1. What is inheritance in C++?
   1. A way to hide data
   2. A way to implement polymorphism
   3. **A way to create new classes from existing classes**
   4. A way to handle exceptions
2. Which keyword is used to derive a class from a base class in C++?
   1. base
   2. super
   3. this
   4. **class**
3. Which access specifier is used for private inheritance in C++?
   1. **private**
   2. protected
   3. public
   4. friend
4. In C++, when deriving a class from a base class, which access specifier grants the most visibility to the members of the base class in the derived class?
   1. private
   2. protected
   3. **public**
   4. friend
5. What is the default access specifier while inheriting a class in C++?
   1. **private**
   2. protected
   3. public
   4. friend
6. Which type of inheritance involves deriving a class from multiple base classes in C++?
   1. Hierarchical Inheritance
   2. Single Inheritance
   3. **Multiple Inheritance**
   4. Multilevel Inheritance
7. What is the keyword used to call the base class constructor from the derived class constructor?
   1. base()
   2. this()
   3. **super()**
   4. parent()
8. In C++, can a derived class access private members of the base class?
   1. Yes
   2. **No**
9. What is the function used to check if a class is derived from another class in C++?
   1. **is\_base\_of()**
   2. derived\_from()
   3. is\_derived\_from()
   4. has\_base()
10. Which type of inheritance involves a class being derived from itself?
    1. Hierarchical Inheritance
    2. Single Inheritance
    3. Multiple Inheritance
    4. **Recursive Inheritance**
11. If a class 'B' is derived from class 'A', and class 'C' is derived from class 'B', which constructor is called first when creating an object of class 'C'?
    1. **Class 'A' constructor**
    2. Class 'B' constructor
    3. Class 'C' constructor
    4. Depends on the order of derivation
12. What is the use of the 'virtual' keyword in inheritance?
    1. It allows private members to be accessed in the derived class
    2. It ensures that only one instance of the base class is created
    3. **It enables late binding and polymorphism**
    4. It prevents the derived class from inheriting certain members
13. In C++, which access specifier is used when deriving a class from a base class to ensure that the public members of the base class remain public in the derived class?
    1. private
    2. protected
    3. **public**
    4. friend
14. What is the syntax to inherit a class 'B' privately from class 'A' in C++?
    1. class B : A;
    2. class B - A;
    3. **class B : private A;**
    4. class B private A;
15. When is the destructor of a base class called when using inheritance in C++?
    1. After the destructor of the derived class
    2. **Before the destructor of the derived class**
    3. It is not automatically called, you have to call it explicitly
    4. At the same time as the destructor of the derived class
16. Which type of inheritance involves creating a chain of classes with each derived class being a base class for the next class in the chain?
    1. Hierarchical Inheritance
    2. Single Inheritance
    3. Multiple Inheritance
    4. **Multilevel Inheritance**
17. In C++, can a derived class have multiple base classes of the same name and type?
    1. Yes
    2. **No**
18. What is the order of constructor and destructor execution in a multiple inheritance scenario in C++?
    1. **Constructor: Base classes in the order of derivation, Destructor: Derived class, Base classes in reverse order of derivation**
    2. Constructor: Derived class, Base classes in reverse order of derivation, Destructor: Base classes in the order of derivation
    3. Constructor: Base classes in reverse order of derivation, Destructor: Base classes in the order of derivation, Derived class
    4. Constructor: Derived class, Destructor: Base classes in the order of derivation, Base classes in reverse order of derivation
19. In C++, which access specifier is used when deriving a class from a base class to ensure that the protected members of the base class remain protected in the derived class?
    1. private
    2. **protected**
    3. public
    4. friend
20. What is the maximum number of levels of inheritance that C++ allows?
    1. 10
    2. 127
    3. 255
    4. **Unlimited**
21. In C++, what is the difference between "protected" and "private" access specifiers when inheriting a class?
    1. Both provide the same level of visibility to the derived class
    2. **"protected" allows access to derived classes, but "private" does not**
    3. "private" allows access to derived classes, but "protected" does not
    4. "protected" members can be accessed in the base class, but "private" members cannot
22. What is hybrid inheritance in C++?
    1. **The combination of single and multiple inheritances**
    2. The combination of public and private inheritance
    3. The combination of hierarchical and multilevel inheritances
    4. The combination of virtual and non-virtual inheritance
23. Can a C++ class be derived from multiple base classes using private inheritance?
    1. **Yes**
    2. No
24. Which type of inheritance involves creating a tree-like structure with a single root class and multiple derived classes?
    1. **Hierarchical Inheritance**
    2. Single Inheritance
    3. Multiple Inheritance
    4. Multilevel Inheritance
25. What is the diamond problem in C++?
    1. A common issue when using virtual functions with multiple inheritance
    2. An issue caused by circular dependencies in header files
    3. A conflict that arises when two derived classes have the same name
    4. **A problem caused by ambiguous access to a member in a multiple inheritance scenario**
26. What is the use of the 'using' keyword in C++ when dealing with inheritance?
    1. **It allows using base class members directly in the derived class**
    2. It specifies which constructor to use in the derived class
    3. It allows renaming derived class members
    4. It enables late binding and virtual function calls
27. In C++, can a derived class override a base class's constructor?
    1. Yes
    2. **No**
28. What is the purpose of the 'protected' access specifier in C++?
    1. To restrict access to members within the same class only
    2. **To make members accessible in derived classes but not in other parts of the program**
    3. To provide full access to all parts of the program
    4. To enable friend classes to access private members
29. What happens when a class is derived privately from a base class in C++?
    1. **All members of the base class become private in the derived class**
    2. All members of the base class become protected in the derived class
    3. All members of the base class become public in the derived class
    4. The base class becomes a friend of the derived class
30. In C++, can a derived class inherit the constructor of its base class?
    1. **Yes**
    2. No
31. What is hybrid inheritance?
    1. A type of inheritance that involves multiple base classes and multiple derived classes.
    2. A type of inheritance where a class can inherit from both another class and an interface.
    3. **A combination of multiple inheritance and single inheritance.**
    4. A type of inheritance that only involves one base class and multiple derived classes.
32. In C++, how is hierarchical inheritance achieved?
    1. By inheriting from multiple base classes.
    2. **By inheriting from a single base class.**
    3. By inheriting from more than one derived class.
    4. By inheriting from multiple unrelated classes.
33. Which inheritance type is prone to the "diamond problem" in C++?
    1. Hybrid inheritance
    2. Hierarchical inheritance
    3. **Multiple inheritance**
    4. Single inheritance
34. The "diamond problem" in C++ arises due to:
    1. Inheriting from multiple base classes.
    2. Using virtual functions in the base class.
    3. Having more than one derived class.
    4. **Ambiguity in method resolution with multiple inheritance.**
35. In hybrid inheritance, a class can inherit from:
    1. Only one base class and one derived class.
    2. **Two or more unrelated base classes.**
    3. One base class and multiple derived classes.
    4. Only two base classes.
36. Which access specifier is used by default in C++ for a class declared using class keyword?
    1. public
    2. **private**
    3. protected
    4. friend
37. In C++, which keyword is used to inherit publicly from a base class?
    1. **public**
    2. inherit
    3. extends
    4. using
38. When using hierarchical inheritance, how many classes are derived from a single base class?
    1. One
    2. Two
    3. Three
    4. **It depends on the design.**
39. What is the primary advantage of using hierarchical inheritance?
    1. **Code reusability**
    2. Method overriding
    3. Data encapsulation
    4. Multiple inheritance
40. In C++, when a class inherits privately from a base class, which members of the base class are accessible in the derived class?
    1. Public members only
    2. Protected members only
    3. **Private members only**
    4. All members
41. The ambiguity problem in multiple inheritance occurs when:
    1. Two or more base classes have the same member variables.
    2. **Two or more base classes have the same member functions.**
    3. A derived class has multiple constructors.
    4. The derived class is inherited privately from the base class.
42. Which keyword is used to resolve ambiguity in multiple inheritance?
    1. super
    2. **base**
    3. this
    4. virtual
43. Which access specifier in C++ allows the derived class to access the public members of the base class?
    1. public
    2. private
    3. **protected**
    4. friend
44. What is the main advantage of multiple inheritance in C++?
    1. Improved code readability
    2. Easier class hierarchy management
    3. **Enhanced code reusability**
    4. Reduced memory consumption
45. When does ambiguity occur in hierarchical inheritance?
    1. When two or more derived classes have the same method names.
    2. **When two or more base classes have the same method names.**
    3. When a class inherits from multiple base classes.
    4. When a class is derived from a single base class.
46. How can ambiguity be resolved in multiple inheritance?
    1. By using the friend keyword
    2. By using the virtual keyword
    3. **By using the scope resolution operator (::)**
    4. By using the using keyword
47. Which inheritance type is used to model "is-a" relationships in C++?
    1. Multiple inheritance
    2. Hierarchical inheritance
    3. Hybrid inheritance
    4. **Single inheritance**
48. Which type of inheritance allows a class to inherit from multiple base classes?
    1. Single inheritance
    2. **Multiple inheritance**
    3. Hierarchical inheritance
    4. Hybrid inheritance
49. What is the main drawback of multiple inheritance in C++?
    1. **It increases code complexity.**
    2. It leads to memory leaks.
    3. It is not supported by the C++ standard.
    4. It can only be used with abstract classes.
50. In C++, can a class inherit from multiple base classes with the same name member function?
    1. Yes, as long as the member functions have the same return type.
    2. Yes, as long as the member functions have different return types.
    3. **No, it will cause ambiguity.**
    4. No, it is not allowed by the C++ compiler.
51. How can the diamond problem in C++ be resolved?
    1. **By using the virtual keyword when inheriting classes.**
    2. By avoiding multiple inheritance altogether.
    3. By using the scope resolution operator (::) to specify which version of the function to use.
    4. By using the friend keyword to grant access to the base class's members.
52. How many specifiers are used to derive a class?
    1. 1
    2. 2
    3. **3**
    4. 4
53. What is a virtual function in C++?
    1. Any member function of a class
    2. All functions that are derived from the base class
    3. All the members that are accessing base class data members
    4. **All the functions which are declared in the base class and is re-defined/overridden by the derived class**
54. Which is the correct syntax of declaring a virtual function?
    1. **virtual int func();**
    2. virtual int func(){};
    3. inline virtual func();
    4. inline virtual func(){};
55. Which statement is incorrect about virtual function.
    1. They are used to achieve runtime polymorphism
    2. **They are used to hide objects**
    3. Each virtual function declaration starts with the virtual keyword
    4. All of the mentioned
56. The concept of deciding which function to invoke during runtime is called \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
    1. late binding
    2. dynamic linkage
    3. static binding
    4. **both late binding and dynamic linkage**
57. What is a pure virtual function?
    1. A virtual function defined inside the base class
    2. **A virtual function that has no definition relative to the base class**
    3. A virtual function that is defined inside the derived class
    4. Any function that is made virtual
58. What is the order of Destructors call when the object of derived class B is declared, provided class B is derived from class A?
    1. Destructor of A followed by B
    2. **Destructor of B followed by A**
    3. Destructor of A only
    4. Destructor of B only
59. Which concept of OOPs is shown by Virtual Functions?
    1. Inheritance
    2. Encapsulation
    3. **Polymorphism**
    4. Abstraction
60. Pick the correct statement.
    1. Virtual function can have different names in the base and derived class
    2. Virtual function cannot be applied in Multiple Inheritance classes
    3. **Virtual function are different in definitions only**
    4. Virtual function does early bindings
61. Which keyword is used to inherit a class in C++?
    1. class
    2. private
    3. protected
    4. **public**
62. Which type of inheritance allows a class to inherit multiple base classes?
    1. Single inheritance
    2. **Multiple inheritance**
    3. Multilevel inheritance
    4. Hierarchical inheritance
63. Which constructor is called first in the inheritance hierarchy?
    1. **Base class constructor**
    2. Derived class constructor
    3. Both are called simultaneously
    4. None of the above
64. Inheritance in C++ supports the concept of:
    1. Code reusability
    2. Encapsulation
    3. Polymorphism
    4. **All of the above**
65. What is the keyword used to invoke the base class constructor from the derived class constructor?
    1. super
    2. **base**
    3. parent
    4. this
66. Which member function in the base class can be overridden in the derived class?
    1. **Public member functions**
    2. Private member functions
    3. Protected member functions
    4. All member functions can be overridden
67. Which keyword is used to denote that a member function in the derived class is intended to override a base class member function?
    1. **override**
    2. virtual
    3. final
    4. over
68. Which of the following is true regarding function overloading and function overriding in C++?
    1. **Function overloading is resolved at compile-time, while function overriding is resolved at runtime.**
    2. Function overloading is resolved at runtime, while function overriding is resolved at compile-time.
    3. Both function overloading and function overriding are resolved at compile-time.
    4. Both function overloading and function overriding are resolved at runtime.
69. Which destructor is called first in the inheritance hierarchy?
    1. Base class destructor
    2. **Derived class destructor**
    3. Both are called simultaneously
    4. None of the above
70. Can a derived class destructor call the base class destructor explicitly?
    1. Yes
    2. **No**
71. What happens when a derived class object is destroyed?
    1. **Derived class destructor is called first, followed by the base class destructor.**
    2. Base class destructor is called first, followed by the derived class destructor.
    3. Only the derived class destructor is called.
    4. Only the base class destructor is called.
72. Which access specifier is used for inheritance in C++ by default?
    1. public
    2. **private**
    3. protected
    4. It depends on the base class definition.
73. Which of the following is true about constructors in a derived class?
    1. A derived class cannot have its own constructor.
    2. A derived class constructor is always called explicitly from the base class constructor.
    3. A derived class constructor is always called implicitly from the base class constructor.
    4. **A derived class can have its own constructor, which may or may not call the base class constructor.**
74. Which type of inheritance involves a class being derived from its own base class?
    1. Single inheritance
    2. Multiple inheritance
    3. **Multilevel inheritance**
    4. Hierarchical inheritance
75. Can a derived class object access private members of the base class?
    1. Yes
    2. **No**
76. In which order are the constructors executed in a multiple inheritance scenario?
    1. From the most derived class to the base class
    2. **From the base class to the most derived class**
    3. In a random order
    4. The constructors are not executed in multiple inheritance.
77. Can a derived class have more than one constructor?
    1. **Yes**
    2. No
78. Which type of inheritance allows the reuse of code from multiple base classes without forming an is-a relationship?
    1. Single inheritance
    2. **Multiple inheritance**
    3. Multilevel inheritance
    4. Hierarchical inheritance
79. Which of the following is true about virtual functions in C++?
    1. A virtual function cannot be overridden.
    2. A virtual function must have a body in the base class.
    3. **A virtual function can be overridden in a derived class.**
    4. A virtual function can only be declared in the base class and defined in the derived class.
80. Which member function is automatically called when an object is created?
    1. **Constructor**
    2. Destructor
    3. Main function
    4. Overloaded operator function
81. When does the destructor of an object get called?
    1. When the object is created
    2. **When the object goes out of scope**
    3. When the object is assigned a new value
    4. When the object is passed by value to a function
82. What is the purpose of the virtual destructor in C++?
    1. To deallocate memory allocated using the 'new' keyword
    2. To ensure proper destruction of derived class objects
    3. **To avoid memory leaks in polymorphic class hierarchies**
    4. All of the above
83. Can constructors be inherited in C++?
    1. Yes
    2. **No**
84. Which of the following is not a valid access specifier in C++?
    1. public
    2. private
    3. protected
    4. **default**
85. In which order are the destructors executed in a multiple inheritance scenario?
    1. **From the most derived class to the base class**
    2. From the base class to the most derived class
    3. In a random order
    4. The destructors are not executed in multiple inheritance.
86. What is the purpose of a pure virtual function in C++?
    1. To declare an abstract base class
    2. **To force derived classes to implement the function**
    3. To provide a default implementation for a function
    4. None of the above
87. Can a derived class object be assigned to a base class object?
    1. **Yes**
    2. No
88. What happens when a base class pointer is assigned a derived class object?
    1. The assignment is not allowed in C++.
    2. **Only the base class portion of the derived class object can be accessed through the pointer.**
    3. The derived class portion of the object is sliced off, and only the base class portion is accessible through the pointer.
    4. Both the base class and derived class portions of the object can be accessed through the pointer.
89. Which concept allows you to reuse the written code?
    1. Encapsulation
    2. Abstraction
    3. **Inheritance**
    4. Polymorphism
90. Which of the following explains Polymorphism?
    1. int func(int, int); float func1(float, float);
    2. int func(int); int func(int);
    3. **int func(float); float func(int, int, char);**
    4. int func(); int new\_func();
91. Which of the following shows multiple inheritances?
    1. A->B->C
    2. A->B; A->C
    3. **A,B->C**
    4. B->A
92. Which design patterns benefit from the multiple inheritances?
    1. **Adapter and observer pattern**
    2. Code pattern
    3. Glue pattern
    4. None of the mentioned
93. What will be the order of execution of base class constructors in the following method of inheritance.class a: public b, public c {...};
    1. **b(); c(); a();**
    2. c(); b(); a();
    3. a(); b(); c();
    4. b(); a(); c();
94. class X, class Y and class Z are derived from class BASE. This is \_\_\_\_\_\_ inheritance.
    1. Multiple
    2. Multilevel
    3. **Hierarchical**
    4. Single
95. Reusability of the code can be achieved in CPP through \_\_\_\_\_\_ .
    1. Polymorphism
    2. Encapsulation
    3. **Inheritance**
    4. Both A and C
96. \_\_\_\_\_\_\_\_\_\_\_ inheritance may lead to duplication of inherited members from a "grandparent" base class.
    1. **Multipath**
    2. Multiple
    3. Multilevel
    4. Hierarchical
97. What is the syntax of inheritance of class?
    1. class name
    2. class name : access specifer
    3. **class name : access specifer class name**
    4. class name : access specifer class name
98. What is meant by containership?
    1. **class contains objects of other class types as its members**
    2. class contains objects of other class types as its objects
    3. both a & b
    4. none of the mentioned
99. If base class has constructor with arguments, then it is \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ for the derived class to have constructor and pass the arguments to base class constructor.
    1. Optional
    2. **Mandatory**
    3. Compiler dependent
    4. Error
100. In Multipath inheritance, in order to remove duplicate set of records in child class, we \_\_\_\_\_\_\_\_\_\_\_ .
     1. Write Virtual function in parent classes
     2. Write virtual functions is base class
     3. **Make base class as virtual base class**
     4. All of these
101. A member declared as ………….. is accessible by the member functions within its class and any class immediately derived from it.
     1. **protected**
     2. privates
     3. public
     4. friend