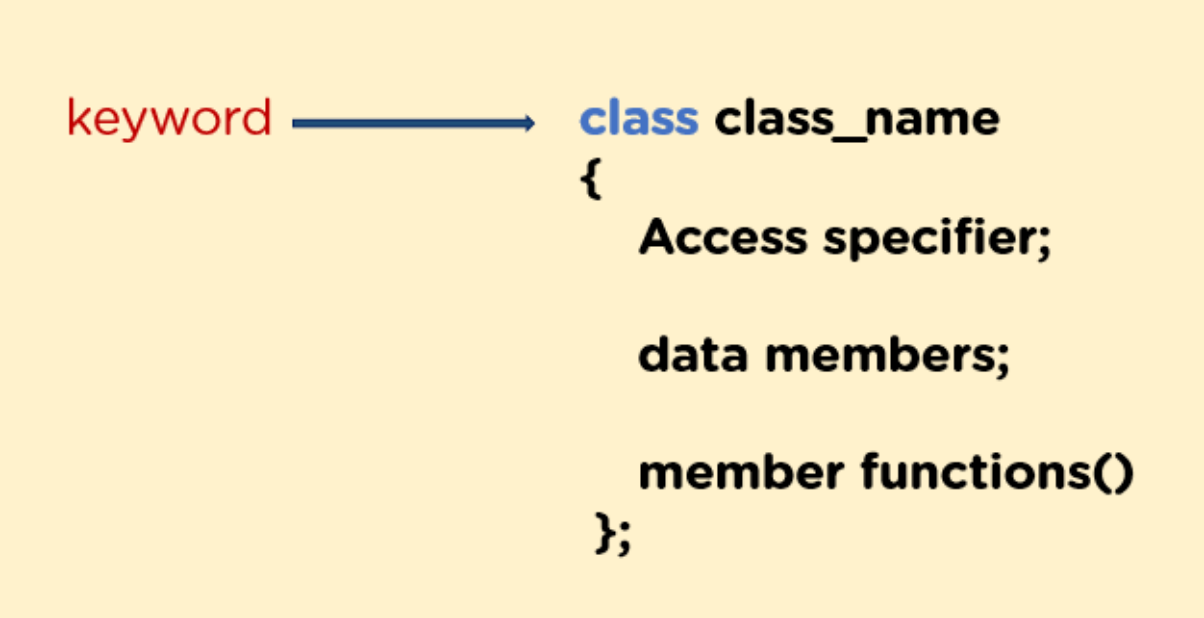
1. What is the difference between C and C++?

C	C++
C is a procedure-oriented programming language	C++ is a partially object-oriented programming language
It follows a top-down approach	It follows a bottom-up approach
C doesn't support function or operator overloading	C++ supports function as well as function overloading
C language doesn't support virtual and friend function	C++ language supports both virtual and friend functions.
C language has 32 keywords	C++ language contains 52 keywords

2) **2. What are classes and objects in C++?**
A class is like a blueprint of an object. It is a user-defined data type with data members and member functions and is defined with the keyword class.



You define objects as an instance of a class. Once it creates the object, then it can operate on both data members and member functions.

3. What are access modifiers?
You use access modifiers to define accessibility for the class members. It defines how to access the members of the class outside the class scope.
There are three types of access modifiers:
Private
Public

Protected

4. Difference between equal to (==) and assignment operator(=)?

The equal to operator == checks whether two values are equal or not. If equal, then it's true; otherwise, it will return false.

The assignment operator = allots the value of the right-side expression to the left operand.

5. What is the difference between a while loop and a do-while loop?

while	do-while
The while loop verifies the condition; if it's true, then it iterates the loop till the condition becomes false.	The do-while loop first iterates the loop body once, then it checks for the condition.

Syntax:
while (condition)
{
statements
}
	Syntax:
	do{
	statements
	}
	while(condition);

If the condition is false in a while loop, then not a single statement will execute inside the loop.	If the condition in a do-while loop is false, then the body will also execute once.

6. What is the size of the int data type?

4 bytes
1 byte
8 bytes
2 bytes
1 - 4 bytes, the integer data type is 4 bytes.

7. Which among the following operators cannot be overloaded?

-
+
?:
%
3 - ?: operator cannot be overloaded because it is not syntactically possible.

8. What among these is used to return the number of characters in the string?

Size
Length
Both size and length
Name
3. Both size and length are used to return the number of characters in the string.

9. Discuss the difference between prefix and postfix?

In prefix (++i), first, it increments the value, and then it assigns the value to the expression.
In postfix (i++), it assigns the value to the expression, and then it increments the variable's value.

10. Can you compile a program without the main function?

Yes, you can compile a program without the main function, but you cannot run or execute the program because the main() function is the entry point, from where all the execution begins. And without the entry point, then you can execute the program.

11. What is std in C++?

std is a standard class in C++
std is a standard file reading header
std is a standard header file
std is a standard namespace
4 - std is a standard namespace in C++

12. What are the four different data types in C++?

Primitive/Basic: Char, int, short, float, double, long, bool, etc.
Derived: Array, pointer, etc.
Enumeration: Enum
User-defined: Structure, class, etc.

13. How is struct different from class?

Structure	Class
Its members are public by default.	Its members are private by default.
The default access specifiers are public when deriving a struct from a class/struct.	The default access specifiers are private when deriving a class.

14. What do you understand about polymorphism in C++?

The term polymorphism refers to the presence of multiple forms. Polymorphism usually occurs when there is a hierarchy of classes that are linked by inheritance.
C++ polymorphism means that depending on the type of object that invokes the function, a different function will be executed.

15. Compare compile time and runtime polymorphism.

Compile-time Polymorphism	**Runtime Polymorphism**
The method to be executed is known at compile time. And the call is resolved by the compiler.	The method to be executed is known at run time. The compiler does not resolve the call.
Provides quicker execution because it is known at the compile time.	Provides slower execution because it is known at the run time.
Achieved by operation or function overloading.	Achieved by function overriding.

16. What is a constructor in C++?

In C++, a function Object is a particular "MEMBER FUNCTION" that shares the same title as the class it belongs to and is used to initialize specific values to an object's data members.

```
#include <iostream>
using namespace std;
class student {
    int no;
public:
    student()
    {
        cout << "Enter the RollNo:";
```

```
        cin >> rno;
    }
    void display()
    {
        cout << endl << rno << "\t";
    }
};
int main()
{
    student s; // constructor gets called automatically when
               // we create the object of the class
    s.display();
    return 0;
}
```

17. What is a virtual function?

A member function in the base class redefined in a derived class is a virtual function. It is declared using the virtual keyword. It ensures that the correct function is called for an object, irrespective of the type of reference/pointer used for the function call. Virtual functions are mainly used for runtime polymorphism.

18. What do you understand about friend class and friend function?

Like a friend function, a friend class can access personal and guarded variables of the type in which it is declared. All member functions for classes specified as friends to another class are friend functions for the friend class.

```
class Node {
private:
int key;
Node* next;
/* Other members of Node Class */
// Now class LinkedList can
// access private members of Node
friend class LinkedList;
};
```

19. What are the three different types of C++ access specifiers?

Public: All member functions and data members are accessible outside the class.
Protected: All member functions and data members are accessible within the class and to the derived class.
Private: All member functions and data members cannot be accessed outside the class.

20. What is an abstraction in C++?

Abstraction means displaying the essential details to the user while hiding the irrelevant or particular details that you don't want to show to a user. It is of two types:
Control abstraction
Data abstraction

21. What are destructors in C++?

A destructor member function is instantly called when an object exits its scope or is specifically deleted by a call to delete.

```
class X {
public:
  // Constructor for class X
  X();
  // Destructor for class X
  ~X();
};
```

22. Is it possible to overload a deconstructor? Give reasons for your answer.

No, it is impossible as destructors do not take arguments or return anything. There has to be only one empty destructor per class. It should have a void parameter list.

23. What is an abstract class? When is it used?

An abstract class is a class whose objects cannot be created. It serves as a parent for the derived classes. Placing a pure virtual function in the class makes it an abstract class.

24. What do you understand about static members and static member functions?

A variable in a class declared as static has its space allocated for the lifetime of the program. Regardless of the number of objects of that class created, there is only a single copy of the static member. The same static member is accessible to all the objects of that class.

A static member function can be called even if no class objects exist. It is accessed using only the class name and the scope resolution operator (::).

25. What is the C++ OOPs concept?

OOPs concept in C++:

Object
Class
Inheritance
Polymorphism
Encapsulation
Abstraction

Object: Anything that exists physically in the real world is called an object.
Class: The collection of objects is called class.
Inheritance: Properties of parent class inherited into child class is known as inheritance.
Polymorphism: It is the ability to exist in more than one form.
Encapsulation: Binding of code and data together into a single unit.
Abstraction: Hiding internal details and showing functionality to the user.

26. When is void() return type used?

You use the void() return type when you don't want to return any value. It specifies that the function doesn't return a value. A function with a void return type completes its task and then returns the control to the caller.

27. What is call by value and call by reference in C++?

In the call by value method, you pass the copies of actual parameters to the function's formal parameters. This means if there is any change in the values inside the function, then that change will not affect the actual values.

In the call-by-reference method, the reference or address of actual parameters is sent to the function's formal parameters. This means any change in values inside the function will be reflected in the actual values.

28. What is an inline function?

An inline function when called expands in line. When you call this function, the whole code of the inline function gets inserted or substituted at the inline function call.

Syntax:

```
 Inline return-type function-name(parameters)
 {
  }
```

29. What are pointers in C++?

Pointers are the variables that store the memory address of another variable. The type of the variable must correspond with the type of pointer.

Syntax: type *name

30. What is a scope resolution operator?

A scope resolution operator is represented as ::
This operator is used to associate function definition to a particular class.
The scope operator is used for the following purposes:
To access a global variable when you have a local variable with the same name.
To define a function outside the class.

31. What should be the output of the following C++ program?

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

int main()
{
    int i=0;
    int x=0;
    do{
        if( i%5 ==0 )
        {
         cout<<x;
         x++;
        }
        ++i;
    }
    while(i<10);

    cout<<x;
    return 0;
}
```

01
02
012
011

3 - 012, the value of i is 0 and i % 5 is equal to 0, so x is displayed and incremented and i is also incremented, then 1 % 5 is not zero, so the condition is not met. Similarly, the process will repeat till 4 % 5 because 5 % 5 is zero. So the value of x is one and gets incremented to two. Then, the process will repeat for 6,7,8,9, but 10 % 10 will be zero again, and the value of x is printed, which is 2.

32. What is a constructor?

A constructor is defined as a member function that is invoked whenever you create an object; it has the same name as that of the class.

There are two types of constructors:

Default constructor: This auto-generated constructor doesn't take any arguments.
Parameterized constructor: In this constructor, it can pass arguments.

33. Define operator overloading and function overloading.

An example of compile-time polymorphism is operator overloading. It is the concept of modifying an existing C++ operator without altering its original meaning.

Let us take an example for this

```
class A
{
};
int main()
{
    A a1,a2,a3;
    a3= a1 + a2;
    return 0;
}
```

34. How to input strings in C++ with spaces?

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

int main()
{
    string s;

    cout << "Enter the sentence with spaces "<<endl;
    getline(cin, s);

    cout << s;
    return 0;
}
```

35. Discuss the difference between new and malloc

new	malloc
new is an operator	malloc() is a function
It calls the constructor	The malloc function doesn't call the constructor

There is no need to specify memory size while using new()	You have to specify the memory size
new operator can be overloaded	malloc() can never be overloaded

36. What is operator overloading?
Operator overloading is a mechanism in which a special meaning is given to an operator. For example, you can overload the '+' operator in a class-like string to concatenate two strings by only using '+.'

37. What is the output of the below C++ program?

```
#include<iostream>

using namespace std;

int main()
{
    enum { blue, green = 5, GREAT };


    cout<<blue<<" "<<GREAT;
}
```

0 3
0 8
0 6
0 2
3 - 0 6. In enum, the element's value is one greater than the previous element. The value of blue is 0 by default, and the value of green is five, so the value of GREAT will become six automatically.

38. What among these is used to return the number of characters in the string?
(i)Size
(ii)Length
(iii)Both size and length
(iv)Name
3 - Both size and length are used to return the number of characters in the string

39. Which among the following statements is correct about the program given below?

```
#include <iostream>
using namespace std;
    int main()
    {
        int x;
        int *p;
        x = 7;
        p = &x;
        cout << *p;
        return 0;
    }
```

- The output will be 7
- The output will be 14
- The output will be 0
- The output will be 1

1 - Output will be 7. Pointer p has the memory address of x, and you display the pointer with a dereference operator that will display the value 7.

41. What is a friend function?

You can define a friend function as a function that can access private, public and protect members of the class. You declare the friend function with the help of the friend keyword. You declare this function inside the class.

42. Which of the following will give the size of object or type?

- sizeof
- malloc
- realloc
- calloc

1 - The sizeof operator is used to give the size of object or type

43. Which of the following is not a member of a class?

- Static function
- Virtual function
- Const function
- Friend function

4 - Among the following, friend function is not a member of the class

43. Which among the following statements is correct about the program given below?

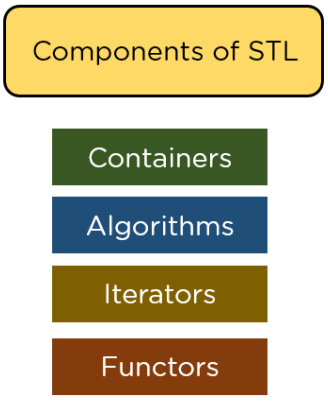
```
#include <iostream>
    using namespace std;
    void duplicate (int& a, int& b, int& c)
    {
        a *= 2;
        b *= 2;
        c *= 2;
    }
    int main ()
    {
        int x = 2, y = 4, z = 5;
        duplicate (x, y, z);
        cout << x << y << z;
        return 0;
    }
```

- The output will be 245
- The output will be 222
- The output will be 4810
- The output will be 4812

3 - Output will be 4810, because you are passing the references of the function here.

44. What is STL?

STL stands for standard template library. It is a library of container templates that provide generic classes and functions.
STL components are containers, algorithms, iterators, and function objects.



45. How to write a program to check if a number is a palindrome or not?

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

int main()
{
    int n, num, digit, rev = 0;
    cout << "Enter a positive number: ";
    cin >> num;
    n = num;

    do
    {
        digit = num % 10;
        rev = (rev * 10) + digit;
        num = num / 10;
    } while (num != 0);

    cout << " The reverse of the number is: " << rev << endl;

    if (n == rev)
        cout << " The number is a palindrome.";
    else
        cout << " The number is not a palindrome.";
    return 0;
}
```

46. What is a copy constructor?

A copy function is a member function that uses another object from the same class to initialize a new thing. A copy function is, to put it simply, a function that produces an object by initializing it with a different object of the same class that has already been constructed.

```
#include <iostream>
using namespace std;
class A
{
  Public:
   int x;
   A(int a) // parameterized constructor.
   {
     x=a;
   }
   A(A &i) // copy constructor
   {
      x = i.x;
   }
};
int main()
```

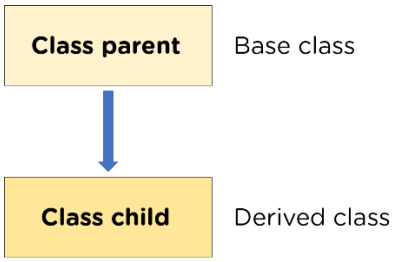
```
{
  A a1(20); // Calling the parameterized constructor.
 A a2(a1); // Calling the copy constructor.
 cout<<a2.x;
  return 0;
}
```

47. Write a program to find the factorial of a number?

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

int main() {
    int n;

    int fact= 1;

    cout << "Enter a positive integer: ";
    cin >> n;

    if (n < 0)
    {
        cout << "Error!";
    }
    else {
        for(int i = 1; i <= n; ++i)
        {
            fact *= i;
        }
        cout << "Factorial of " << n << " = " << fact;
    }
    return 0;
```

48. What is inheritance?

Inheritance is the mechanism in which you can create a new class i.e. child class from the existing class i.e. parent class. This child class is also known as a derived class and the parent class is also called Base class.



49. What is Abstraction?

Abstraction can be defined as a technique in which you only show functionality to the user i.e., the details that you want the user to see, hiding the internal details or implementation details.

50. What should be the output of the below code?

```
#include <iostream>

using namespace std;

int main()
{
    int a=5;
    int b=6;
    int c;

    c= (a>b) ? a : b;
    cout<<c;
    return 0;
}
```

- 5
- 4
- 7
- 6

4 - 6, Ternary operator is used, the value of a is less than b which violates the condition that is why 6 is the answer.

51. How to find the frequency of a number in C++?

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

int frequency(int num, int k)
{
    int c=0;
    while(num>0)
    {
        if(num%10 == k)
        {
            c++;
        }
        num=num/10;
    }
    return c;
}

int main()
{
    int num= 1332422523;

    int k=2;
    cout<<frequency(num, k);
    return 0;
}
```

52. What should be the output of the below code?

```cpp
#include <iostream>

using namespace std;

int main()
{
    int i;
    int j=10;

    i=(j++, j+100, 999+j);

    cout<<i;
    return 0;
}
```

- 1010
- 1001
- 11
- 1000

1 - 1010, the value of j is incremented to 11, then j value is added to 100 but not assigned, and at last the value of j i.e 11 is added to 999 which gives us 1010.

53. What should be the correct statement about string objects in C++?

- String objects should necessarily be terminated by a null character
- String objects have a static size
- String objects have a dynamic size
- String objects use extra memory than required

3 - String objects have a dynamic size.

54. How is a shallow copy different from a deep copy?

Shallow Copy	Deep Copy

It stores the references of objects to the original memory address.	It makes a fresh and separate copy of an entire object and its unique memory address.
Faster	Comparatively slower
It reflects changes made to the new/copied object in the original object.	It doesn’t reflect changes made to the new/copied object in the original object.

55. How are virtual functions different from pure virtual functions?
A virtual function is a base class member function that a derived class can modify. A member function of a base class that is a pure virtual function must be defined in the derived type; otherwise, the derived class will become abstract as well.

```
class
interface Woocommerce{
void print();
}
class Bat implements Woo{
public void print(){System.out.println("Woo!");}
public static void main(String args[]){
Bat obj = new Bat();
obj.print();
 }
}
```

56. Class D is derived from a base class B. If creating an object of type D, what order will the constructors of these classes get called?
The derived class consists of two parts: the base part and the derived part. C++ constructs derived objects in phases. The process begins with constructing the most-base class (at the top of the inheritance tree), followed by each child class construction in order, and then the most-child class. Thus, first, the Constructor of class B will be called, and then the constructor of class D.

57. Can a virtual function be called from a constructor?
A virtual process may be called a function Object, but exercise caution. It might perform differently than expected. The virtual call mechanism in a function Object is disabled since overriding from derived classes hasn't happened yet. Building blocks are used to create objects, "base before derived."

```
class Dog{
void make(){
System.out.println("labrador");
}
}
public class Big extends Dog{
void make(){
System.out.println("Big Dog labrador ");
}
public static void main(String args[]){
Dog ob1 = new Big();
ob1.make();
```

}}

58. What are void pointers?

In C, a void pointer has no connection to any particular data type. It designates a location for specific data within the storage. This indicates that it is pointing to a variable's address. It also goes by the name "general purpose pointer."

```
#include <iostream>
using namespace std;
int main()
{
int a = 10;
char b = 'x';
void* p = &a; // void pointer holds address of int 'a'
p = &b; // void pointer holds address of char 'b'
}
```

59. What is this pointer in C++?

A class, struct, or union form only has access to this pointer within non-static member variables. The arrow shows the object for which the member function is called.

```
string food = "Pizza"; // A food variable of type string
string* ptr = &food; // A pointer variable, with the name ptr, that stores the address of food
// Output the value of food (Pizza)
cout << food << "\n";
// Output the memory address of food (0x6dfed4)
cout << &food << "\n";
// Output the memory address of food with the pointer (0x6dfed4)
cout << ptr << "\n";
```

60. How would you deallocate and allocate memory in C++?

The heap is used in C to allocate dynamic memory, and these functions are part of the standard library. Malloc() and free are the two important dynamic memory operations (). The size of the desired memory area in bytes is the only parameter accepted by the malloc() function.

```
#include <iostream>
#include <cstdlib>
#include <cstring>
using namespace std;
int main() {
  char *user;
  user = (char *) malloc(25);
  strcpy(user, "Jane_Eyre");
  cout << "User Name = " << user << " " << &user << endl;
  free(user);
}
```

61.What do you mean by Call by Value and Call by Reference?
In this programming language to call a function we have 2 methods: Call by Value and Call by Reference

Call by Value	Call by Reference
A copy of a variable is passed.	A variable itself is passed fundamentally.
Calling a function by sending the values by copying variables.	Calling a function by sending the address of the passed variable.
The changes made in the function are never reflected outside the function on the variable. In short, the original value is never altered in Call by Value.	The changes made in the functions can be seen outside the function on the passed function. In short, the original value is altered in Call by reference.
Passed actual and formal parameters are stored in different memory locations. Therefore, making Call by Value a little memory insufficient	Passed actual and formal parameters are stored in the same memory location. Therefore, making Call by Reference a little more memory efficient.

62. Define token in C++
A token is the smallest individual element of a program that is understood by a compiler. A token comprises the following:
Keywords – That contain a special meaning to the compiler
Identifiers – That hold a unique value/identity
Constants – That never change their value throughout the program
Strings – That contains the homogenous sequence of data
Special Symbols – They have some special meaning and cannot be used for another purpose; eg: [] () {}, ; * = #
Operators – Who perform operations on the operand

63.What is the difference between function overloading and operator overloading?

Function Overloading	Operator Overloading
It is basically defining a function in numerous ways such that there are many ways to call it or in simple terms you have multiple versions of the same function	It is basically giving practice of giving a special meaning to the existing meaning of an operator or in simple terms redefining the pre-redefined meaning
Parameterized Functions are a good example of Function Overloading as just by changing the argument or parameter of a function you make it useful for different purposes	Polymorphism is a good example of an operator overloading as an object of allocations class can be used and called by different classes for different purposes
Example of Function Overloading:	Example of Operator Overloading:

Function Overloading	Operator Overloading
int GFG(int X, int Y); int GFG(char X, char Y);	int GFG() = X() + Y(); int GFG() = X() – Y();

64)What is the difference between virtual functions and pure virtual functions?

Virtual Function	Pure Virtual Function
A Virtual Function is a member function of a base class that can be redefined in another derived class.	A Pure Virtual Function is a member function of a base class that is only declared in a base class and defined in a derived class to prevent it from becoming an abstract class.
A virtual Function has its definition in its respective base class.	There is no definition in Pure Virtual Function and is initialized with a pure specifier **(= 0).**
The base class has a virtual function that can be represented or instanced; In simple words, its object can be made.	A base class having pure virtual function becomes abstract that cannot be represented or instanced; In simple words, it means its object cannot be made.