1. Which of the following is true about constructors in C++?
   1. They have a return type
   2. **They are used to create objects**
   3. They can be inherited
   4. They can be declared as private
2. What is a default constructor in C++?
   1. A constructor with no arguments
   2. **A constructor created by the compiler when no constructor is defined**
   3. A constructor with default arguments
   4. A constructor that initializes all member variables to their default values
3. Which of the following constructors is called when a derived class object is created?
   1. Base class constructor
   2. Derived class constructor
   3. **Both base class and derived class constructors**
   4. None of the above
4. How can you call the parameterized constructor of the base class from a derived class?
   1. Using the super() keyword
   2. Using the base() keyword
   3. **Using the constructor's name with arguments**
   4. It is called automatically
5. What will happen if you do not define any constructors in a C++ class?
   1. It will result in a compilation error
   2. **The compiler will generate a default constructor**
   3. The program will not compile
   4. The program will throw a runtime error
6. What is the purpose of a destructor in C++?
   1. To create objects
   2. To allocate memory
   3. **To destroy objects**
   4. To free memory
7. When is a destructor called in C++?
   1. When an object is created
   2. **When an object goes out of scope**
   3. When a constructor is called
   4. When a destructor is explicitly called
8. Can a class have multiple destructors in C++?
   1. Yes, as many as needed
   2. **No, only one destructor is allowed per class**
   3. It depends on the compiler
   4. Only if the class has multiple constructors
9. What happens if you do not define a destructor in a C++ class?
   1. **The compiler will generate a default destructor**
   2. The program will not compile
   3. The program will throw a runtime error
   4. Destructors are not mandatory in C++
10. Which of the following is true about the order of destructor calls in multiple inheritance?
    1. The most derived class destructor is called first, then the base class destructors
    2. **The base class destructors are called first, then the most derived class destructor**
    3. The destructors are called in the order of class declaration in the multiple inheritance hierarchy
    4. Destructors are not called in multiple inheritance
11. What are static members in C++?
    1. **Members declared with the 'static' keyword**
    2. Members declared inside the main function
    3. Members that are constant
    4. Members of a class that are accessed using objects
12. What is the purpose of static data members in C++?
    1. To allow multiple instances of data members in each object
    2. **To share the same data among all objects of the class**
    3. To restrict access to data members
    4. To create constant data members
13. How is a static member function different from a non-static member function?
    1. Static member functions can access only static data members
    2. Non-static member functions cannot be called without objects
    3. **Static member functions can be called without objects**
    4. Non-static member functions cannot have a return type
14. Can a static member function access non-static data members of a class?
    1. Yes, but only with the help of a friend function
    2. Yes, but only if the data members are marked as const
    3. **No, static member functions cannot access non-static data members**
    4. Yes, without any restrictions
15. How can you access a static data member of a class without using an object?
    1. **By using the class name and the scope resolution operator**
    2. By using the new keyword
    3. By using the this pointer
    4. By using a friend function
16. What is a copy constructor in C++?
    1. **A constructor that copies data from one object to another**
    2. A constructor that creates objects using a copy of the base class
    3. A constructor that copies all member variables to their default values
    4. A constructor with the 'copy' keyword
17. When is the copy constructor called in C++?
    1. When an object is created
    2. When an object goes out of scope
    3. **When an object is passed by value as a parameter to a function**
    4. When a destructor is called
18. What is the signature of a copy constructor?
    1. ClassName()
    2. ClassName(ClassName)
    3. **ClassName(const ClassName&)**
    4. ClassName(ClassName&)
19. What is the default behavior of the copy constructor provided by the compiler?
    1. **It performs a shallow copy of the object**
    2. It performs a deep copy of the object
    3. It generates a compilation error
    4. It generates a warning message
20. How can you prevent a class from being copied using the copy constructor?
    1. **By declaring the copy constructor as private**
    2. By declaring the copy constructor as public
    3. By not defining the copy constructor in the class
    4. By using the 'delete' keyword with the copy constructor
21. Can a destructor be overloaded in C++?
    1. Yes, with different parameter lists
    2. Yes, with different return types
    3. **No, destructors cannot be overloaded**
    4. Yes, with different access specifiers
22. What is a virtual destructor in C++?
    1. **A destructor declared with the 'virtual' keyword**
    2. A destructor that calls other destructors
    3. A destructor that prevents the destruction of objects
    4. A destructor that can be called without objects
23. When is a virtual destructor useful in C++?
    1. When a class has no destructor defined
    2. **When a class is used as a base class and objects of derived classes are deleted through base class pointers**
    3. When a class has a large number of objects
    4. When a class has static data members
24. What is the purpose of a pure virtual destructor in C++?
    1. To prevent the creation of objects of a class
    2. To allow objects to be created without using a constructor
    3. **To make the class abstract**
    4. To provide a default implementation for destructors
25. Can a pure virtual destructor have a definition in the base class?
    1. Yes, it must have a definition
    2. **Yes, but it is not mandatory**
    3. No, it cannot have a definition
    4. Only if the class has a virtual destructor as well
26. What is the scope of a static data member in C++?
    1. It is limited to the block where it is declared
    2. It is limited to the function where it is declared
    3. **It is limited to the class where it is declared**
    4. It is limited to the object where it is declared
27. How are static data members initialized in C++?
    1. They are initialized to default values
    2. They are initialized using the 'new' keyword
    3. They are initialized using the constructor
    4. **They are initialized to zero by default**
28. What is the lifetime of a static data member in C++?
    1. **It exists as long as the program runs**
    2. It exists as long as the object is in scope
    3. It exists as long as the function is being executed
    4. It exists only within the constructor
29. How can you access a private static data member of a class?
    1. By using the 'friend' keyword
    2. By using a public getter function
    3. **By using the scope resolution operator and in the class name**
    4. By using a derived class
30. What happens if a class has both a default constructor and a destructor defined?
    1. The program will not compile
    2. The default constructor will be called first, followed by the destructor
    3. The destructor will be called first, followed by the default constructor
    4. **It depends on the order of object creation and destruction**
31. Can a class have a constructor with no arguments and a constructor with default arguments?
    1. Yes, but only if they are both private
    2. Yes, as long as they have different access specifiers
    3. No, a class can have either a constructor with no arguments or a constructor with default arguments, but not both
    4. **Yes, without any restrictions**
32. Can a constructor be called explicitly in C++?
    1. **Yes, using the constructor's name and arguments**
    2. Yes, but only from another constructor
    3. No, constructors cannot be called explicitly
    4. Only if the constructor is static
33. What is the difference between a constructor and a destructor in C++?
    1. **A constructor is used to create objects, while a destructor is used to destroy objects**
    2. A destructor is called when an object goes out of scope, while a constructor is called when an object is created
    3. Constructors have a return type, while destructors do not have a return type
    4. Constructors are used for dynamic memory allocation, while destructors are used for static memory allocation