C++ Questions & Answers

1. What is the use of the `inline` keyword in C++?

- The `inline` keyword suggests to the compiler that it should attempt to embed the function's code into the caller's code to reduce the overhead of a function call. It is typically used for small, frequently called functions.

2. Explain the diamond problem in C++.

- The diamond problem occurs in multiple inheritance when a class inherits from two classes that both inherit from a common base class. This creates ambiguity about which base class member to use. It can be solved using virtual inheritance.

```
class A {
public:
    void display() { std::cout << "Class A" << std::endl; }
};

class B : virtual public A {};
class C : virtual public A {};

class D : public B, public C {};

int main() {
    D obj;
    obj.display();  // No ambiguity
    return 0;
}
```

3. What is a friend function in C++?

C++ Questions & Answers

4. What is the difference between `struct` and `class` in C++?
- The primary difference is the default access specifier: `struct` members are public by default, while `class` members are private by default.

5. What is the difference between delete and delete[] in C++?
- `delete` is used to deallocate memory allocated for a single object, while `delete[]` is used to deallocate memory allocated for an array of objects.

```
int* p = new int;
delete p;

int* arr = new int[10];
delete[] arr;
```

6. What are preprocessor directives in C++?
- Preprocessor directives are commands that are processed by the preprocessor before the actual compilation of code begins. Examples include `#include`, `#define`, `#if`, and `#endif`.

```
#include <iostream>
#define PI 3.14

int main() {
    std::cout << "Value of PI: " << PI << std::endl;
    return 0;
}
```

7. What is the purpose of the `explicit` keyword in C++?
- The `explicit` keyword is used to prevent the compiler from using a constructor for implicit conversions. It ensures that constructors can only be called explicitly.

C++ Questions & Answers

memory the pointers refer to.

9. What is the use of the `namespace` keyword in C++?

- The `namespace` keyword is used to declare a scope that holds a set of identifiers (variables, functions, etc.). This helps to organize code and prevent name conflicts.

```
namespace MyNamespace {
    void display() {
        std::cout << "Hello from MyNamespace" << std::endl;
    }
}

int main() {
    MyNamespace::display();
    return 0;
}
```

10. What is a static member in C++?

- A static member is a class member that is shared by all objects of the class. It is associated with the class itself rather than any object instance.

```
class MyClass {
public:
    static int count;
    MyClass() {
        count++;
    }
};

int MyClass::count = 0;

int main() {
```

C++ Questions & Answers

```
}
```

11. What is the difference between malloc() and new in C++?

- `malloc()` is a C library function that allocates memory but does not call constructors. `new` is an operator in C++ that allocates memory and calls the constructor for the object.

```
int* p = (int*)malloc(sizeof(int));  // malloc
*p = 10;
free(p);

int* q = new int;  // new
*q = 20;
delete q;
```

12. What is a constructor in C++?

- A constructor is a special member function that is called when an object of the class is instantiated. It initializes the object.

```
class MyClass {
public:
    int data;
    MyClass(int value) : data(value) {}  // Constructor
};

int main() {
    MyClass obj(10);
    std::cout << "Data: " << obj.data << std::endl;
    return 0;
}
```

13. Explain the differences between C and C++.

C++ Questions & Answers

14. What is a virtual function in C++?

- A virtual function is a member function declared with the `virtual` keyword. It allows derived classes to override the function and enables dynamic (run-time) polymorphism.

15. What is a template in C++?

- A template is a blueprint or formula for creating a generic class or function. It allows for type-independent code, enabling functions and classes to operate with different data types.

```
template <typename T>
T add(T a, T b) {
    return a + b;
}

int main() {
    std::cout << "Int: " << add<int>(2, 3) << std::endl;
    std::cout << "Double: " << add<double>(2.5, 3.5) << std::endl;
    return 0;
}
```

16. What are access specifiers in C++?

- Access specifiers determine the accessibility of class members. The three access specifiers are `public`, `private`, and `protected`.

```
class MyClass {
private:
    int privateData;
public:
    int publicData;
protected:
    int protectedData;
};
```

C++ Questions & Answers

- A move constructor transfers resources from one object to another, leaving the original object in a valid but unspecified state. It is used to implement move semantics and optimize performance.

18. What is a destructor in C++?

- A destructor is a special member function that is called when an object is destroyed. It performs cleanup tasks and releases resources.

19. What are the differences between pointers and references in C++?

- A pointer can be reassigned and can be null, while a reference must be initialized when declared and cannot be null or reassigned. References are typically safer and simpler to use.

20. How is exception handling implemented in C++?

- Exception handling in C++ is implemented using `try`, `catch`, and `throw` keywords. Exceptions are thrown using `throw` and caught using `catch`, with `try` blocks enclosing code that might throw an exception.

21. What is a class and an object in C++?

- A class is a blueprint for creating objects, defining data and behavior. An object is an instance of a class, representing a specific entity with state and behavior.

```
class MyClass {
public:
    void display() {
        std::cout << "Hello from MyClass" << std::endl;
    }
};

int main() {
    MyClass obj;    // Object of MyClass
    obj.display();  // Calling member function
    return 0;
}
```

C++ Questions & Answers

and easier to manage.

```
typedef int Number;
Number num = 10;
```

23. Explain the use of smart pointers in C++.
- Smart pointers are template classes in the Standard Template Library (STL) that manage dynamic memory automatically, ensuring proper deallocation. Examples include `std::unique_ptr`, `std::shared_ptr`, and `std::weak_ptr`.

24. What are virtual destructors in C++?
- A virtual destructor ensures that the correct destructor is called for derived classes when an object is deleted through a base class pointer. It is essential for proper cleanup in polymorphic base classes.

25. What is dynamic memory allocation in C++?
- Dynamic memory allocation allocates memory at runtime using `new` and `delete` operators, allowing for flexible memory management.

26. What are rvalue references in C++?
- Rvalue references are a type of reference that can bind to temporary (rvalue) objects, enabling move semantics and efficient resource transfer.

27. What is the purpose of the `std` namespace?
- The `std` namespace contains the standard C++ library functions, classes, and objects. It helps organize code and avoid name conflicts.

28. What is the use of the “mutable” keyword in C++?
- The `mutable` keyword allows a class member to be modified even if it is part of an object declared as `const`. It is useful for caching and lazy evaluation.

29. What is the difference between stack and heap memory in C++?

C++ Questions & Answers

30. What is the significance of the “this” pointer in C++?

- The `this` pointer is an implicit pointer available in member functions, pointing to the object for which the member function is called. It is used to access the object's members and differentiate between member variables and parameters.

```
class MyClass {
private:
    int num;
public:
    MyClass(int num) : num(num) {}

    void printNum() {
        std::cout << "Object's num: " << this->num << std::endl;
    }
};

int main() {
    MyClass obj(10);
    obj.printNum();
    return 0;
}
```

31. What are the uses of the `const` keyword in C++?

- The `const` keyword is used to define constant variables, constant member functions, and constant parameters. It ensures that the value cannot be modified, promoting code safety and readability.

32. Explain memory leaks in C++ and how to avoid them.

- Memory leaks occur when dynamically allocated memory is not properly deallocated, leading to wasted memory. They can be avoided using smart pointers, careful memory management, and tools like Valgrind to detect leaks.

C++ Questions & Answers

leaks, whereas Java simplifies memory management at the cost of performance.

34. What is a copy constructor in C++?

- A copy constructor is a special constructor that initializes a new object as a copy of an existing object. It performs a member-wise copy of the object's data.

```
class MyClass {
private:
    int data;
public:
    MyClass(int data) : data(data) {}

    // Copy constructor
    MyClass(const MyClass& other) : data(other.data) {}

    int getData() const {
        return data;
    }
};
```

35. What is the Standard Template Library (STL) in C++?

- The Standard Template Library (STL) is a collection of template classes and functions providing common data structures and algorithms, such as vectors, lists, stacks, queues, and sort algorithms.

```
#include <iostream>
#include <vector>

int main() {
    std::vector<int> vec = {1, 2, 3, 4, 5};
    for (int num : vec) {
        std::cout << num << " ";
    }
```

C++ Questions & Answers