a benefit of using CLASS in early childhood education settings?
a) Provides feedback to teachers to improve their practice
b) Supports professional development
c) Enhances student academic achievement
d) Promotes a positive classroom climate
Answer: c) Enhances student academic achievement

Lec 8 - MEMBER FUNCTIONS

1. **What is the primary purpose of member functions in object-oriented programming?**
   A. To declare variables
   B. To initialize objects
   C. To perform operations on objects
   D. To create new objects
   Answer: C

   Which keyword is used to define a member function in C++?
   A. class
   B. function
   C. method
   D. this
   Answer: C

   Which type of member function can access private data members of a class?
   A. Public
   B. Private
   C. Protected
   D. Friend
   Answer: B

   Which type of member function does not have access to the this pointer?
   A. Static
   B. Virtual
   C. Inline
   D. Friend
   Answer: A

   Which keyword is used to call a member function on an object in C++?
   A. new
   B. delete
   C. this
   D. dot operator (.)
   Answer: D

   Which type of member function is used to initialize an object with a set of default values?
   A. Constructor
   B. Destructor
   C. Virtual function
   D. Operator function
   Answer: A

   Which type of member function is called automatically when an object is destroyed?
   A. Constructor
   B. Destructor
   C. Virtual function
   D. Operator function
   Answer: B

   Which type of member function can be called without creating an object of the class?
   A. Constructor

B. Destructor
C. Static function
D. Friend function
Answer: C

Which keyword is used to access a data member of a class within a member function?
A. private
B. public
C. protected
D. this
Answer: D

Which type of member function is used to overload operators such as +, -, *, and /?
A. Constructor
B. Destructor
C. Virtual function
D. Operator function
Answer: D

Lec 9 - SHALLOW COPY

1. **What is shallow copy?**
   a) A copy of the data itself
   b) A copy of the pointers or references to the data members
   c) A copy of the entire object
   Answer: b

   What happens when changes are made to the data in a shallow copy?
   a) The original object is also changed
   b) The original object remains unchanged
   c) The new object is destroyed
   Answer: a

   Which type of copy is a shallow copy?
   a) A deep copy
   b) A partial copy
   c) A pointer copy
   Answer: c

   What is the purpose of shallow copying?
   a) To create a new object with the same data as the original object
   b) To create a copy of the original object
   c) To create a reference to the original object
   Answer: c

   Can a shallow copy be modified without affecting the original object?
   a) Yes
   b) No
   Answer: b

   Which programming languages support shallow copying by default?
   a) Java
   b) Python
   c) C++
   Answer: c

   What is the difference between a shallow copy and a deep copy?
   a) A shallow copy only copies pointers, while a deep copy copies the entire object
   b) A shallow copy copies the entire object, while a deep copy only copies pointers
   c) There is no difference between the two
   Answer: a

   Is it possible to create a shallow copy manually in C++?
   a) Yes
   b) No
   Answer: a

   What happens if a shallow copy is deleted before the original object?
   a) The original object is deleted

b) The new object is deleted
c) Both the original and new objects are deleted
Answer: b

Can a shallow copy be used to create an independent copy of an object?
a) Yes
b) No
Answer: b

Lec 10 - USES OF THIS POINTER

1. **In object-oriented programming, what is the "this" pointer?**
   a) A reference to the object that is currently being operated on
   b) A reference to the parent object
   c) A reference to the child object
   d) A reference to the base class
   Answer: a) A reference to the object that is currently being operated on

   What is the main use of the "this" pointer?
   a) To access member variables or functions of the current object
   b) To access member variables or functions of another object
   c) To create a new object
   d) To destroy an object
   Answer: a) To access member variables or functions of the current object

   Can the "this" pointer be used to access member variables of other objects of the same class?
   a) Yes
   b) No
   Answer: b) No

   Can the "this" pointer be used to pass the object as an argument to another function?
   a) Yes
   b) No
   Answer: a) Yes

   Can the "this" pointer be used to return the object from a function?
   a) Yes
   b) No
   Answer: a) Yes

   What is the benefit of using the "this" pointer?
   a) It helps to differentiate between multiple objects of the same class
   b) It helps to create new objects
   c) It helps to destroy objects
   d) It helps to access variables of other classes
   Answer: a) It helps to differentiate between multiple objects of the same class

   Is the "this" pointer supported by all programming languages?
   a) Yes
   b) No
   Answer: b) No

   In C++, what is the syntax for using the "this" pointer to access a member variable?
   a) this.memberVariable
   b) memberVariable.this
   c) this->memberVariable
   d) memberVariable->this
   Answer: c) this->memberVariable

   Can the "this" pointer be used outside of a member function?
   a) Yes

b) No
Answer: b) No

Is the "this" pointer a constant or a variable?
a) Constant
b) Variable
Answer: a) Constant

Lec 11 - USAGE EXAMPLE OF CONSTANT MEMBER FUNCTIONS

1. **In which scenario would you use a constant member function?**
   a. When you want to modify the object
   b. When you want to ensure that the object cannot be modified
   c. When you want to improve performance
   d. Both b and c
   Answer: d

   Which of the following is an example of a scenario where constant member functions would be useful?
   a. Implementing a class that represents a car
   b. Implementing a class that represents a mathematical vector
   c. Implementing a class that represents a text editor
   d. Implementing a class that represents a video game character
   Answer: b

   Can a constant member function modify the state of the object it is called on?
   a. Yes
   b. No
   Answer: b

   What is the benefit of using constant member functions?
   a. They allow you to modify the object
   b. They improve performance
   c. They ensure that the object cannot be modified
   d. They allow you to access private member variables
   Answer: c

   Which keyword is used to declare a member function as constant?
   a. const
   b. static
   c. virtual
   d. volatile
   Answer: a

   Which of the following is an example of a constant member function for a class representing a mathematical vector?
   a. void setX(double x)
   b. double getX() const
   c. double length()
   d. void normalize()
   Answer: b

   What is the purpose of a constant member function for a class representing a mathematical vector?
   a. To modify the vector
   b. To return the length of the vector
   c. To normalize the vector
   d. To ensure that the vector cannot be modified
   Answer: d

   Which of the following is an example of a scenario where constant member functions

would not be useful?
a. Implementing a class that represents a bank account
b. Implementing a class that represents a calendar event
c. Implementing a class that represents a temperature sensor
d. Implementing a class that represents a musical instrument
Answer: d

What is the return type of a constant member function?
a. void
b. int
c. double
d. Depends on the implementation
Answer: d

Can you call a non-constant member function from a constant member function?
a. Yes
b. No
Answer: b

Lec 12 - ACCESSING STATIC DATA MEMBER

1. **What is a static data member in C++?**
   a) A data member that can only be accessed by member functions
   b) A data member that can be accessed by any function or method within the class
   c) A data member that is unique to each instance of a class
   d) A data member that is declared using the const keyword
   Answer: b

   How is a static data member declared in C++?
   a) Using the const keyword
   b) Using the static keyword
   c) Using the public keyword
   d) Using the friend keyword
   Answer: b

   How is a static data member accessed in C++?
   a) Using the object name followed by the dot operator
   b) Using the object name followed by the arrow operator
   c) Using the class name followed by the dot operator
   d) Using the class name followed by the arrow operator
   Answer: c

   Which of the following statements is true about static data members?
   a) They are unique to each instance of a class
   b) They can only be accessed by member functions
   c) They are shared among all objects of a class
   d) They are declared using the const keyword
   Answer: c

   What is the default value of a static data member in C++?
   a) 0
   b) 1
   c) Null
   d) Undefined
   Answer: a

   What is the scope of a static data member in C++?
   a) Global scope
   b) Local scope
   c) Class scope
   d) Namespace scope
   Answer: c

   What is the lifetime of a static data member in C++?
   a) Until the end of the program
   b) Until the end of the function in which it is declared
   c) Until the object is destroyed
   d) Until it is explicitly deleted
   Answer: a

   How many instances of a static data member are there in a class?
   a) One for each instance of the class

b) One for all instances of the class
c) One for each member function
d) None of the above
Answer: b

Which keyword is used to access a static data member outside the class in C++?
a) private
b) public
c) static
d) friend
Answer: c

Can a static data member be modified by a non-static member function?
a) Yes
b) No
Answer: a

Lec 13 - POINTER TO OBJECTS

1. **What is a pointer to an object in C++?**
   a) A variable that stores the value of an object
   b) A variable that stores the memory address of an object
   c) A variable that stores the size of an object
   d) A variable that stores the name of an object
   Answer: b) A variable that stores the memory address of an object.

   What is the syntax to declare a pointer to an object in C++?
   a) int ptr;
   b) obj pointer;
   c) obj *ptr;
   d) obj -> pointer;
   Answer: c) obj *ptr;

   How is the value of an object pointed to by a pointer accessed in C++?
   a) Using the * operator
   b) Using the & operator
   c) Using the -> operator
   d) Using the . operator
   Answer: a) Using the * operator.

   What is the purpose of using pointers to objects in C++?
   a) To dynamically allocate memory for objects
   b) To pass objects to functions by reference
   c) To manipulate objects indirectly
   d) All of the above
   Answer: d) All of the above.

   Can a pointer to an object be null in C++?
   a) Yes
   b) No
   Answer: a) Yes.

   What is the difference between a pointer to an object and a reference to an object in C++?
   a) A pointer can be null, while a reference cannot.
   b) A pointer can be reassigned to point to a different object, while a reference cannot.
   c) A pointer requires the * operator to access the object's value, while a reference does not.
   d) All of the above.
   Answer: d) All of the above.

   How is memory allocated for an object pointed to by a pointer in C++?
   a) Using the new operator
   b) Using the delete operator
   c) Using the malloc function
   d) Using the free function
   Answer: a) Using the new operator.

   What is the purpose of the -> operator in C++?
   a) To access a member of a class or structure pointed to by a pointer

b) To declare a pointer to an object
c) To declare a reference to an object
d) None of the above
Answer: a) To access a member of a class or structure pointed to by a pointer.

How can a pointer to an object be passed to a function in C++?
a) By value
b) By reference
c) By const reference
d) All of the above
Answer: b) By reference.

What is a dangling pointer in C++?
a) A pointer that points to a null object
b) A pointer that points to an object that has been deleted or deallocated
c) A pointer that points to a new object
d) A pointer that points to an object that has not been initialized
Answer: b) A pointer that points to an object that has been deleted or deallocated.

Lec 14 - COMPOSITION

1. **What is composition in object-oriented programming?**
   a) A way of creating complex objects by combining simpler objects or data types
   b) A way of inheriting properties and behaviors from a parent class
   c) A way of creating objects from a template or blueprint
   Answer: a

   In composition, which class contains an instance of another class as a member variable?
   a) The parent class
   b) The child class
   c) Both classes
   Answer: b

   Which symbol is used to denote composition in a UML class diagram?
   a) A solid line with an arrow pointing to the contained class
   b) A dashed line with an arrow pointing to the contained class
   c) A solid line connecting the two classes
   Answer: b

   What is the purpose of using composition in object-oriented programming?
   a) To create objects with complex behavior
   b) To simplify the implementation of inheritance
   c) To create objects with a strong is-a relationship
   Answer: a

   How does composition differ from inheritance?
   a) Composition is a type of association between classes, while inheritance is a way of inheriting properties and behaviors from a parent class
   b) Composition is used to create a strong is-a relationship between classes, while inheritance is used to combine behaviors from multiple classes
   c) Composition and inheritance are identical concepts
   Answer: a

   Which keyword is used to define a composition relationship in C++?
   a) extends
   b) implements
   c) none
   Answer: c

   What happens to the contained object when the containing object is destroyed in composition?
   a) The contained object is destroyed automatically
   b) The contained object remains alive
   c) It depends on how the composition is implemented
   Answer: a

   Can a class have multiple instances of another class as member variables in composition?
   a) Yes

b) No
c) It depends on the programming language being used
Answer: a

Which of the following is an example of composition in real-world objects?
a) A car's engine and transmission
b) A car and a truck
c) A car's tires and brakes
Answer: a

Which of the following is not a benefit of using composition in object-oriented programming?
a) Encapsulation of behavior and data
b) Code reuse
c) Simplified implementation of inheritance
Answer: c

Lec 15 - AGGREGATION

1. **What is aggregation in object-oriented programming?**
   a. A type of inheritance
   b. A way of creating complex objects by combining simpler objects
   c. A type of association between classes where one class contains a collection of another class's objects
   Answer: c

   Can the contained objects in aggregation exist independently of the containing object?
   a. Yes
   b. No
   Answer: a

   How is aggregation represented in a UML class diagram?
   a. With a solid line and an arrow pointing to the contained class
   b. With a dashed line and an arrow pointing to the contained class
   c. With a dotted line and an arrow pointing to the contained class
   Answer: b

   What is the purpose of using aggregation in object-oriented programming?
   a. To create complex objects by combining simpler objects
   b. To inherit properties and behaviors from a parent class
   c. To encapsulate behavior and data
   Answer: a

   Can a class have multiple instances of another class as member variables in aggregation?
   a. Yes
   b. No
   Answer: a

   How does aggregation differ from composition?
   a. In aggregation, the contained objects cannot exist independently of the containing object
   b. In composition, the contained objects can exist independently of the containing object
   c. There is no difference between aggregation and composition
   Answer: b

   Can the contained objects be shared among multiple containing objects in aggregation?
   a. Yes
   b. No
   Answer: a

   What happens to the contained objects when the containing object is destroyed in aggregation?
   a. The contained objects are automatically destroyed
   b. The contained objects continue to exist independently of the containing object
   c. It depends on the implementation
   Answer: b

   How does aggregation support code reuse?
   a. By allowing for the creation of complex objects by combining simpler objects

b. By inheriting properties and behaviors from a parent class
c. By encapsulating behavior and data
Answer: a

What are some real-world examples of aggregation?
a. A car's engine and transmission
b. A house's rooms and furniture
c. A human's body parts
Answer: b

Lec 16 - OPERATOR OVERLOADING

1. **Which keyword is used to overload operators in C++?**
   a. override
   b. overload
   c. friend
   d. operator
   Answer: d

   Which of the following operators cannot be overloaded in C++?
   a. +
   b. &&
   c. =
   d. ->
   Answer: d

   Which of the following is a unary operator?
   a. +
   b. ++
   c. <<
   d. -
   Answer: d

   Which of the following is used to overload the subscript operator?
   a. []
   b. ()
   c. {}
   d. <>
   Answer: a

   Which of the following is used to overload the insertion operator for cout?
   a. <<
   b. >>
   c. ::
   d. ->
   Answer: a

   Which of the following is used to overload the prefix increment operator?
   a. ++
   b. --
   c. ==
   d. /
   Answer: a

   Which of the following operators has the highest **precedence**?
   a. ||
   b. *
   c. !=
   d. +
   Answer: b

   Which of the following is a binary operator?
   a. !

b. ^
c. ~
d. &&
Answer: b

Which of the following is a comparison operator?
a. !
b. +
c. <
d. &
Answer: c

Which of the following is used to overload the addition operator?
a. &&
b. |
c. ^
d. +
Answer: d

Lec 17 - OVERLOADING ASSIGNMENT OPERATOR

1. **What is the syntax for overloading the assignment operator in C++?**
   A) operator = ()
   B) operator ()
   C) operator ()
   D) operator += ()
   Answer: A

   Which of the following is a valid signature for an overloaded assignment operator that takes a reference to the class object as a parameter?
   A) MyClass operator=(MyClass& obj)
   B) void operator=(const MyClass& obj)
   C) MyClass& operator=(const MyClass& obj)
   D) void operator=(MyClass& obj)
   Answer: C

   What is the return type of an overloaded assignment operator?
   A) void
   B) the class type
   C) int
   D) bool
   Answer: B

   How is the overloaded assignment operator invoked in C++?
   A) using the = operator
   B) using the copy constructor
   C) using the constructor
   D) using the destructor
   Answer: A

   What is the purpose of overloading the assignment operator in C++?
   A) to allow objects of a class to be assigned values using the = operator
   B) to allow objects of a class to be compared using the == operator
   C) to allow objects of a class to be initialized using the constructor
   D) to allow objects of a class to be destroyed using the destructor
   Answer: A

   Which of the following is true about the copy constructor and the assignment operator?
   A) they both take a reference to the class object as a parameter
   B) they both return a reference to the class object
   C) they both perform a shallow copy of the object's data members
   D) they both perform a deep copy of the object's data members
   Answer: D

   What is the difference between the copy constructor and the assignment operator?
   A) the copy constructor creates a new object, while the assignment operator modifies an existing object
   B) the copy constructor takes a const reference to the object, while the assignment operator takes a non-const reference

C) the copy constructor performs a deep copy of the object, while the assignment operator performs a shallow copy
D) the copy constructor returns a reference to the object, while the assignment operator returns void
Answer: A

How do you avoid issues with self-assignment when overloading the assignment operator?
A) by checking for self-assignment using the == operator
B) by using a copy constructor to create a new object and then assigning it to the existing object
C) by checking for self-assignment using the this pointer
D) by using a swap function to swap the contents of the object with a temporary object
Answer: D

Which of the following is a common practice when overloading the assignment operator?
A) returning a copy of the object from the function
B) returning a reference to the object from the function
C) using dynamic memory allocation to perform a deep copy of the object's data members
D) using the default implementation of the operator provided by the compiler
Answer: C

Which of the following is true about the return type of the overloaded assignment operator?
A) it must be a built-in type such as int or bool
B) it can be any user-defined type
C) it must be the same type as the class being overloaded
D) it can be a different type from the class being overloaded
Answer: C

Lec 18 - SELF ASSIGNMENT PROBLEM

1. **What is self-assignment?**
   a) Assigning a pointer to a different object
   b) Assigning an object to itself
   c) Assigning a value to a constant variable
   d) Assigning a value to a variable of a different data type
   Answer: b

   What can happen if self-assignment is not handled properly?
   a) Memory leaks
   b) Undefined behavior
   c) Corruption of the object's data
   d) All of the above
   Answer: d

   Which operator is commonly affected by the self-assignment problem?
   a) Comparison operator
   b) Unary operator
   c) Binary operator
   d) Assignment operator
   Answer: d

   What is a common technique for handling self-assignment in the assignment operator?
   a) Copying the object to a temporary object before performing the copy
   b) Checking if the object being assigned is the same as the original object before performing the copy
   c) Swapping the object with a copy of itself
   d) None of the above
   Answer: b

   What is the purpose of handling self-assignment in the assignment operator?
   a) To avoid memory leaks
   b) To prevent undefined behavior
   c) To ensure proper functioning of the program
   d) All of the above
   Answer: d

   Which of the following is a potential issue that can arise from self-assignment?
   a) Data corruption
   b) Memory leaks
   c) Undefined behavior
   d) All of the above
   Answer: d

   Why is it important to properly handle self-assignment in the assignment operator?
   a) To prevent crashes
   b) To avoid undefined behavior
   c) To optimize program performance
   d) All of the above
   Answer: b

   How can self-assignment be checked in the assignment operator?

b) Using a conditional statement
c) Using a loop
d) None of the above
Answer: b

Which of the following is an example of self-assignment?
a) a = b
b) a = a
c) a = &b
d) a = *b
Answer: b

What can happen if self-assignment is not handled properly in a program?
a) The program may crash
b) The program may behave unpredictably
c) The program may run slower than expected
d) All of the above
Answer: d

Lec 19 - STREAM INSERTION OPERATOR

1. **What is the return type of the stream insertion operator (<<)?**
   a) void
   b) istream&
   c) ostream&
   d) string
   Answer: c) ostream&

   Which of the following is an example of overloading the stream insertion operator for a custom class?
   a) int x; cin >> x;
   b) cout << "Hello, World!" << endl;
   c) cout << obj;
   d) cin << obj;
   Answer: c) cout << obj;

   Which of the following is an example of using the stream insertion operator to output multiple values?
   a) cout << "The value of x is " << x;
   b) cout << "The sum of " << x << " and " << y << " is " << x + y;
   c) cout << "Enter a value: ";
   d) cout << "The result is " << result << endl;
   Answer: b) cout << "The sum of " << x << " and " << y << " is " << x + y;

   Which of the following is a possible implementation of the stream insertion operator for a custom class?
   a) ostream& operator<<(ostream& out, MyClass obj) { /* implementation / }
   b) istream& operator<<(istream& in, MyClass obj) { / implementation / }
   c) MyClass& operator<<(MyClass obj) { / implementation / }
   d) void operator<<(MyClass obj) { / implementation / }
   Answer: a) ostream& operator<<(ostream& out, MyClass obj) { / implementation */ }

   Which of the following is a correct syntax for using the stream insertion operator to output an object?
   a) cout << MyClass;
   b) cout << object.MyClass;
   c) object << cout;
   d) cout << object;
   Answer: d) cout << object;

   Which of the following is a correct syntax for overloading the stream insertion operator for a custom class?
   a) void operator<<();
   b) void operator<<(ostream& out);
   c) void operator<<(ostream& out, MyClass obj);
   d) void operator<<(MyClass obj);
   Answer: c) void operator<<(ostream& out, MyClass obj);

   Which of the following is a correct implementation of the stream insertion operator for a

custom class that has private data members?
a) ostream& operator<<(ostream& out, MyClass obj) { out << obj.privateMember; }
b) ostream& operator<<(ostream& out, MyClass obj) { obj.privateMember << out; }
c) ostream& operator<<(ostream& out, MyClass obj) { obj.getPrivateMember() << out; }
d) None of the above.
Answer: c) ostream& operator<<(ostream& out, MyClass obj) { obj.getPrivateMember() << out; }

Which of the following is an example of using the stream insertion operator to output a literal value?
a) cout << "Hello, World!";
b) cout << x;
c) cin >> x;
d) cout << "Enter a value: ";
Answer: a) cout << "Hello, World!";

Which of the following is a correct implementation of the stream insertion operator for a custom class that has a non-static data member?
a) ostream& operator<<(ostream& out, MyClass obj) { obj.dataMember << out; }
b) ostream& operator<<(ostream& out, MyClass obj) { obj.dataMember >> out; }
c) ostream& operator<<(ostream& out, MyClass obj) { out << obj.dataMember; }
d) None of the above.
Answer: c) ostream& operator<<(ostream& out, MyClass obj) { out << obj.dataMember; }

Which of the following is an example of using the stream insertion operator

Lec 20 - SUBSCRIPT [] OPERATOR

1. **Which operator is used to access elements of an array or container class?**
   a) () operator
   b) {} operator
   c) [] operator
   d) -> operator
   Answer: c) [] operator

   What is the parameter type for an overloaded subscript operator function?
   a) int
   b) char
   c) string
   d) Depends on the type of the elements being accessed
   Answer: d) Depends on the type of the elements being accessed

   Which of the following is a valid use of the subscript operator?
   a) accessing the nth character of a string
   b) accessing the nth element of an array
   c) accessing the nth element of a vector
   d) all of the above
   Answer: d) all of the above

   What is the return type of the subscript operator function?
   a) void
   b) int
   c) char
   d) Depends on the type of the elements being accessed
   Answer: d) Depends on the type of the elements being accessed

   Which of the following is true regarding the subscript operator overloading?
   a) Only one overload of the subscript operator is allowed per class.
   b) The overload function must be a member function of the class.
   c) The overload function must be a friend function of the class.
   d) The overload function must take two parameters.
   Answer: b) The overload function must be a member function of the class.

   What is the purpose of subscript operator overloading?
   a) To provide a custom element access behavior for user-defined classes.
   b) To access private data members of a class.
   c) To perform arithmetic operations on array elements.
   d) None of the above.
   Answer: a) To provide a custom element access behavior for user-defined classes.

   Which of the following is a disadvantage of using the subscript operator?
   a) It can lead to out-of-bounds access.
   b) It is slower than pointer arithmetic.
   c) It cannot be used with containers like maps and sets.
   d) It cannot be overloaded for user-defined classes.
   Answer: a) It can lead to out-of-bounds access.

   Which of the following is true regarding the subscript operator overloading for a

container class?

a) The operator function must return a reference to the element being accessed.
b) The operator function must return a copy of the element being accessed.
c) The operator function must take a single parameter of type int.
d) The operator function is not allowed to modify the container.
Answer: a) The operator function must return a reference to the element being accessed.

Which of the following is a valid example of subscript operator overloading?

a) int operator[](int i);
b) void operator[](int i);
c) int& operator[](int i);
d) int* operator[](int i);
Answer: c) int& operator[](int i);

What happens if the subscript operator function returns a copy of the element being accessed?

a) The copy is returned by value and can be modified independently of the original element.
b) The copy is returned by reference and any modifications made to it will affect the original element.
c) The program will not compile.
d) None of the above.
Answer: a) The copy is returned by value and can be modified independently of the original element.

Lec 21 - BEHAVIOR OF ++ AND -- FOR PRE-DEFINED TYPES

What is the result of the following code snippet?

css

Copy code

int a = 5;

int b = a++;

a) a = 5, b = 6

b) a = 6, b = 5

c) a = 5, b = 5

d) a = 6, b = 6

Answer: a) a = 5, b = 6

What is the result of the following code snippet?

css

Copy code

int a = 5;

int b = ++a;

a) a = 5, b = 6

b) a = 6, b = 5

c) a = 5, b = 5

d) a = 6, b = 6

Answer: b) a = 6, b = 6

What is the result of the following code snippet?

css

Copy code

float a = 5.0;

float b = a++;

a) a = 5.0, b = 6.0

b) a = 6.0, b = 5.0

c) a = 5.0, b = 5.0

d) a = 6.0, b = 6.0

Answer: a) a = 6.0, b = 5.0

What is the result of the following code snippet?

css

Copy code

float a = 5.0;

float b = ++a;

a) a = 5.0, b = 6.0

b) a = 6.0, b = 5.0

c) a = 5.0, b = 5.0

d) a = 6.0, b = 6.0

Answer: b) a = 6.0, b = 6.0

What is the result of the following code snippet?

css

Copy code

char a = 'a';

char b = a++;

a) a = 'a', b = 'b'

b) a = 'b', b = 'a'

c) a = 'a', b = 'a'

d) a = 'b', b = 'b'

Answer: a) a = 'b', b = 'a'

What is the result of the following code snippet?

css

Copy code

char a = 'a';

char b = ++a;

a) a = 'a', b = 'b'

b) a = 'b', b = 'a'

c) a = 'a', b = 'a'

d) a = 'b', b = 'b'

Answer: b) a = 'b', b = 'b'

What is the result of the following code snippet?

css

Copy code

int a = 5;

int b = a-- + 3;

a) a = 5, b = 8

b) a = 4, b = 7

c) a = 5, b = 7

d) a = 4, b = 8

Answer: c) a = 4, b = 8

What is the result of the following code snippet?

css

Copy code

```
int a = 5;

int b = --a + 3;
```

a) a = 5, b = 7

b) a = 4, b = 7

c) a = 5, b = 6

d) a = 4, b = 6

Answer: b) a = 4, b = 6

Lec 22 - PRACTICAL IMPLEMENTATION OF INHERITANCE IN C

1. **Inheritance in C can be implemented using:**
   a) Structures and function pointers
   b) Classes and objects
   c) Inheritance keyword
   d) None of the above
   Answer: a) Structures and function pointers

   Which of the following is not a type of inheritance?
   a) Single inheritance
   b) Multiple inheritance
   c) Hierarchical inheritance
   d) Parallel inheritance
   Answer: d) Parallel inheritance

   The derived class inherits:
   a) All the properties and methods of the base class
   b) Only the properties of the base class
   c) Only the methods of the base class
   d) None of the above
   Answer: a) All the properties and methods of the base class

   What is the syntax to define a derived structure in C?
   a) struct Derived : Base {}
   b) struct Derived extends Base {}
   c) struct Derived : public Base {}
   d) struct Derived : private Base {}
   Answer: c) struct Derived : public Base {}

   Inheritance is used to:
   a) Achieve code reusability
   b) Encapsulate data
   c) Control access to data
   d) None of the above
   Answer: a) Achieve code reusability

   Which type of inheritance allows a derived class to inherit from multiple base classes?
   a) Single inheritance
   b) Multiple inheritance
   c) Hierarchical inheritance
   d) Hybrid inheritance
   Answer: b) Multiple inheritance

   Which keyword is used to call the constructor of the base class from the derived class constructor?
   a) super
   b) base
   c) this
   d) parent
   Answer: b) base

   Which type of inheritance involves creating a new class that inherits from a base class,

and then creating another class that inherits from the new class?
a) Single inheritance
b) Multiple inheritance
c) Hierarchical inheritance
d) Hybrid inheritance
Answer: c) Hierarchical inheritance

Which of the following is not a benefit of inheritance?
a) Code reusability
b) Improved maintainability
c) Reduced coupling
d) Increased code complexity
Answer: d) Increased code complexity

Which of the following is an example of polymorphism?
a) Overriding a method in a derived class
b) Calling a method from the base class
c) Inheriting properties from a base class
d) None of the above
Answer: a) Overriding a method in a derived class