C++ Programming Q and A

---

1. What is C++?

C++ is a general-purpose programming language that is an extension of the C programming language. It supports object-oriented programming, generic programming, and low-level memory manipulation.

2. Explain the difference between C and C++.

C++ is an extension of C with additional features like classes, objects, and polymorphism, making it an object-oriented programming language. C is a procedural programming language.

3. What is an object in C++?

An object in C++ is an instance of a class. It encapsulates data and behavior and represents a real-world entity.

4. What is a class in C++?

A class in C++ is a blueprint for creating objects. It defines attributes (data members) and methods (functions) that characterize the objects.

5. Explain the concept of inheritance in C++.

Inheritance is a feature that allows a class to inherit properties and behaviors from another class, creating a relationship between the base (parent) and derived (child) classes.

6. What is polymorphism in C++?

Polymorphism allows objects of different types to be treated as objects of a common type. It includes function overloading and overriding.

7. What is encapsulation in C++?

Encapsulation is the bundling of data and methods that operate on the data within a single unit, called a class. It hides the implementation details from the outside world.

8. How do you declare and initialize variables in C++?

Variables in C++ are declared with a data type and can be initialized using the assignment operator (`=`). For example, `int num = 10;`.

9. Explain the purpose of the `const` keyword in C++.
The `const` keyword is used to declare constants or to specify that a variable's value cannot be modified.

10. What is a pointer in C++?
A pointer is a variable that stores the memory address of another variable. It allows indirect access to the value stored in that memory address.

C++ Control Structures:

11. What are control structures in C++?
Control structures are statements that manage the flow of execution in a program. Examples include if statements, loops, and switch statements.

12. Explain the difference between `if` and `switch` statements in C++.
The `if` statement is used for conditional branching, while the `switch` statement is used for multi-way branching based on the value of an expression.

13.How do you use the `for` loop in C++?
The `for` loop is used for iterating a statement or a block of statements for a specific number of times. It consists of an initialization, condition, and increment/decrement expression.

14. What is the purpose of the `while` loop in C++?
The `while` loop repeats a statement or a block of statements as long as a given condition is true. It evaluates the condition before executing the loop body.

15. Explain the concept of the `do-while` loop in C++.
The `do-while` loop is similar to the `while` loop, but it evaluates the condition after executing the loop body. It ensures that the loop body is executed at least once.

16. How do you break out of a loop in C++?
The `break` statement is used to terminate the loop and transfer control to the statement following the loop.

17. What is the purpose of the `continue` statement in C++?
The `continue` statement is used to skip the rest of the loop body and move to the next iteration of the loop.

18.Explain the concept of nested loops in C++.
Nested loops are loops inside another loop. They allow you to perform more complex iterations and are often used for working with multi-dimensional arrays.

19. How do you use the `switch` statement in C++?
The `switch` statement is used for multi-way branching based on the value of an expression. It contains cases with constant values and an optional `default` case.

20. What is the ternary conditional operator (`? :`) in C++?
The ternary conditional operator is a shorthand way of writing an `if-else` statement. It evaluates a condition and returns one of two values based on whether the condition is true or false.

C++ Functions:

21. How do you declare and define a function in C++?
A function is declared by specifying its return type, name, and parameters. It is defined by providing the function body.

22. What is the difference between formal parameters and actual parameters in C++?
Formal parameters are the parameters listed in the function declaration, while actual parameters are the values passed to the function during a function call.

23. What is function overloading in C++?
Function overloading is the ability to define multiple functions with the same name but different parameter lists. The compiler selects the appropriate function based on the number and types of arguments.

24.How do you pass parameters by reference in C++?
Parameters can be passed by reference by using the `&` symbol in the function declaration. This allows the function to modify the actual arguments.

25. What is the purpose of default arguments in C++ functions?
Default arguments provide a default value for a function parameter. If the caller does not provide a value for that parameter, the default value is used.

26.Explain the concept of recursion in C++.
Recursion is a programming technique where a function calls itself to solve a smaller instance of the problem. It must have a base case to avoid infinite recursion.

27. What is a lambda expression in C++?
A lambda expression is an anonymous function that can capture variables from its enclosing scope. It provides a concise way to define functions inline.

28. How do you use function pointers in C++?
Function pointers store the address of functions. They are declared by specifying the return type and parameter types of the function they can point to.

29. What is the purpose of the `inline` keyword in C++?
The `inline` keyword suggests to the compiler that it should attempt to generate inline code for a function, reducing the overhead of function calls.

30. How do you use namespaces in C++?
Namespaces are used to organize code into separate logical units. They are declared using the `namespace` keyword, and their members are accessed using the scope resolution operator (`::`)

31. What are the basic data types in C++?
Basic data types in C++ include `int`, `float`, `double`, `char`, `bool`, and `void`.

32. Explain the concept of pointers in C++.
Pointers are variables that store memory addresses. They are used for dynamic memory allocation, passing parameters by reference, and working with arrays.

33. How do you use the `sizeof` operator in C++?
The `size of` operator returns the size, in bytes, of a data type or an object. It is commonly used for memory allocation and manipulation.

34. What is a reference variable in C++?
A reference variable is an alias for an existing variable. It provides an alternative name for the same memory location and is often used as function parameters.

35. Explain the concept of type casting in C++.
Type casting is the conversion of a variable from one data type to another. It can be implicit or explicit using casting operators like `static_cast` and `dynamic_cast`.

36. How do you use the `auto` keyword in C++?
The `auto` keyword is used for automatic type inference, allowing the compiler to deduce the data type of a variable based on its initializer.

37. What is the purpose of the `const` qualifier in C++?
The `const` qualifier is used to specify that a variable's value cannot be changed. It is also used to indicate that a function does not modify the object it is called on.

38. Explain the bitwise operators in C++ (`&`, `|`, `^`, `<<`, `>>`).
Bitwise operators perform operations at the bit level. `&` is AND, `|` is OR, `^` is XOR, `<<` is left shift, and `>>` is right shift.

39. What is the purpose of the `typeid` operator in C++?
The `typeid` operator returns type information about an expression, including information about its dynamic type.

40.How do you use the conditional operator (`?:`) in C++?
The conditional operator is a ternary operator that returns one of two values based on a specified condition. It is a shorthand for an `if-else` statement.

41. What is a C++ string, and how is it different from a character array?
C++ string is a sequence of characters stored as an object, while a character array is a collection of characters stored in a contiguous memory location.

42. How do you declare and initialize a string in C++?
Strings can be declared using the `string` class and initialized with a sequence of characters or another string.

43. Explain the concept of string concatenation in C++.
String concatenation is the process of combining two or more strings into a single string. It can be done using the `+` operator or the `append` method.

44. How do you find the length of a string in C++?
The `length()` or `size()` member functions of the `string` class can be used to find the length of a string.

45. What is the purpose of the `c_str()` method in C++ strings?
The `c_str()` method returns a pointer to a null-terminated character array representing the contents of the string. It is often used when interfacing with C-style functions.

46. Explain the concept of arrays in C++.
Arrays in C++ are collections of elements of the same data type stored in contiguous memory locations. They can be one-dimensional or multi-dimensional.

47. How do you declare and initialize an array in C++?
Arrays can be declared by specifying the data type and the array name, and they can be initialized using curly braces `{}` with the values.

48. What is the difference between an array and a vector in C++?
Vectors are a part of the Standard Template Library (STL) and provide dynamic sizing, whereas arrays have a fixed size. Vectors also provide additional functions and memory management.

49.How do you access elements in an array in C++?
Array elements are accessed using indices within square brackets. The index starts at 0 for the first element.

50. What is a two-dimensional array, and how is it accessed in C++?
A two-dimensional array is an array of arrays. It is accessed using two indices—one for the row and one for the column.

51. What is dynamic memory allocation in C++
Dynamic memory allocation refers to the process of allocating memory at runtime using operators like `new` and `delete`.

52. How do you allocate memory for a single variable dynamically in C++?
Use the `new` operator to allocate memory for a single variable dynamically. For example, `int *ptr = new int;`.

53.What is the purpose of the `delete` operator in C++?
The `delete` operator is used to deallocate memory that was previously allocated using the `new` operator, preventing memory leaks.

54. How do you allocate memory for an array dynamically in C++?
Use the `new` operator with square brackets to allocate memory for an array dynamically. For example, `int *arr = new int[10];`.

55. What is the difference between `delete` and `delete[]` in C++?
Use `delete` for deallocating memory allocated with `new`, and use `delete[]` for deallocating memory allocated with `new[]`.

56. Explain the concept of memory leaks in C++?
Memory leaks occur when a program allocates memory but fails to deallocate it, resulting in the loss of memory that cannot be used by the program.

57. What is a smart pointer in C++?

Smart pointers are objects that act like pointers but provide automatic memory management. They automatically release allocated memory when the pointer is no longer needed.

58. How do you use `malloc` and `free` for memory allocation in C++?

`malloc` is used to allocate memory, and `free` is used to deallocate memory. However, in C++, it is recommended to use `new` and `delete` for dynamic memory management.

59. What is the purpose of the `nullptr` keyword in C++?

`nullptr` is a keyword introduced in C++11 and serves as a better alternative to `NULL` for representing a null pointer.

60. Why C++?

The use of C++ is varied such as:

– It is used in developing graphical user interface-based applications like adobe photoshop.

– It is used in developing games as it overrides the complexity of 3D games.

– There is much-animated software developed in C++

– Most of the compilers are written in C++.

– Google Chrome, Mozilla Firefox etc. web browsers are developed using C++

There are many more such uses that make C++ the desired language.

61.why c++? What is namespace in C++?

If there are two or more functions with the same name defined in different libraries then how will the compiler know which one to refer to? Thus namespace came to the picture. A namespace defines the scope and differentiates functions, classes, variables etc. with the same name available in different libraries. The namespace starts with the keyword “namespace”. The syntax for the same is as follows:

62. What is operator overloading in C++?

Operator overloading in C++ is an overloaded declaration is declaration in the same scope of function or operator declared with the same name more than once.

63.How to learn C++?

C++ is a programming language which is an extension of C. Thus, one should prefer to learn C first (it’s not necessary). After learning C, then understand the basic difference between C and C++. Implement all the basic programs you learnt in C in C++ also. Then dive into the OOPs concept of C++. Do as much hands-on as possible to understand basic OOPs, and then dive into advanced-level OOPs. When all the basics are clear, build

a small game to understand the structure and remain concepts if any. By following all these steps one can learn C++.

64.What is the difference between C and C++?

The difference between c and c++ is that C++ is an object-oriented language, which means that it has all the features of C as well as its own thing which is the concept of OOP. C++ has many functionalities of OOP that are missing from C such as encapsulation, abstraction, classes, objects, etc.

Feature	C	C++
Programming Paradigm	Procedure-oriented	Functions cannot be defined inside structures
Data Hiding Support	Does not support data hiding	Supports data hiding
Relationship between C and C++	C is a subset of C++	C++ is a superset of C
Function and Operator Overloading Support	Does not support function and operator overloading	Supports function and operator overloading
Functions Inside Structures	Functions cannot be defined inside structures	Functions can be defined inside structures

65. What is a template in C++?

A template in C++ is used to pass data types as parameters. These make it easier and simpler to use classes and functions.

```
template <typename T>
  int fun (T a,T b)
        {
            return (a+b);
        }
        int main(){
            cout<<fun<int>(11,22);
        }
```

66.What is using namespace std in C++?

Using namespace std in C++ tells the compiler that you will be making use of the namespace called ‘std’. The ‘std’ namespace contains all the features of the standard

library. You need to put this statement at the start of all your C++ codes if you don't want to keep on writing std:: infront of every variable/string or whatever standard library feature you are making use of, as it becomes tedious to do so.

67. How to download turbo C++ for windows 10?

To download turbo c++ follow the steps mentioned below:

Step-1: Download turbo C++ from http://www.turboccom/p/download.html

Step-2: Extract the Turbo.C.zip file.

Step-3: Run setup.exe file.

Step-4: Follow the instructions mentioned.

68.How to download turbo C++ for windows 10?

How to paste in turbo C++?

Paste in turbo C++ can be done by two techniques:

Shift+Insert

Open the file in notepad with .cpp extension. Make the changes and save it. After saving the file, you can open it from the Turbo C++ application file menu from where you stored the cpp file.

What is a pointer in C++?

Pointers in C++ are a data type that store the memory address of another variable.

For eg.

```
char *str = "Hi, How are you?";
        Here the pointer variable *str points to the string "Hi, How are you?"
        or
        int age;
        int *int_value;
        *int_value = &age;
        cout<<"Enter your age please:";
        cin>>age;
        cout<<"\n Your age is:"<<*int_value;
        // this will print your age as the variable is pointing to the variable age.
```

69.What is a function in C++?

A function in C++ is a block of code that can be referenced from anywhere in the system and that serves a specific purpose.

```
int fun(){
        int a = 11;
        return 11;
    }
    int main(){
        int b = fun();
```

```
}
```

70.What is a destructor in C++?

Destructors in c++ are special functions/methods that are used to remove memory allocation for objects. They are called usually when the scope of an object ends. eg. when a function ends you can call it a destructor.

1

They are of the same name as the class - syntax - ~<classname>();

71.What is function overloading in C++?

Function Overloading happens in C++ when two or more functions share the same name. They can be differentiated on the basis of the type of data they are passing as parameters or even the number of parameters they are passing. eg. int fun(char a); & int fun(int b); & void fun(int a, int b)

72.What is stl in C++?

Stl is the standard template library. It is a library that allows you to use a standard set of templates for things such as: Algorithms, functions, Iterators in place of actual code.

```
queue<int> Q;
    for(k=0;k<10;k++)
    {
        Q.push(k);
    }
```


73.How to run a C++ program in cmd?

verify gcc installtion using the command:

$ gcc -vthen go to your working directory or folder where your code is:

$ cd <folder_name>then build the file containing your c code as such:

$ gcc main.cpp

or

$ g++ -o main main.cpp then run the executable generated in your system:

$ main.exe

74.What is type casting in C++?

Type casting in C is used to change the data type. They are of two types: Implicit Type Conversion: It is automatic. Explicit Type Conversion: It is user-defined.

75.How to use a string in C++?

A string is a sequence of characters. In C++, the string is a data type as well as a header file. This header file consists of powerful functions of string manipulation. A variable of string is declared as follows:

string str= "Hello";
And to use string one needs to include the header file.
// Include the string library
#include <string>
// Create a string variable
string str= "Hello";
What is stream in C++?
Stream refers to a stream of characters to be transferred between program thread and i/o.

76.What is the difference between structure and class in C++?
The difference between structure and class is as follows:
– By default, the data members of the class are private whereas data members of structure are public.
– While implementing inheritance, the access specifier for struct is public whereas for class its private.
– Structures do not have data hiding features whereas class does.
– Structures contain only data members whereas class contains data members as well as member functions.
– In structure, data members are not initialized with a value whereas in class, data members can be initialised.
– Structures are stored as stack in memory whereas class is stored as heap in memory.

77.How to clear screen in C++?
One can clear screen using – clrscr() or system(“clear”).
How to compile and run C program in notepad++ ?
To compile and run c program in notepad++ follow the steps mentioned below:
Step-1: Download and install notepad++
Step-2: Download and install MinGw gcc along with gcc.
Step-3: Configure notepad++ for gcc. This step can be further divided into two sub-steps. A: Create C compiler tool in Notepad++
B: Creating C execution tool.
Step-4: Execute C program in Notepad++

78.How many keywords in C++?
There are 95 reserved keywords in C++ which are not available for re-definition or overloading.

79.What is iostream in C++?
It is a header file that includes basic objects such as cin, cout, cerr, clog.

80.How to give space in C++?

In C++ programming, the space can be given using the following code.

cout << " " ;

Which operator cannot be overloaded in C++ ?

Some of the operators that cannot be overloaded are as follows:

– Dot operator- "."

– Scope resolution operator- "::"

– "sizeof" operator

– Pointer to member operator- ".*"

81.How to copy and paste in turbo C++ ?

Press Ctrl + Insert to copy.

Press Shift + Insert to paste.

82.What is an exception in C++?

Runtime abnormal conditions that occur in the program are called exceptions. These are of 2 types:

– Synchronous

– Asynchronous

C++ has 3 specific keywords for handling these exceptions:

– try

– catch

– throw

83.What is the difference between C++ and Java?

This is one of the most common c++ interview questions asked, the difference between c++ and java are as follows:

– C++ supports goto statements whereas Java does not.

– C++ is majorly used in system programming whereas Java is majorly used in application programming.

– C++ supports multiple inheritance whereas Java does not support multiple inheritance

– C++ supports operator overloading whereas Java does not support operator overloading.

– C++ has pointers which can be used in the program whereas Java has pointers but internally.

– C++ uses a compiler only whereas Java uses both compiler and interpreter.

– C++ has both call by value and call by reference whereas Java supports only call by value.

– C++ supports structures and joins whereas Java does not support structure and joins

– Java supports unsigned right shift operator (>>>) whereas C++ does not.

– C++ is interactive with hardware whereas Java is not that interactive with hardware.

84.What is stack in C++?

A linear data structure which implements all the operations (push, pop) in LIFO (Last In First Out) order. Stack can be implemented using either arrays or linked list.The operations in Stack are

– Push: adding element to stack

– Pop: removing element from stack

– isEmpty: returns true if stack is empty

– Top: returns the top most element in stack

85.What is conio.h in C++?

Conio.h is a header file used for console input and output operations and is used for creating text based user interfaces.

86.How to exit from turbo C++?

To exit Turbo C++, use the Quit option under the File Menu, or press Alt + X.

87.What is iterator in C++?

Any object which has an ability to iterate through elements of the range it has been pointing to is called iterator.

88.What is :: in C++?

:: is called a scope resolution operator which is used to access global variables with the same name as of local variables, for defining functions outside the class, for accessing static variables, and for referring to a class inside of another class.

89.What is enum in C++?

enum is abbreviation of Enumeration which assigns names to integer constant to make a program easy to read. Syntax for the same:

1

```
enum enum_name{const1, const2, ..... };
```

90.How to save a file in C++?

When you have written code in the file (notepad),save the file as “hello.cpp.” If you want to write in a file using C++ code, you can do it using iostream and fstream libraries in C++.

```
#include <iostream>
#include <fstream>
using namespace std;
int main () {
  ofstream file_name;
  file_name.open ("sample.txt");
```

```
file_name<< "Write in the file";
file_name.close();
return 0;}
```

91.Which operators can be overloaded in C++?

List of operators that can be overloaded are:

+ , - , * , / , % , ^, & , | , ~ , !, =, ++ , --, ==, != , && , ||

+= , -= , /= , %= , ^= , &=, |= , *= , = , [] , (), ->, ->* , new , new [] , delete , delete []

The library <bits/stdc++.h> in c++ is used to include all the libraries.

92.How to maximize turbo C++ window?

Alt+Enter is the keyboard shortcut used to maximize (full screen) turbo C++.

93.What is an expression in C++?

An expression is a combination of operators, constants and variables. These seven types of expressions for examples:

- Constant expressions: 89 +10/0
- Integral expressions: x * y
- Floating expressions: 189
- Relational expressions: a<=b
- Logical expressions: a > b && a == 7
- Pointer expressions: *ptr
- Bitwise expressions: p << 5

94. Why namespace std is used in C++?

If the program does not have using namespace std; then when you write cout <<; you would have to put std::cout <<; same for other functions such as cin, endl etc.

95.Which is the best C++ compiler?

There are several good compilers for C++ such as:

- ➔ MinGW /
- ➔ GCC- Borland c++
- ➔ Dev C++
- ➔ Embracadero
- ➔ Clang
- ➔ Visual C++
- ➔ Intel C++
- ➔ Code Block
- ➔ GCC and clang are great compilers if the programmers target more portability with good speed.

➔ Intel and other compilers target speed with relatively less emphasis on portability.

96.What are the different data types present in C++?

The 4 data types in C++ are:

➔ Primitive Datatype such as char, short, int, float, long, double, bool, etc.
➔ Derived datatype such as an array, pointer, etc.
➔ Enumeration such as enum
➔ User-defined data types such as structure, class, etc.

What are the advantages of C++?

➔ Mid-level programming language
➔ Portability
➔ C++ has the concept of inheritance
➔ Multi-paradigm programming language
➔ Memory Management
➔ C++ is a highly portable language
➔ Fast and Powerful
➔ C++ contains a rich function library

97.What is the difference between reference and pointer?

	Reference	**Pointers**
Usage	Refers to an existing variable by another name.	Stores the address of a variable.
Null Value	Cannot have a null value assigned.	Can have a null value assigned.
Referencing	Referenced by the value.	Referenced but can be passed by reference.
Initialization	Must be initialized on declaration.	No need to be initialized on declaration.
Memory	Shares the same memory address with the original variable.	Has its own memory address and size on the stack.

98.What is visual C++?

C++ is a standardized language and Visual C++ is a product that implements the standard of C++. One can write portable C++ programs using Visual C++, but one can

also use Microsoft-only extensions which destroy portability but enhances your productivity.

99.What is stl in C++ with example?

STL in C++ is a library and abbreviation of Standard Template Library. STL is a generalized library that provides common programming data structures/ container classes, functions, algorithms, and iterators. STL has four components

- Algorithms: Searching and sorting algorithms such as binary search, merge sort etc.
- Containers: Vector, list, queue, arrays, map etc.
- Functions: They are objects that act like functions.
- Iterators: It is an object that allows transversing through elements of a container, e.g., vector<int>::iterator.

100. What is flush in C++?

std::flush synchronizes the stream buffer with its controlled output sequence.

Advanced C++ Interview Questions

This section of the blog talks about advanced C++ Interview Questions for your reference.

101.What is a class in C++?

C language is not an object-oriented programming language, so it is a constant attempt of C++ to introduce OOPs. Class is a user-defined data type that defines a blueprint of data type. For example,

```
class Circle{
public:
 float radius; }
```

102.What is inline function in C++?

Inline functions are functions used to increase the execution time of a program. Basically, if a function is inline, the compiler puts the function code wherever the function is used during compile time.

103.What is friend function in C++?

A friend function has access rights to all private and protected members of the class.

```
class Circle
{
double radius;
public:
friend void printradius( Circle c );
};
void printradius(Circle c )
```

```
{
/* Because printradius() is a friend of Circle, it can  directly access any member of this class */
cout << "Radius of circle: "
<< c.width;}int main()
{
Circle c;
// Use friend function to print the radius.
printradius( c);  return 0;
}
```

104. How to use vector in C++?

A sample code to see how to use vector in C++ is as follows:

```
#include<iostream>
#include<vector>
using namespace std;
int main()
{
vector <string> vec_1;
vec_push_back("sample code");
vec_push_back("change example");
for(vector <string>::iterator i=vec_begin();i!=vec_end();++i)
cout<<*i;
return 0;
}
```


105.What is vector in C++?

A sequence of containers to store elements, a vector is a template class of C++. Vectors are used when managing ever-changing data elements. The syntax of creating vector.

1
2
3
4
5

vector <type> variable (number of elements)

For example:

```
 vector <int> rooms (9);
```

106.What is scope resolution operator in C++?

Scope resolution operator in c++ is denoted by double colon (::). It can be used:

– when there is a local variable with same name as of global variable

– When a function has to be defined outside a class

– When class's static variables needs to be accessed

– When a class inside another class has to be referred

– In case of multiple Inheritance

107.What are character constants in C++?

A character constant is member of the character set in which a program is written which is surrounded by single quotation marks (').

108.What are templates in C++?

A feature that allows functions and classes to operate with generic types which means a function or class can work on different data types without being rewritten is called a template.

109.How to sort vector in C++?

```
#include <bits/stdc++.h>
using namespace std;
int main()
{
   vector<int> vec{ 1,9,4,3,2,8,5,7};
   sort(vec.begin(), vec.end());
   for (auto x : v)
      cout << x << "" "";
   return 0;
}
```


110.What is pure virtual function in C++?

A pure virtual function is a type of virtual function which does not have implementation, but is only declared. It is declared by assigning 0 in declaration.

111.How to empty a vector in C++?

Std::vector::empty tests whether a vector is empty or not. A sample code for illustrating the same is as follows:

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

int main ()
```

```
{
 std::vector<int> vec;
 int add (0);
 for (int i=1;i<=5;i++) vec.push_back(i);
 while (!vec.empty())
 {
   add+= vec.back();
   vec.pop_back();
 }
 std::cout << add;
 return 0;
}
```

112.How to remove segmentation fault in C++?

Segmentation fault indicates an error memory corruption. In layman terms, when a piece of code tries to do read and write operation in a read only location in memory. Below are the reasons and solutions for segmentation error:

Reason: Accessing an address that is freed

```
int* p = malloc(8);
*p = 100;
 free(p);
 *p = 110;
```

Solution: Before freeing the pointer check the assignment or any operation required to perform.

Reason: Accessing out of array index bounds

```
int arr[2];
arr[3] = 10;
```

Solution: Correcting the array bound

Reason: Improper use of scanf()

```
int n = 2;
scanf("%d",n);
```

Solution: To avoid this is the only solution

Reason: Dereferencing uninitialized pointer

```
int *p;
printf("%d",*p);
```

Solution: A pointer must point to valid memory before accessing it.

Reason: Stack Overflow

Solution: It can be resolved by having a base condition to return from the recursive function.

How to initialize a 2d vector in C++?

The syntax to initialize a 2d vector is as follows:

std::vector<std::vector<int> > name_of_vector;

For example: std::vector<std::vector<int> > v { { 1, 2, 1 },

{ 2, 6, 7 } };

C++ OOPS Interview Questions

C++ Interview Questions also include questions on OOPs Concepts. This section on C++ OOPS Interview Questions will help you learn more about the concepts.

113.What is oops in C++?

OOP or Object Oriented Programming in C++ is a type of programming in which you create objects and classes to emulate real-world concepts such as Abstraction, Polymorphism, Encapsulation, and Inheritance.

Here classes are data types that allow you to list several types of data within it and even functions. You can access these classes with the help of class objects

114.What is a constructor in C++?

Constructor in C++ is a method in the class which has the same name as that of the class and is followed by parentheses (). It is automatically called when an object of a class is created.

```
class Hello {     // The class

  public:          // Access specifier

    Hello() {     // Constructor

      cout << ""Hello World!"";
    }
};
int main() {
  Hello obj;    // Create an object of Hello (this will call the constructor)
  return 0;
}
```

115.What is inheritance in C++?

Inheritance in C++ is just like a child inherits some features and attributes from his parent similarly a class inherit attributes and methods from another class. The parent class is called base class and the child class is called derived class.

```
// Base class
class Food_Item{
  public:
    void taste() {
    cout << ""The taste of every food item is different. \n"" ;
  }
};
// Derived class
class Chips: public Food_Item{
  public:
    void taste() {
    cout << ""The taste of chips is salty \n"" ;   }
};
```

116.What is object in C++?

Class in C++ provides a blueprint for object, that means, object is created from the class.

117.What is encapsulation in C++?

To prevent access to data directly, Encapsulation is the process that combines data variables and functions in a class. This is achieved by doing the following:

- ➢ Making all data variables private.
- ➢ Creating getter and setter functions for data variables.

118.What is an abstraction in C++?

Abstraction in C++ means showing only what is necessary. It's part of the Object-oriented Programming concept. Abstraction is used to hide any irrelevant data from the outside world and only show what is absolutely necessary for the outside world to use.

eg. Classes use the abstraction concept to only show relevant data types or elements. This is done through access specifiers such as: public, private, and protected.

119.What is a member function in C++?

Member functions are those functions that you declare within a class, they are members of the class. You can reference them using class objects. Eg:

```
class A
{
public:
```

```
    int add(int b)
    {
    a = b * 10;
    return a;
    };
};
```

120.What is a virtual base class in C++?

Let's understand this with an example.

You Have 4 classes: W,X,Y,Z

Here X & Y inherit from W. So they both have similar features being inherited from W.

Now, Z inherits from both X & Y

Here Z may inherit similar features from X & Y as they both have inherited them from W. This can cause issues and that's why we use virtual base classes as they stop multiple features of a class from appearing in another class.

121.1How to access private members of a class in C++?

Private members of the class are not accessible by object or function outside the class. Only functions inside the class can access them or friend functions. However, pointers can be used to access private data members outside the class.

The sample code is as follows:


```
#include <iostream>
using namespace std;
class sample_test{
private:
   int n;
public:
   sample_test() { n = 45; }
   int display() {
return n;
    }
};
```

122.How to call a base class constructor from a derived class in C++?

A base class constructor will be called whenever the derived class constructor is called. Upon the creation of a derived class object, the order of constructor execution is: base class constructor then Default class constructor.

123.What is an abstract class in C++?

An abstract class in C++ is such that cannot be used directly and is used to form a base class for others to inherit from.

If you create an object for an abstract class the compiler will throw an error at you.

124.What is containership in C++?

Containership in C++ is a relationship in which a class's object is nested within another class. The class that contains the object is called a container class and the class whose object is stored is called a contained class.

125.What is data hiding in C++?

An object-oriented technique of hiding data members is called data hiding. In other words, giving restricted access to the data members so as to maintain object integrity.
Polymorphism Concept
polymorphism in C++

126.What is runtime polymorphism in C++?

Polymorphism means having many forms whether it is a function or operator in programming.
Runtime polymorphism is achieved by function overriding.
#include <bits/stdc++.h>

127.What is copy constructor in C++?

A copy constructor in c++ is a constructor which creates an object by initializing it with an object of the same class, which has been created previously.
The syntax for copy constructor is as follows:

128.1How is modularity introduced in C++?

Modularity is a way of mapping encapsulated abstractions into real and physical modules which is closely related to Encapsulation. It is a concept in which separate programs are divided into separate modules.
For example, when building a house it is built in modular way. First foundation is laid, then structure is made and so on.

129.What is the size of an empty class in C++?

The size of an empty class is 1 byte generally just to ensure that the two different objects will have different addresses.

130.C++ Programming Interview Questions

Programming is an important aspect for any programmer or developer. This section of the blog talks about c++ interview questions that will be beneficial to programming. Here is the list of the top 20 c++ programming questions.

131.How to write hello world in C++?

Hello world in C++ is as follows:

```
#include <iostream>
int main()
{
  std::cout << "Hello, World!";
  return 0;
}
```

132.How to input string in C++?

There are three ways to input a string, using cin, get, and getline. All three methods are mentioned in the sample program below.

```
#include <iostream>
using namespace std;

int main()
{
   char s[10];

   cout << "Enter a string: ";
   cin >> str;

   cout << "\nEnter another string: ";
   cin.get(s, 10);

   getline(cin, str);

   return 0;
}
1
```

133.How to reverse a string in C++?

To reverse a string, a sample code is mentioned below.

```
#include<iostream>
#include<string.h>
using namespace std;
int main ()
{
   char n[50], t;
```

```
   int i, j;
   cout << "Enter a string : ";
   gets(n);
   i = strlen(n) - 1;
   for (j = 0; j < i; j++,i--)
   {
      t = s[j];
      s[j] = s[i];
      s[i] = t;
   }
   cout << "\nReverse string : " << s;
   return 0;
}
```

134.How to convert integer to string in C++?

There are 2 approaches to convert integer variables to string. Both the approaches with a sample code are mentioned below.

Approach-1

```
#include<iostream>
#include<string>
using namespace std;
void main()
{
   int n= 1;
   string s= to_string(n);
   cout << s;
}
```

Approach-2

```
#include<iostream>
#include <sstream>
#include <string>
using namespace std;
int main()
{
   int n = 17;

   // declaring output string stream
   ostringstream s1;

   // Sending a number as a stream into output str
   s<< n;
```

```
    // the str() converts number into string
    string fin = s.str();
    // Displaying the string
    cout << fin;
    return 0;
}
```

135.How to input string in C++ with spaces?

The code to input a string in C++ with spaces is as follows:

```
#include <iostream>
#include <string>
using namespace std;
int main()
{
    string s;
    cout << "Enter the sentence";
    getline(cin, s);
    cout << str;
    return 0; }
```

136.How to dynamically allocate a 2d array in C++?

There are several methods by which one can allocate memory to 2D array dynamically one of which is as follows:

```
#include <iostream>
int main()
{
    int row = 2, col = 2;
    int* a =  new int[row * col];

    int i, j, count = 0;
    for (i = 0; i <  row; i++)
      for (j = 0; j < col; j++)
         *(a+ i*col + j) = count++;

    for (i = 0; i <  row; i++)
      for (j = 0; j < col; j++)
         printf("%d ", *(a + i*col + j));

    delete[ ] a;
    return 0;
```

}

137.How to use goto statement in C++ ?

Goto statement provided unconditional jump in the code.

The syntax is

1

2

goto label;

label: statement;

```
#include <iostream>
using namespace std;

void main () {
   float d, avg, add = 0.0;
   int j, n;
   cin >> n;

   for(j = 1; j <= n; ++j)
   {
      cout << "Enter number" << i;
      cin >> d;

      if(d < 0.0)
      {
            goto jump;
      }
      add+= d;
   }

jump:
   avg = add/ (j- 1);
   cout << avg;
 }
```

138.What is function overriding in C++?

When a function with same name is present in both parent and child class then it is called function overriding.

```
#include <iostream>
using namespace std;
class parent {
```

```
public:
  void display(){
    cout<<"Parent Class";
  }
};
class child: public parent{
public:
  void display() {
    cout<<"Child Class";
  }
};
int main() {
  child o = parent();
  o.display();
  return 0;
}
```

139.What is bool in C++?

Bool is a data type in C++ which takes two values- True and False.

The syntax is as follows:

1

```
bool b1 = true;
#include<iostream>
using namespace std;
int main()
{
   int a= 60, b= 70;
   bool c, d;
   c= a== b; // false

   c= a< b; // true

   cout <<b1;
   cout << b2 ;

   return 0;
}
```

140.How to set decimal places in C++ ?

For limiting the decimal places in C++ there are five functions : floor(), ceil(), trunc(), round() and setprecision(). Out of these five, only setprecision() function is used for

setting the decimal places to put as output. All the functions are mentioned in the following sample code.

```
#include<bits/stdc++.h>
using namespace std;
int main()
{
   float a =33333;
   cout << floor(a) << endl;
   cout << ceil(a) << endl;
   cout << trunc(a) << endl;
   cout << round(a) << endl;
   cout << setprecision(2) << a;
   return 0;
}
```

141.1How to get absolute value in C++?

In C++, there are three functions in the cstdlib header file to return the absolute value of the integer. Those are:

abs()
labs()
llabs()

The syntax for all the functions is same

1

function_name(integer value)

The difference lies in the range for integer value being passed as an argument.

For abs() its type int in C++.

For labs(), its type long int in C++

For llabs() its long long int in C++.

The sample code for illustrating the three functions is as follows:

```
#include <cstdlib>
#include <iostream>

using namespace std;

int main()
{
   int a, b, c;

   a = abs(22);
```

```
    b= labs(1234355L);
    c= llabs(1234863551LL);
    cout << a;
    cout << b;
    cout<< c;
    return 0;
}
```

142.How to concatenate string in C++ ?

The strings in C++ can be concatenated in two ways- one considering them string objects and the second concatenating them C-style strings.

```
#include <iostream>
using namespace std;

int main()
{
    string s_1, s_2, fin;
    cout << "Enter string";
    getline (cin, s_1);
    cout << "Enter string ";
    getline (cin, s_2);
    fin= s_1 + s_2;
    cout << fin;

    char str1[50], str2[50], fin[100];

    cout << "Enter string";
    cin.getline(str1, 50);
    cout << "Enter string";
    cin.getline(str2, 50);
    strcat(str1, str2);
    cout << "str1 = " << str1 << endl;
    cout << "str2 = " << str2;
    return 0;
}
```


143.How to convert char to int in C++ ?

There are three methods for converting char variable to int type variable. These are as follows:

atoi()
sscanf()
typecasting
A sample code depicting all three functions are as follows:

```
#include<stdio.h>
#include<stdlib.h>
int main() {
  char *s = "6790";
  char d = 's';
  int a,b,c;
  sscanf(s, "%d", &a); // Using sscanf
  printf("a : %d", a);
  b = atoi(s); // Using atoi()
  printf(“b : %d", b);
  c = (int)(d); // Using typecasting
  printf("c : %d", c);
  return 0;
}
```

144.1How to generate random numbers in C++ with a range?

Using the rand() function we can generate random numbers in C++ within a range.

```
#include <iostream>
#include <random>
int main()
{
  int max=100, min=54,i;
  int range = max - min + 1;
  for (i=min; i<max;i++)
   {
      int num = rand() % range + min;
      cout<<num;
   }
   return 0;
}
```

145.How to find absolute value in C++?

To find the absolute value in c++, we can use abs() function. The abs() function in C++ returns the absolute value of an integer number.

```
#include <iostream>
#include <cstdlib>
using namespace std;
```

```
int main()
{
      int a=3.456;
      int x = abs(a);
      cout << x;
         return 0;
}
```

146.How to write a class in C++?

A class in C++ is the building block that leads to Object-Oriented programming and is a user-defined data type which holds data and functions. The syntax to write a class in C++ is as follows:

```
Class (keyword) Class_Name (this is user defined)
{
   Access specifier: // private, public, protected
   Data members //int, char, float, double etc. variables to be used
   Member function() { }  // Methods to access data members
};    //Class end
```

For example:

```
class Sample
{
   // Access specifier
   private:

   // Data Members
   string s;

   // Member Functions()
   void printname()
   {
     cout << s;
   }
};
```


146.How to use strcmp function in C++?

strcmp() function is an in-built function of <string.h> header file which takes two strings as arguments and compares these two strings lexicographically.

The syntax of the function is as follows:

1

```
int strcmp(const char *l, const char *r );
```

```
#include<stdio.h>
#include<string.h>
int main()
{
   // z has greater ASCII value than g
   char a[] = "zfz";
   char b[] = "gfg";
   int r = strcmp(a, b);
   if (r==0)
      printf("Strings are equal");
   else
      printf("Strings are unequal");
   printf("%d" , r);
   return 0;
}
```

147.How to write to a file in C++?

A file is read in c++ using a fstream header file.

```
#include <iostream>
#include <fstream>
using namespace std;
int main()
{
   ofstream fout;
   string r;
   fout.open("test.txt");
   while (fout) {
      getline(cin, r);
      if (r == "-1")
         break;
      fout << line << endl;
   }
   fout.close();

   ifstream fin;
   fin.open("test.txt");
   while (fin) {
       getline(fin, line);
        cout << line << endl;
   }
   fin.close();
```

```
    return 0;
}
```

148.What is stringstream in C++?

Stringstream is a class in c++ that associates a string object with a stream allowing to read from the string as if it were a stream.

Syntax is as follows:

stringstream string_name(str);

Basic operations are as follows:

clear()

str()

<<

149. What is a pointer in C++?

A pointer is a variable that stores the memory address of another variable. It allows direct manipulation of memory.

150. Explain the difference between malloc() and new for memory allocation.

malloc() is a function in C used for dynamic memory allocation, while new is an operator in C++ that allocates memory and calls constructors.

151. What are the different types of inheritance supported by C++?

C++ supports various types of inheritance: single, multiple, multilevel, hierarchical, and hybrid inheritance.

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

A virtual function is a member function that is declared within a base class and redefined by a derived class. It allows dynamic binding and polymorphism.

153. Explain the concept of function overriding in C++.

Function overriding occurs when a derived class provides a specific implementation for a function that is already defined in its base class.

154. What is the difference between reference and pointer in C++?

A reference is an alias for a variable and must be initialized when declared. A pointer stores the memory address of a variable and can be reassigned.

155. What is the purpose of the const keyword in C++?

The const keyword is used to declare constants or to specify that an object or variable is not modifiable.

156. Explain the use of the sizeof operator in C++.
The sizeof operator returns the size, in bytes, of its operand, which can be a data type, variable, or expression.

157. What is a constructor in C++?
A constructor is a special member function that is automatically called when an object is created. It initializes the object's data members.

158. What is the difference between delete and delete[] in C++?
delete is used to deallocate memory for a single object allocated with new, while delete[] is used to deallocate memory for arrays allocated with new[].

159. What is the purpose of the volatile keyword in C++?
The volatile keyword is used to indicate that a variable can be modified by something external to the program, such as hardware, preventing optimization by the compiler.

160. Explain the difference between a shallow copy and a deep copy.
A shallow copy creates a new object and copies the members' values from the original object, while a deep copy creates a new object and copies the contents of the original object, including any dynamically allocated memory.

161.What are const member functions in C++?
Const member functions are member functions that promise not to modify the state of an object. They are declared using the const keyword.

162.What is a lambda expression in C++?
A lambda expression is an anonymous function used to create function objects on the fly, providing a concise way to define small functions.

163.What is the purpose of the auto keyword in C++11?
The auto keyword allows the compiler to automatically deduce the data type of a variable at compile-time, reducing the need for explicit type declarations.

164. Explain the purpose of the try, throw, and catch keywords in C++ for exception handling.
try is used to enclose code that might throw an exception, throw is used to throw an exception, and catch is used to handle the thrown exception.

165.What is a function object (functor) in C++?
A function object, also known as a functor, is an object that can be invoked as if it were a function due to overloading the operator().

166. Explain the role of reinterpret_cast in C++.
reinterpret_cast is a type of casting operator used for type conversions that are considered unsafe by the compiler. It is used to convert one pointer type to another or to reinterpret the bit pattern of an object.

167.What are the different storage classes in C++?
C++ has four storage classes: auto, register, static, and extern, each with its own scope, lifetime, and storage location characteristics.

168. What is the purpose of the std::move function in C++?
std::move is used to convert a value to an rvalue, allowing the move semantics and efficient transfer of resources, such as ownership of dynamically allocated memory.

169. What is the purpose of the typeid operator in C++?
The typeid operator allows runtime identification of the type of an object or expression.

170. Explain the use of const_cast in C++.
const_cast is used to add or remove const or volatile qualifiers from variables, allowing modifications that would otherwise violate const-correctness.

171. What is the purpose of the std::thread class in C++11?
The std::thread class is used for creating and managing threads in C++, providing support for concurrent execution.

172. Explain the concept of RAII in C++.
RAII (Resource Acquisition Is Initialization) is a programming technique where resource management is tied to object lifetimes, ensuring resources are properly released when the object goes out of scope.

173. What is the difference between std::vector and std::array in C++?
std::vector is a dynamic array that can change size at runtime, while std::array is a fixed-size array with a size determined at compile-time.

174. What is the purpose of the std::unique_ptr in C++?
std::unique_ptr is a smart pointer that manages the memory of a single object and ensures automatic deallocation when it goes out of scope.

175. Explain the concept of function templates in C++.
Function templates allow writing generic functions that can operate with any data type, promoting code reusability.

176. What is the purpose of the std::map in C++?
std::map is a standard associative container that stores elements in key-value pairs and ensures keys are unique.

177. What is a copy constructor in C++?
A copy constructor is a special constructor that initializes a new object with a copy of an existing object of the same class.

178. Explain the concept of move semantics in C++11.
Move semantics allow transferring the resources (such as memory ownership) from one object to another, enabling more efficient resource management.

179. What is a pure virtual function?
A pure virtual function is a virtual function declared in a base class that has no implementation and is meant to be overridden by derived classes. It's denoted by appending = 0 to the function declaration.

180. What are the differences between a shallow copy and a deep copy in C++?
A shallow copy involves copying the memory addresses of the original object's data members, while a deep copy involves creating a new copy of each of the original object's data members.

181. What is the difference between std::shared_ptr and std::unique_ptr?
std::shared_ptr allows multiple pointers to point to the same object, keeping a reference count. std::unique_ptr represents exclusive ownership of an object.

182.Explain the purpose of the mutable keyword in C++.
The mutable keyword is used to declare data members of a class that can be modified even in const member functions.

183.What is the difference between an abstract class and an interface in C++?
An abstract class can have both implemented and unimplemented methods, while an interface only contains method declarations with no implementations.

184. What are the advantages of using references in C++?
References provide a way to alias an existing variable, avoid unnecessary copying, and can be used to pass variables to functions by reference.

185. Explain the concept of function overloading in C++.
Function overloading involves defining multiple functions in the same scope but with different parameter lists. The functions must have different parameters or a different number of parameters.

186. What is the purpose of the std::move function in C++?
std::move is used to cast an lvalue to an rvalue reference, facilitating move semantics and enabling the transfer of resources from one object to another.

187. What is the difference between std::unique_ptr and std::shared_ptr?
std::unique_ptr represents exclusive ownership of a dynamically allocated object and cannot be copied, while std::shared_ptr allows multiple pointers to share ownership of the same object using reference counting.

188. Explain the role of the mutable keyword in C++.
The mutable keyword is used to declare members of a class that can be modified even in const member functions. It allows changing the state of an object marked as const.

189.What is the role of the std::initializer_list in C++?
std::initializer_list is a standard library type introduced in C++11 that allows initialization of objects using a list-like syntax.

190. What is the use of the constexpr specifier in C++?
constexpr specifies that a variable or function can be evaluated at compile-time, enabling computations to be performed during compilation rather than runtime.

191. Explain the role of the decltype keyword in C++.
decltype is used to obtain the type of an expression or a variable. It returns the type without actually evaluating the expression.

192. What are the differences between a stack and a queue in C++?
A stack follows the Last-In, First-Out (LIFO) principle, while a queue follows the First-In, First-Out (FIFO) principle.

193.Explain the purpose of the std::thread class in C++.
std::thread is used to create and manage concurrent threads in C++, allowing multiple parts of a program to run concurrently.

194. What is a lambda expression in C++?
A lambda expression is an anonymous function introduced in C++11 that allows the creation of functions inline without a formal function definition.

195. What is the use of the std::unordered_map in C++?
std::unordered_map is an associative container that stores elements formed by key-value pairs, enabling fast access to elements using a hash-based mechanism.

196. Explain the differences between std::vector and std::list in C++.
std::vector is a dynamic array that allows fast random access, while std::list is a doubly linked list optimized for insertion and deletion of elements.

197. What is a template in C++?
A template is a feature in C++ that allows writing generic classes or functions that can work with any data type without the need for duplication.

198. Explain the purpose of the virtual keyword in C++.
The virtual keyword is used to declare a member function in a base class as virtual, enabling polymorphic behavior and dynamic binding.

199. What is the role of the std::mutex class in C++?
std::mutex is used to provide mutual exclusion to shared resources in a multithreaded environment, preventing simultaneous access by multiple threads.

200. Explain the use of the std::atomic class in C++.
std::atomic is used to provide atomic operations on shared variables, ensuring that operations on these variables are indivisible.

201. What is a forward declaration in C++?
A forward declaration is a declaration of a function, class, or variable before it is defined, allowing its use before the actual definition.

202. What is the role of the std::condition_variable in C++?
std::condition_variable is used in conjunction with mutexes to synchronize threads by waiting for a condition to be met before proceeding.

203. Explain the concept of a range-based for loop in C++.
A range-based for loop is used to iterate through elements in a range, such as arrays, containers, or initializer lists, providing a simpler syntax for iteration.

204. What are the differences between a constructor and a destructor in C++?
A constructor is invoked when an object is created and initializes the object, while a destructor is invoked when an object is destroyed and cleans up resources.

205.Explain the role of the std::move function in C++.
std::move is used to convert an lvalue into an rvalue, facilitating the use of move semantics, which is useful for efficient resource management.

206. What is the purpose of the std::initializer_list in C++11?
std::initializer_list allows initialization of containers, such as arrays or standard containers, using brace-enclosed lists.

207. What are the differences between const and constexpr in C++?
const is used to declare constants that are known at compile time, while constexpr is used to specify that an expression can be evaluated at compile time.

208. Explain the concept of function pointers in C++.
Function pointers store the address of a function, enabling dynamic invocation of functions at runtime.

209. What is the purpose of the std::forward function in C++?
std::forward is used in forwarding references (universal references) to preserve the value category of function arguments.

210. What are the differences between std::set and std::unordered_set in C++?
std::set is an ordered associative container, while std::unordered_set is a hash-based unordered associative container.

211. What is the purpose of the override specifier in C++11?
The override specifier is used to explicitly declare that a function in a derived class overrides a virtual function from a base class.

212. Explain the use of the const qualifier in member functions in C++.
A const member function promises not to modify the object's state and can be invoked on const and non-const objects.

213. What is a dangling pointer in C++?
A dangling pointer is a pointer that points to a memory location that has been deallocated or no longer valid.

214. What is the difference between a stack and a queue in C++?
A stack is a last-in, first-out (LIFO) data structure, while a queue is a first-in, first-out (FIFO) data structure.

215. What are the benefits of using smart pointers in C++?
Smart pointers help manage memory by automatically deallocating memory when it's no longer needed, reducing memory leaks and dangling pointers.

216. Explain the purpose of the std::function class in C++.
std::function is a wrapper class that can store and invoke callable objects, including function pointers, function objects, and lambdas.

217. What is the role of the typename keyword in C++ template programming?
The typename keyword is used to indicate that a dependent name in a template refers to a type.

218.What is the role of the std::pair class in C++?
std::pair is a utility template class that holds two objects or values as a single entity.

219.Explain the role of the std::any class in C++17.
std::any is a class introduced in C++17 that provides a type-safe container for single values of any type.

220.What is the purpose of the std::tuple class in C++?
std::tuple is a standard library class that allows grouping together multiple values of different types into a single object.

221. Explain the differences between std::map and std::unordered_map in C++.
std::map is a sorted associative container based on a red-black tree, while std::unordered_map is a hash table-based associative container.

223. What is function template specialization in C++?
Function template specialization allows defining specialized versions of a function template for specific data types.

224. Explain the purpose of the std::array class in C++.
std::array is a container introduced in C++11 that represents a fixed-size array and provides various container-like functionalities.

225. What is a static_assert in C++?
static_assert is used to perform compile-time assertions, allowing programmers to check conditions at compile-time and generate errors if conditions aren't met.

226. Explain the purpose of the std::chrono library in C++.
std::chrono is a library introduced in C++11 for handling time-related operations, providing utilities for durations, time points, and clocks.

227. What is the role of the decltype(auto) specifier in C++?
decltype(auto) is a feature introduced in C++14 that deduces the type of a variable or expression by using the decltype rules.

228. What are the differences between std::move and std::forward in C++?
std::move casts an lvalue to an rvalue reference, enabling move semantics, while std::forward performs conditional forwarding of references, preserving value categories.

229. Explain the purpose of the std::thread::join() function in C++.
- std::thread::join() is used to wait for a thread to finish its execution before proceeding further in the program.

230. What is the purpose of the std::optional class in C++17?
- std::optional is a class introduced in C++17 that represents an object that may or may not contain a value.

231. Explain the concept of move semantics in C++11.
- Move semantics allow the efficient transfer of resources, such as memory ownership, from one object to another, reducing unnecessary copying.

232. What is the role of the std::tie function in C++?
- std::tie is used to create a tuple of references that can be used to unpack the elements of a tuple or pair.

233. Explain the purpose of the std::algorithm library in C++.
- std::algorithm provides a collection of functions for performing common operations on containers, such as sorting, searching, and modifying elements.

234. What is the purpose of the std::transform function in C++?
- std::transform applies a specific operation to each element in a range and stores the result in another range.

235. Explain the concept of variadic templates in C++.
- Variadic templates allow the creation of functions or classes that accept a variable number of arguments of varying types.

236. What is the purpose of the std::forward_list class in C++?
- std::forward_list is a container introduced in C++11 that implements a singly linked list.

237. Explain the role of the std::mutex class in C++.
- std::mutex is used for providing exclusive access to shared resources among multiple threads.

238. What is the use of the std::next and std::prev functions in C++?
- std::next and std::prev are used to obtain iterators that point to elements succeeding or preceding a specified position in a container.

239. Explain the purpose of the std::async function in C++.
- std::async is used for launching asynchronous tasks and obtaining futures to their results.

240. What is the use of std::filesystem in C++17?
- std::filesystem is a library introduced in C++17 that provides facilities for performing file system operations, such as file/directory manipulation and querying.

241. Explain the purpose of the std::variant class in C++17.
- std::variant is a class introduced in C++17 that represents a type-safe union, allowing objects to hold values of different types.

242. What is the std::byte type in C++17 used for?
- std::byte is a type introduced in C++17 that represents a byte of memory and is used for low-level manipulations and I/O operations.

243. Explain the concept of structured binding in C++17.
- Structured binding is a feature introduced in C++17 that allows unpacking the elements of a tuple-like object into individual variables.

244. What is the purpose of the std::future class in C++?
- std::future is a class used for accessing the result of asynchronous operations and retrieving values from asynchronously executed functions.

245. Explain the differences between std::async, std::thread, and std::promise in C++.
- std::async is used to launch asynchronous tasks, std::thread creates and manages threads, and std::promise provides a mechanism to store a value or an exception that can be retrieved asynchronously.

246. What is a lambda capture in C++?
- Lambda capture specifies how variables from the surrounding scope are accessed within a lambda function, allowing variables to be captured by value or reference.

247. Explain the role of the std::array_view in C++20.
- std::array_view is introduced in C++20 to provide a non-owning view of a contiguous sequence of objects, similar to std::span.

248. What is the purpose of the std::ranges library in C++20?
- std::ranges in C++20 provides a set of range-based algorithms and views, offering a more composable and expressive way to work with sequences of elements.

249. Explain the concept of spaceship operator (<=>) in C++20.
- The spaceship operator is introduced in C++20 for three-way comparison. It returns a value indicating the relationship between two operands, typically for sorting or comparisons.

250. What are concepts in C++20?
- Concepts in C++20 provide a way to specify constraints on template parameters, allowing better error messages and improving template usability.

251. Explain the purpose of the std::source_location in C++20.
- std::source_location in C++20 provides information about the source file, line, column, and function where it's used, aiding debugging and logging.

252. What is the purpose of the std::format function in C++20?
- std::format in C++20 is used for string formatting, providing a type-safe and extensible formatting mechanism.

253. Explain the role of the consteval specifier in C++20.
- consteval is introduced in C++20 to ensure that a function is evaluated at compile-time, similar to constexpr, but with stricter rules.

254. What are the differences between std::unique_lock and std::lock_guard in C++?
- std::unique_lock provides more flexibility than std::lock_guard, allowing locking and unlocking of mutexes at different points, while std::lock_guard locks at construction and unlocks at destruction.

255. What is the purpose of the <charconv> header in C++17?
- <charconv> in C++17 provides functions for converting between character sequences and numeric types, offering improved performance compared to traditional conversion functions.

256. Explain the purpose of the std::jthread in C++20.
- std::jthread in C++20 is a joinable thread that supports cooperative cancellation, allowing a thread to request cancellation from another thread.

257. What are coroutines in C++20?
- Coroutines in C++20 are a language feature that allows writing asynchronous code in a sequential manner, simplifying tasks involving asynchronous operations.

258. What is the role of the std::stop_token in C++20 coroutines?
- std::stop_token is used in C++20 coroutines to check for and request the termination of a coroutine.

259. Explain the purpose of the std::counted_iterator in C++20.
- std::counted_iterator in C++20 is an iterator adaptor that produces a sequence of values by repeating a specific value a certain number of times.