1. What is C++ programming language?
   C++ is a high-level programming language that extends the C programming language with additional features, including object-oriented programming (OOP) capabilities.
2. What is the difference between C and C++?
   C++ extends C with features like classes, objects, inheritance, and polymorphism, making it an object-oriented programming language. C++ also supports exception handling and namespaces, which are not available in C.
3. What is object-oriented programming (OOP)?
   OOP is a programming paradigm that organizes data and behavior into objects, allowing for modular and reusable code. It emphasizes concepts like encapsulation, inheritance, and polymorphism.
4. What are the four pillars of OOP?
   The four pillars of OOP are encapsulation, inheritance, polymorphism, and abstraction.
5. What is encapsulation in C++?
   Encapsulation is the process of bundling data and methods into a single unit called a class. It provides data hiding and allows access to data through controlled interfaces.

6. What is a class in C++?
   A class is a user-defined data type that serves as a blueprint for creating objects. It encapsulates data and methods that operate on that data.
7. What is an object in C++?
   An object is an instance of a class. It represents a specific entity with its own state and behavior.

8. What is inheritance in C++?
   Inheritance is a mechanism that allows a class to inherit properties and behaviors from

another class. It promotes code reuse and supports the concept of hierarchical relationships between classes.

9. What is polymorphism in C++?
   Polymorphism allows objects of different classes to be treated as objects of a common base class. It enables different objects to respond differently to the same function call.
10. What is the difference between compile-time polymorphism and runtime polymorphism?
    Compile-time polymorphism, also known as static polymorphism, is achieved through function overloading and template specialization. Runtime polymorphism, also known as dynamic polymorphism, is achieved through function overriding and virtual functions.

11. What are virtual functions in C++?
    Virtual functions are functions declared in a base class and overridden in derived classes. They allow late binding, enabling the correct function to be called at runtime based on the object's actual type.
12. What is an abstract class in C++?
    An abstract class is a class that cannot be instantiated and serves as a base for derived classes. It may contain pure virtual functions, making it an interface.
13. What is a pure virtual function?
    A pure virtual function is a virtual function that is declared in a base class but has no implementation. It must be overridden in derived classes.
14. What is a constructor in C++?
    A constructor is a special member function that is called automatically when an object is created. It initializes the object's state.

15. What is a destructor in C++?

A destructor is a special member function that is called automatically when an object is destroyed. It cleans up resources and performs necessary cleanup operations.

16. What is function overloading in C++?
    Function overloading allows multiple functions with the same name but different parameter lists to coexist in a class. The appropriate function is selected based on the arguments used during the function call.
17. What is operator overloading in C++?
    Operator overloading allows operators to be redefined for user-defined types. It enables custom-defined behavior for operators when applied to objects of a class.
18. What is the ‘this’ pointer in C++?
    The ‘this’ pointer is a special pointer available in member functions of a class. It points to the object on which the member function is being called, allowing access to its member variables and methods.
19. What is a friend function in C++?
    A friend function is a function that is not a member of a class but has access to its private and protected members. It is declared inside the class with the ‘friend’ keyword.
20. What is a template in C++?
    A template is a mechanism that allows generic programming in C++. It allows writing functions and classes that can operate on different data types without code duplication.

21. What is function template specialization?
    Function template specialization allows providing a specialized implementation for a specific data type within a template. It overrides the generic template implementation

for that specific type.

22. What is the difference between shallow copy and deep copy?
    A shallow copy copies the values of the member variables, including pointers, resulting in multiple objects pointing to the same memory locations. A deep copy creates a new object with its own copies of the member variables, including dynamically allocated memory.
23. What is a const member function?
    A const member function is a member function that does not modify the object's state. It is declared with the 'const' keyword at the end of the function declaration.

24. What are const objects in C++?
    Const objects are objects whose state cannot be modified after creation. They are declared using the 'const' keyword.
25. What is the difference between const reference and const pointer?
    A const reference must be initialized with a valid object and cannot be reassigned to refer to a different object. A const pointer can be assigned to different objects but cannot modify the object it points to.

26. What is dynamic memory allocation in C++?
    Dynamic memory allocation allows allocating memory at runtime using operators 'new' and 'delete'. It enables creating objects and data structures dynamically.
27. What is a smart pointer in C++?
    A smart pointer is a wrapper class that manages dynamically allocated memory. It automatically releases the memory when it is no longer referenced, preventing memory leaks.
28. What is the difference between unique_ptr and shared_ptr?

unique_ptr is a smart pointer that owns and manages a dynamically allocated object. It ensures exclusive ownership and cannot be copied. shared_ptr allows multiple pointers to refer to the same object and maintains a reference count to track object lifetime.

29. What is the ‘const’ keyword used for in function declarations?
    The ‘const’ keyword in function declarations indicates that the function does not modify the object’s state on which it is called. It is used for member functions that do not modify the object.
30. What is the ‘mutable’ keyword in C++?
    The ‘mutable’ keyword allows modifying a member variable of a class inside a const member function. It exempts the variable from the constness of the function.

31. What is the difference between ‘new’ and ‘malloc’ in C++?
    ‘new’ is an operator in C++ that dynamically allocates memory for an object and calls the object’s constructor. ‘malloc’ is a function in C that dynamically allocates memory but does not call the object’s constructor.
32. What is a namespace in C++?
    A namespace is a declarative region that provides a scope for the identifiers it contains. It prevents naming conflicts and organizes code into logical groups.
33. What is the ‘std’ namespace in C++?
    The ‘std’ namespace is the standard namespace in C++ that contains the standard library classes, functions, and objects.
34. What is a constructor initializer list?
    A constructor initializer list allows initializing member variables of a class in the constructor’s initialization section. It provides an efficient way to initialize member variables directly.
35. What is the difference between stack and heap memory allocation?
    Stack memory allocation is managed by the compiler and automatically allocates and

deallocates memory for local variables. Heap memory allocation is controlled manually using ‘new’ and ‘delete’ operators and allows dynamic allocation of memory.

36. What is the ‘inline’ keyword in C++?
    The ‘inline’ keyword suggests the compiler to replace function calls with the actual function code. It is used for small functions to improve performance.
37. What is the difference between a reference and a pointer?
    A reference is an alias for an object and must be initialized at declaration. It cannot be reassigned to refer to a different object. A pointer is a variable that holds the memory address of an object and can be reassigned to point to different objects.
38. What are function pointers in C++?
    Function pointers store the memory addresses of functions. They allow dynamic invocation of functions and are used in callback mechanisms and function dispatching.
39. What is a virtual destructor?
    A virtual destructor is a destructor declared in a base class as virtual. It ensures that the destructor of the most derived class is called when deleting an object through a base class pointer.
40. What is the difference between an interface and an abstract class?
    An interface in C++ is a class with pure virtual functions, making it an interface with no implementation. An abstract class can have both regular and pure virtual functions and can provide some default implementation.

41. What is the ‘delete’ operator used for in C++?
    The ‘delete’ operator is used to deallocate memory allocated with the ‘new’ operator. It releases the dynamically allocated memory and calls the object’s destructor.
42. What is the difference between a shallow copy and a deep copy in C++?
    A shallow copy copies the memory addresses of the member variables, resulting in multiple objects pointing to the same memory locations. A deep copy creates a new

object and copies the values of the member variables, including dynamically allocated memory.

43. What is a constructor chaining in C++?
    Constructor chaining refers to calling one constructor from another constructor of the same class. It allows code reuse and helps initialize objects with different constructor arguments.
44. What is the ‘typeid’ operator in C++?
    The ‘typeid’ operator returns information about the dynamic type of an object at runtime. It is often used in conjunction with polymorphism to perform type checking.
45. What is multiple inheritance in C++?
    Multiple inheritance allows a class to inherit from multiple base classes. It enables a derived class to inherit characteristics and behaviors from multiple parent classes.
46. What is a virtual base class in C++?
    A virtual base class is a base class that is declared as virtual in a derived class. It ensures that only one instance of the base class exists in the inheritance hierarchy, avoiding ambiguity in multiple inheritance scenarios.
47. What are pure virtual functions and abstract classes in C++?
    A pure virtual function is a virtual function declared in a base class but does not have any implementation. An abstract class is a class that contains at least one pure virtual function, making it an interface.
48. What is the difference between an interface and a class in C++?
    A class in C++ can have both member variables and member functions, providing implementation details. An interface is a class with pure virtual functions, defining only the interface without any implementation.
49. What is the ‘using’ keyword in C++?
    The ‘using’ keyword is used for two purposes in C++. Firstly, it can be used to create an alias for a data type or a namespace. Secondly, it can be used to bring a specific

member of a base class into the derived class scope.

50. What is the ‘explicit’ keyword in C++?
    The ‘explicit’ keyword is used to specify that a constructor or conversion function should only be called explicitly and not implicitly. It prevents implicit type conversions.

## Related Posts:

1. Machine Learning Interview Q&A
2. Python Interview Q&A
3. Top PHP Interview Questions and Answers for Success
4. C Interview Q&A
5. Java Interview Q&A
6. Jupyter Notebook Inteview Q&A
7. Computer Networks Interview Q&A
8. C# Q and A
9. Android App Deveopment Q&A
10. R Interview Q&A
11. HTML Interview Q&A
12. Basic computer interview Q&A
13. Data Structure Interview Q&A
14. Vb Net top 50 interview questions and answers