1. What is the difference between OOP and SOP?

Object-Oriented Programming	Structural Programming
Object-Oriented Programming is a type of programming which is based on objects rather than just functions and procedures	Provides logical structure to a program where programs are divided functions
Bottom-up approach	Top-down approach
Provides data hiding	Does not provide data hiding
Can solve problems of any complexity	Can solve moderate problems
Code can be reused thereby reducing redundancy	Does not support code reusability

2. What is Object Oriented Programming?

Object-Oriented Programming(OOPs) is a type of programming that is based on objects rather than just functions and procedures. Individual objects are grouped into classes. OOPs implements real-world entities like inheritance, polymorphism, hiding, etc into programming. It also allows binding data and code together.

3. Why use OOPs?

- OOPs allows clarity in programming thereby allowing simplicity in solving complex problems
- Code can be reused through inheritance thereby reducing redundancy
- Data and code are bound together by encapsulation
- OOPs allows data hiding, therefore, private data is kept confidential
- Problems can be divided into different parts making it simple to solve
- The concept of polymorphism gives flexibility to the program by allowing the entities to have multiple forms

4. What are the main features of OOPs?

- Inheritance
- Encapsulation
- Polymorphism

- Data Abstraction**Classes and Objects OOPs Interview Questions**

5. What is an object?

An object is a real-world entity which is the basic unit of OOPs for example chair, cat, dog, etc. Different objects have different states or attributes, and behaviors.

6. What is a class?

A class is a prototype that consists of objects in different states and with different behaviors. It has a number of methods that are common the objects present within that class.

7. What is the difference between a class and a structure?

Class: User-defined blueprint from which objects are created. It consists of methods or set of instructions that are to be performed on the objects.

Structure: A structure is basically a user-defined collection of variables which are of different data types.

8. Can you call the base class method without creating an instance?

Yes, you can call the base class without instantiating it if:

- It is a static method
- The base class is inherited by some other subclass

9. What is the difference between a class and an object?

Object	Class
A real-world entity which is an instance of a class	A class is basically a template or a blueprint within which objects can be created
An object acts like a variable of the class	Binds methods and data together into a single unit
An object is a physical entity	A class is a logical entity
Objects take memory space when	A class does not take memory space

they are created	when created
Objects can be declared as and when required	Classes are declared just once

OOPs Interview Questions – Inheritance

10. What is inheritance?

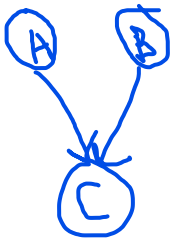
Inheritance is a feature of OOPs which allows classes inherit common properties from other classes. For example, if there is a class such as ‘vehicle’, other classes like ‘car’, ‘bike’, etc can inherit common properties from the vehicle class. This property helps you get rid of redundant code thereby reducing the overall size of the code.

11. What are the different types of inheritance?

- Single inheritance
- Multiple inheritance
- Multilevel inheritance
- Hierarchical inheritance
- Hybrid inheritance

12. What is the difference between multiple and multilevel inheritance?

Multiple Inheritance	Multilevel Inheritance
Multiple inheritance comes into picture when a class inherits more than one base class	Multilevel inheritance means a class inherits from another class which itself is a subclass of some other base class
Example: A class defining a child inherits from two base classes Mother and Father	Example: A class describing a sports car will inherit from a base class Car which inturn inherits another class Vehicle



13. What is hybrid inheritance?

Hybrid inheritance is a combination of multiple and multi-level inheritance.

14. What is hierarchical inheritance?

Hierarchical inheritance refers to inheritance where one base class has more than one subclasses. For example, the vehicle class can have ‘car’, ‘bike’, etc as its subclasses.

15. What are the limitations of inheritance?

- Increases the time and effort required to execute a program as it requires jumping back and forth between different classes
- The parent class and the child class get tightly coupled
- Any modifications to the program would require changes both in the parent as well as the child class
- Needs careful implementation else would lead to incorrect results

16. What is a superclass?

A superclass or base class is a class that acts as a parent to some other class or classes. For example, the Vehicle class is a superclass of class Car.

17. What is a subclass?

A class that inherits from another class is called the subclass. For example, the class Car is a subclass or a derived of Vehicle class.

OOPs Interview Questions – Polymorphism

18. What is polymorphism?

Polymorphism refers to the ability to exist in multiple forms. Multiple definitions can be given to a single interface. For example, if you have a class named Vehicle, it can have a method named speed but you cannot define it because different vehicles have different speed. This method will be defined in the subclasses with different definitions for different vehicles.

19. What is static polymorphism?

Static polymorphism (static binding) is a kind of polymorphism that occurs at compile time. An example of compile-time polymorphism is method overloading.

20. What is dynamic polymorphism?

Runtime polymorphism or dynamic polymorphism (dynamic binding) is a type of polymorphism which is resolved during runtime. An example of runtime polymorphism is method overriding.

21. What is method overloading?

Method overloading is a feature of OOPs which makes it possible to give the same name to more than one methods within a class if the arguments passed differ.

22. What is method overriding?

Method overriding is a feature of OOPs by which the child class or the subclass can redefine methods present in the base class or parent class. Here, the method that is overridden has the same name as well as the signature meaning the arguments passed and the return type.

23. What is operator overloading?

Operator overloading refers to implementing operators using user-defined types based on the arguments passed along with it.

24. Differentiate between overloading and overriding.

Overloading	Overriding
Two or more methods having the same name but different parameters or signature	Child class redefining methods present in the base class with the same parameters/ signature
Resolved during compile-time	Resolved during runtime

OOPs Interview Questions – Encapsulation

25. What is encapsulation?

Encapsulation refers to binding the data and the code that works on that together in a single unit. For example, a class. Encapsulation also allows data-hiding as the data specified in one class is hidden from other classes.

26. What are 'access specifiers'?

that determine the accessibility of methods, classes, etc in OOPs. These access specifiers allow the implementation of encapsulation. The most

common access specifiers are public, private and protected. However, there are a few more which are specific to the programming languages.

27. What is the difference between public, private and protected access modifiers?

Name	Accessibility from own class	Accessibility from derived class	Accessibility from world
Public	Yes	Yes	Yes
Private	Yes	No	No
Protected	Yes	Yes	No

Data abstraction

28. What is data abstraction?

Data abstraction is a very important feature of OOPs that allows displaying only the important information and hiding the implementation details. For example, while riding a bike, you know that if you raise the accelerator, the speed will increase, but you don't know how it actually happens. This is as the implementation details are hidden from the rider.

29. How to achieve data abstraction?

Data abstraction can be achieved through:

- Abstract class
- Abstract method

30. What is an abstract class?

An abstract class is a class that consists of abstract methods. These methods are basically declared but not defined. If these methods are to be used in some subclass, they need to be exclusively defined in the subclass.

Next

31. Can you create an instance of an abstract class?

No. Instances of an abstract class cannot be created because it does not have a complete implementation. However, instances of subclass inheriting the abstract class can be created.

32. What is an interface?

It is a concept of OOPs that allows you to declare methods without defining them. Interfaces, unlike classes, are not blueprints because they do not contain detailed instructions or actions to be performed. Any class that implements an interface defines the methods of the interface.

33. Differentiate between data abstraction and encapsulation.

Data abstraction	Encapsulation
Solves the problem at the design level	Solves the problem at the implementation level
Allows showing important aspects while hiding implementation details	Binds code and data together into a single unit and hides it from the world

To know more about data abstraction, below articles might help you:

Methods and Functions OOPs interview questions

34. What are virtual functions?

Virtual functions are functions that are present in the parent class and are overridden by the subclass. These functions are used to achieve runtime polymorphism.

35. What are pure virtual functions?

Pure virtual functions or are functions that are only declared in the base class. This means that they do not contain any definition in the base class and need to be redefined in the subclass.

36. What is a constructor?

A constructor is a special type of method that has the same name as the class and is used to initialize objects of that class.

37. What is a destructor?

A destructor is a method that is automatically invoked when an object is destroyed. The destructor also recovers the heap space that was allocated to the destroyed object, closes the files and database connections of the object, etc.

38. Types of constructors

differ from language to language. However, all the possible constructors are:

- Default constructor
- Parameterized constructor
- Copy constructor
- Static constructor
- Private constructor

39. What is a copy constructor?

A creates objects by copying variables from another object of the same class. The main aim of a copy constructor is to create a new object from an existing one.

40. What is the use of 'finalize'?

Finalize as an object method used to free up unmanaged resources and cleanup before Garbage Collection(GC). It performs memory management tasks.

41. What is Garbage Collection(GC)?

GC is an implementation of automatic memory management. The Garbage collector frees up space occupied by objects that are no longer in existence.

42. Differentiate between a class and a method.

Class	Method
A class is basically a template that binds the code and data together	Callable set of instructions also called a procedure or function that

into a single unit. Classes consist of methods, variables, etc	are to be performed on the given data

43. Differentiate between an abstract class and an interface?

Basis for comparison	Abstract Class	Interface
Methods	Can have abstract as well as other methods	Only abstract methods
Final Variables	May contain final and non-final variables	Variables declared are final by default
Accessibility of Data Members	Can be private, public, etc	Public by default
Implementation	Can provide the implementation of an interface	Cannot provide the implementation of an abstract class

44. What is a final variable?

A variable whose value does not change. It always refers to the same object by the property of non-transversity.

OOPs Interview Questions – Exception Handling

45. What is an exception?

An exception is a kind of notification that interrupts the normal execution of a program. Exceptions provide a pattern to the error and transfer the error to the exception handler to resolve it. The state of the program is saved as soon as an exception is raised.

46. What is exception handling?

Exception handling in Object-Oriented Programming is a very important concept that is used to manage errors. An exception handler allows errors to be thrown and caught and implements a centralized mechanism to resolve them.

47. What is the difference between an error and an exception?

Error	Exception

Errors are problems that should not be encountered by applications	Conditions that an application might try to catch

48. What is a try/ catch block?

A try/ catch block is used to handle exceptions. The try block defines a set of statements that may lead to an error. The catch block basically catches the exception.

49. What is a finally block?

A finally block consists of code that is used to execute important code such as closing a connection, etc. This block executes when the try block exits. It also makes sure that finally block executes even in case some unexpected exception is encountered.

OOPs Interview Questions – Limitations of OOPs

50. What are the limitations of OOPs?

- Usually not suitable for small problems
- Requires intensive testing
- Takes more time to solve the problem
- Requires proper planning
- The programmer should think of solving a problem in terms of objects

1) What is OOPS?

OOPS is abbreviated as Object Oriented Programming system in which programs are considered as a collection of objects. Each object is nothing but an instance of a class.

2) Write basic concepts of OOPS?

Following are the concepts of OOPS:

1. Abstraction
2. Encapsulation
3. Inheritance
4. Polymorphism

3) What is a class?

A class is simply a representation of a type of object. It is the blueprint/plan/template that describes the details of an object.

4) What is an Object?

An object is an instance of a class. It has its own state, behavior, and identity.

5) What is Encapsulation?

Encapsulation is an attribute of an object, and it contains all data which is hidden. That hidden data can be restricted to the members of that class.

Levels are Public, Protected, Private, Internal, and Protected Internal.

6) What is Polymorphism?

Polymorphism is nothing but assigning behavior or value in a subclass to something that was already declared in the main class. Simply, polymorphism takes more than one form.

7) What is Inheritance?

Inheritance is a concept where one class shares the structure and behavior defined in another class. If Inheritance applied to one class is called Single Inheritance, and if it depends on multiple classes, then it is called multiple Inheritance.

8) What are manipulators?

Manipulators are the functions which can be used in conjunction with the insertion (<<) and extraction (>>) operators on an object. Examples are endl and setw.

9) Explain the term constructor

A constructor is a method used to initialize the state of an object, and it gets invoked at the time of object creation. Rules for constructor are:

- Constructor Name should be the same as a class name.
- A constructor must have no return type.

10) Define Destructor?

A destructor is a method which is automatically called when the object is made of scope or destroyed. Destructor name is also same as class name but with the tilde symbol before the name.

11) What is an Inline function?

An inline function is a technique used by the compilers and instructs to insert complete body of the function wherever that function is used in the program source code.

12) What is a virtual function?

A virtual function is a member function of a class, and its functionality can be overridden in its derived class. This function can be implemented by using a keyword called virtual, and it can be given during function declaration.

A virtual function can be declared using a token(virtual) in C++. It can be achieved in C/Python Language by using function pointers or pointers to function.

13) What is a friend function?

A friend function is a friend of a class that is allowed to access to Public, private, or protected data in that same class. If the function is defined outside the class cannot access such information.

A friend can be declared anywhere in the class declaration, and it cannot be affected by access control keywords like private, public, or protected.

14) What is function overloading?

Function overloading is a regular function, but it can perform different tasks. It allows the creation of several methods with the same name which differ from each other by the type of input and output of the function.

15) What is operator overloading?

Operator overloading is a function where different operators are applied and depends on the arguments. Operator,-,* can be used to pass through the function, and it has its own precedence to execute

16) What is an abstract class?

An abstract class is a class which cannot be instantiated. Creation of an object is not possible with an abstract class, but it can be inherited. An abstract class can contain only an Abstract method. Java allows only abstract method in abstract class while other languages allow non-abstract method as well.

17) What is a ternary operator?

The ternary operator is said to be an operator which takes three arguments. Arguments and results are of different data types, and it depends on the function. The ternary operator is also called a conditional operator.

18) What is the use of finalize method?

Finalize method helps to perform cleanup operations on the resources which are not currently used. Finalize method is protected, and it is accessible only through this class or by a derived class.

19) What are the different types of arguments?

A parameter is a variable used during the declaration of the function or subroutine, and arguments are passed to the function body, and it should match with the parameter defined. There are two types of Arguments.

- Call by Value – Value passed will get modified only inside the function, and it returns the same value whatever it is passed into the function.
- Call by Reference – Value passed will get modified in both inside and outside the functions and it returns the same or different value.

20) What is the super keyword?

The super keyword is used to invoke the overridden method, which overrides one of its superclass methods. This keyword allows to access overridden methods and also to access hidden members of the superclass.

It also forwards a call from a constructor, to a constructor in the superclass.

21) What is method overriding?

Method overriding is a feature that allows a subclass to provide the implementation of a method that overrides in the main class. It will override the implementation in the superclass by providing the same method name, same parameter, and same return type.

22) What is an interface?

An interface is a collection of an abstract method. If the class implements an interface, it thereby inherits all the abstract methods of an interface.

Java uses Interface to implement multiple inheritances.

23) What is exception handling?

An exception is an event that occurs during the execution of a program. Exceptions can be of any type – Runtime exception, Error exceptions. Those exceptions are adequately handled through exception handling mechanism like try, catch, and throw keywords.

24) What are tokens?

A compiler recognizes a token, and it cannot be broken down into component elements. Keywords, identifiers, constants, string literals, and operators are examples of tokens.

Even punctuation characters are also considered as tokens. Example: Brackets, Commas, Braces, and Parentheses.

25) What is the main difference between overloading and overriding?

Overloading is static Binding, whereas Overriding is dynamic Binding. Overloading is nothing but the same method with different arguments, and it may or may not return the equal value in the same class itself.

Overriding is the same method names with the same arguments and return types associated with the class and its child class.

26) What is the main difference between a class and an object?

An object is an instance of a class. Objects hold multiple information, but classes don't have any information. Definition of properties and functions can be done in class and can be used by the object.

A class can have sub-classes, while an object doesn't have sub-objects.

27) What is an abstraction?

Abstraction is a useful feature of OOPS, and it shows only the necessary details to the client of an object. Meaning, it shows only required details for an object, not the inner constructors, of an object. Example – When you

want to switch on the television, it is not necessary to know the inner circuitry/mechanism needed to switch on the TV. Whatever is required to switch on TV will be shown by using an abstract class.

28) What are the access modifiers?

Access modifiers determine the scope of the method or variables that can be accessed from other various objects or classes. There are five types of access modifiers, and they are as follows:

- Private
- Protected
- Public
- Friend
- Protected Friend

29) What are sealed modifiers?

Sealed modifiers are the access modifiers where the methods can not inherit it. Sealed modifiers can also be applied to properties, events, and methods. This modifier cannot be used to static members.

30) How can we call the base method without creating an instance?

Yes, it is possible to call the base method without creating an instance. And that method should be “Static method.”

Doing Inheritance from that class.-Use Base Keyword from a derived class.

31) What is the difference between new and override?

The new modifier instructs the compiler to use the new implementation instead of the base class function. Whereas, Override modifier helps to override the base class function.

32) What are the various types of constructors?

There are three types of constructors:

– Default Constructor – With no parameters.

– Parametric Constructor – With Parameters. Create a new instance of a class and also passing arguments simultaneously.

– Copy Constructor – Which creates a new object as a copy of an existing object.

33) What is early and late Binding?

Early binding refers to the assignment of values to variables during design time, whereas late Binding refers to the assignment of values to variables during run time.

34) What is 'this' pointer?

THIS pointer refers to the current object of a class. THIS keyword is used as a pointer which differentiates between the current object with the global object. It refers to the current object.

35) What is the difference between structure and a class?

The default access type of a Structure is public, but class access type is private. A structure is used for grouping data, whereas a class can be used for grouping data and methods. Structures are exclusively used for data, and it doesn't require strict validation, but classes are used to encapsulate and inherent data, which requires strict validation.

36) What is the default access modifier in a class?

The default access modifier of a class is Internal and the default access modifier of a class member is Private.

37) What is a pure virtual function?

A pure virtual function is a function which can be overridden in the derived class but cannot be defined. A virtual function can be declared as Pure by using the operator =0.

39) What is dynamic or run time polymorphism?

Dynamic or Run time polymorphism is also known as method overriding in which call to an overridden function is resolved during run time, not at the compile time. It means having two or more methods with the same name, same signature but with different implementation.

40) Do we require a parameter for constructors?

No, we do not require a parameter for constructors.

41) What is a copy constructor?

This is a special constructor for creating a new object as a copy of an existing object. There will always be only one copy constructor that can be either defined by the user or the system.

42) What does the keyword virtual represented in the method definition?

It means we can override the method.

43) Whether static method can use nonstatic members?

False.

44) What are a base class, subclass, and superclass?

The base class is the most generalized class, and it is said to be a root class.

A Subclass is a class that inherits from one or more base classes.

The superclass is the parent class from which another class inherits.

45) What is static and dynamic Binding?

Binding is nothing but the association of a name with the class. Static Binding is a binding in which name can be associated with the class during compilation time, and it is also called as early Binding.

Dynamic Binding is a binding in which name can be associated with the class during execution time, and it is also called as Late Binding.

46) How many instances can be created for an abstract class?

Zero instances will be created for an abstract class. In other words, you cannot create an instance of an Abstract Class.

47) Which keyword can be used for overloading?

Operator keyword is used for overloading.

48) What is the default access specifier in a class definition?

Private access specifier is used in a class definition.

49) Which OOPS concept is used as a reuse mechanism?

Inheritance is the OOPS concept that can be used as a reuse mechanism.

50) Which OOPS concept exposes only the necessary information to the calling functions?

Encapsulation