***[Multiple Inheritance Notes in PDF]

What is the use of finalize method in java ?

finalize() method in Java is a method of the Object class that is used to perform cleanup activity before destroying any object. It is called by Garbage collector before destroying the objects from memory. finalize() method is called by default for every object before its deletion.

What is method overriding?
Conclusion. In Java, method overriding occurs when a subclass (child class) has the same method as the parent class. In other words, method overriding occurs when a subclass provides a particular implementation of a method declared by one of its parent classes.

what is the super keyword in java ?

The super() function is used to give access to methods and properties of a parent or sibling class. The super() function returns an object that represents the parent class.

The super keyword refers to superclass (parent) objects. It is used to call superclass methods, and to access the superclass constructor. The most common use of the super keyword is to eliminate the confusion between superclasses and subclasses that have methods with the same name.

What are tokens? in java ?

In Java, tokens are the basic building blocks of a program. They are the smallest individual units of a program that have meaning to the compiler and are used to represent the various elements of a program, such as keywords, identifiers, operators, and literals.

Java Interface ?

Another way to achieve abstraction in Java, is with interfaces.

An interface is a completely "abstract class" that is used to group related methods with empty bodies:

To access the interface methods, the interface must be "implemented" (kinda like inherited) by another class with the implements keyword (instead of extends). The body of the interface method is provided by the "implement" class.

Notes on Interfaces:

- Like abstract classes, interfaces cannot be used to create objects (in the example above, it is not possible to create an "Animal" object in the MyMainClass)
- Interface methods do not have a body - the body is provided by the "implement" class
- On implementation of an interface, you must override all of its methods
- Interface methods are by default abstract and public

- Interface attributes are by default public, static and final
- An interface cannot contain a constructor (as it cannot be used to create objects)

Why And When To Use Interfaces?

1) To achieve security - hide certain details and only show the important details of an object (interface).

2) Java does not support "multiple inheritance" (a class can only inherit from one superclass). However, it can be achieved with interfaces, because the class can implement multiple interfaces. Note: To implement multiple interfaces, separate them with a comma (see example below).

what is an abstraction in java ?

Abstraction in Java is a process of hiding the implementation details from the user and showing only the functionality to the user. It can be achieved by using abstract classes, methods, and interfaces. An abstract class is a class that cannot be instantiated on its own and is meant to be inherited by concrete classes.

what are the access modifiers in java ?

Access modifiers are keywords that can be used to control the visibility of fields, methods, and constructors in a class. The four access modifiers in Java are public, protected, default, and private.

what are sealed modifiers in java ?

Sealed means that only the classes designated by the programmer can inherit from that particular class, thereby restricting access to it. when a class is declared sealed, the programmer must specify the list of classes that can inherit it.

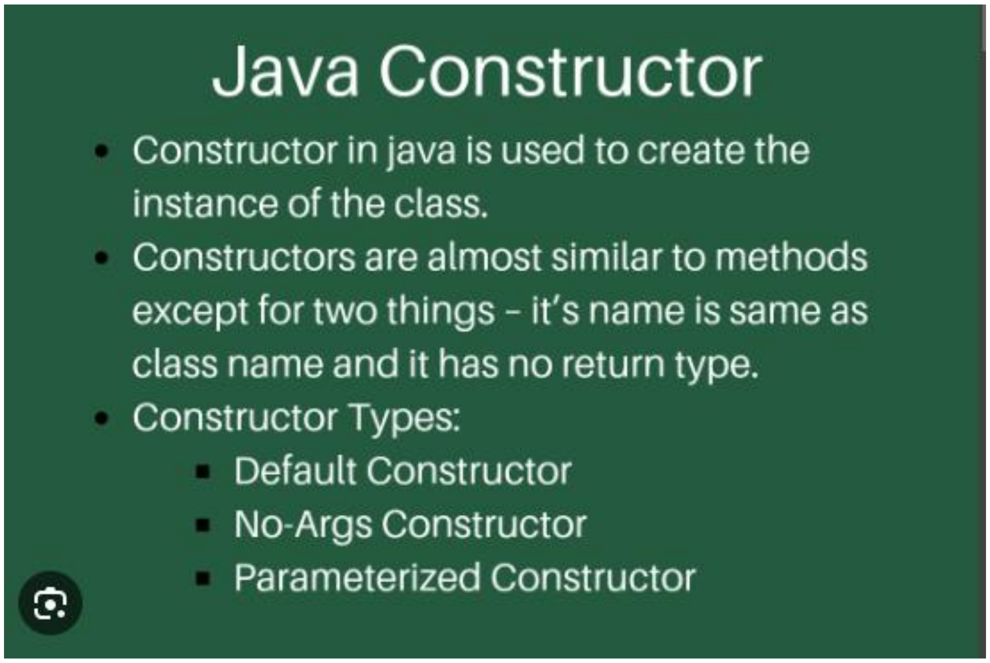
the sealed modifier prevents other classes from inheriting from it.

How can we call the base method without creating an instance?
It is possible if it's a static method. It is possible by inheriting from that class also. It is possible from derived classes using base keyword.

What is the difference between new and override?

The override keyword is used to extend or modify a virtual/abstract method, property, indexer, or event of base class into a derived class. The new keyword is used to hide a method, property, indexer, or event of base class into derived class.

what are the various types of constructors in java ?



Java Constructor

- Constructor in java is used to create the instance of the class.
- Constructors are almost similar to methods except for two things – it's name is same as class name and it has no return type.
- Constructor Types:
  - Default Constructor
  - No-Args Constructor
  - Parameterized Constructor

What is binding in Java?

In java, binding refers to the process of linking a method call to its corresponding method implementation. Binding decides which method implementation is classed when a method is called. Binding can happen either at runtime or at compile time.

Difference between Early and Late Binding in Java ?

Early Binding: The binding which can be resolved at compile time by the compiler is known as static or early binding. Binding of all the static, private and final methods is done at compile-time.

```
public class NewClass {
        public static class superclass {
    }

            static void print()

            {

                    System.out.println("print in superclass.");

            }

        }

        public static class subclass extends superclass {

            static void print()

            {

                    System.out.println("print in subclass.");

            }

        }


        public static void main(String[] args)

        {

            superclass A = new superclass();

            superclass B = new subclass();

            A.print();

            B.print();

        }

}
```

Late binding: In the late binding or dynamic binding, the compiler doesn’t decide the method to be called. Overriding is a perfect example of dynamic binding. In overriding both parent and child classes have the same method

```
public class NewClass {

    public static class superclass {

        void print()

        {

            System.out.println("print in superclass.");

        }

    }


    public static class subclass extends superclass {

        @Override

        void print()

        {

            System.out.println("print in subclass.");

        }

    }


    public static void main(String[] args)
```

```
        {
                superclass A = new superclass();
                superclass B = new subclass();
                A.print();
                B.print();
        }
}
```

Difference table between early and late binding:

Early Binding	Late Binding
It is a compile-time process	It is a run-time process
The method definition and method call are linked during the compile time.	The method definition and method call are linked during the run time.
Actual object is not used for binding.	Actual object is used for binding.
For example: Method overloading	For example: Method overriding

Early Binding	Late Binding
Program execution is faster	Program execution is slower

what is 'this' pointer in java ?

The this keyword refers to the current object in a method or constructor. The most common use of the this keyword is to eliminate the confusion between class attributes and parameters with the same name (because a class attribute is shadowed by a method or constructor parameter).

What is the difference between structure and a class?
Structures and classes differ in the following particulars: Structures are value types; classes are reference types. A variable of a structure type contains the structure's data, rather than containing a reference to the data as a class type does. Structures use stack allocation; classes use heap allocation.

What is the default access modifier in a class?

When no access modifier is specified for a class, method, or data member – It is said to be having the **default** access modifier by default. The data members, classes, or methods that are not declared using any access modifiers i.e. having default access modifiers are accessible **only within the same package**.

What is a pure virtual function with an example?

A pure virtual function is a function that must be overridden in a derived class and need not be defined.

What is difference between virtual function and pure virtual function?

A virtual function is a member function in a base class that can be redefined in a derived class. A pure virtual function is a member function in a base class whose declaration is provided in a base class and implemented in a derived class.

what are a base class subclass and superclass in java ?

A class that is derived from another class is called a subclass (also a derived class, extended class, or child class). The class from which the subclass is derived is called a superclass (also a base class or a parent class).

Which keyword can be used for overloading?
If both parent & child classes have the same method, then the child class would override the method available in its parent class. By using the super keyword we can take advantage of both classes (child and parent) to achieve this. We create an object of child class as it can inherit the parent class methods.

How many instances can be created for an abstract class?

The answer to the question of how many instances of an abstract class can be created is zero. That is, we cannot create an instance of an abstract class as it does not have any complete implementation. An abstract class acts like a template or an empty structure.

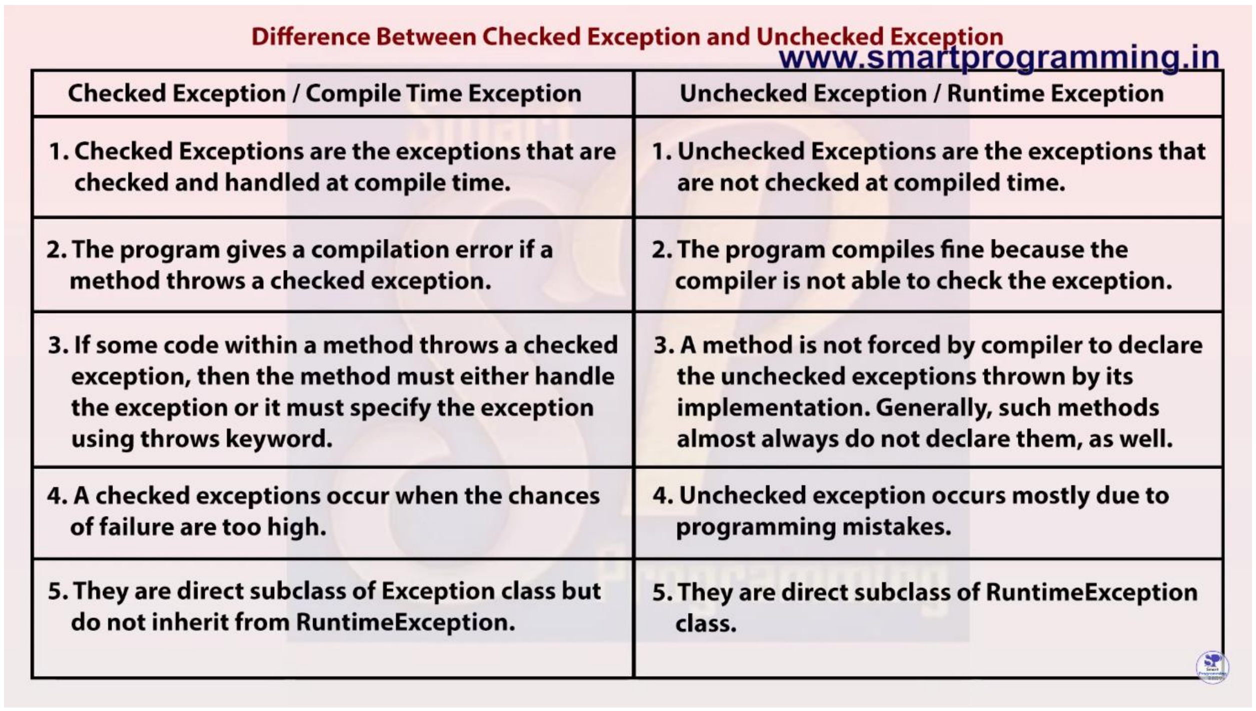
Which OOPS concept is used as a reuse mechanism?

Inheritance is the object-oriented programming concept where an object is based on another object. Inheritance is the mechanism of code reuse. The object that is getting inherited is called the superclass and the object that inherits the superclass is called a subclass.

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

Data hiding is the concept of oops which means exposing only necessary information to clients.

Difference Between Checked and unchecked Exception?



Difference Between Checked Exception and Unchecked Exception

Checked Exception / Compile Time Exception	Unchecked Exception / Runtime Exception
1. Checked Exceptions are the exceptions that are checked and handled at compile time.	1. Unchecked Exceptions are the exceptions that are not checked at compiled time.
2. The program gives a compilation error if a method throws a checked exception.	2. The program compiles fine because the compiler is not able to check the exception.
3. If some code within a method throws a checked exception, then the method must either handle the exception or it must specify the exception using throws keyword.	3. A method is not forced by compiler to declare the unchecked exceptions thrown by its implementation. Generally, such methods almost always do not declare them, as well.
4. A checked exceptions occur when the chances of failure are too high.	4. Unchecked exception occurs mostly due to programming mistakes.
5. They are direct subclass of Exception class but do not inherit from RuntimeException.	5. They are direct subclass of RuntimeException class.

ASPECT	JDK (Java Development Kit)	JRE (Java Runtime Environment)	JVM (Java Virtual Machine)
Purpose	For Java development	For running Java applications	Executes Java bytecode
Components Included	Yes, includes JRE components	Yes, includes JVM and libraries	No, it's a part of JRE
Contains	Compiler, debugger, libraries, and other development tools	JVM, libraries, runtime environment	Interpreter and runtime environment for Java bytecode
Usage	Used for creating Java applications	Required to execute Java programs	Executes Java bytecode on a computer
Development	Essential for development	Not necessary for development	Not directly involved in development
Dependency	Dependent on system architecture	Dependent on operating system	Dependent on the platform