(Finals) Chapter 7: Introduction to Inheritance

Study online at https://quizlet.com/_eyazcv

Inheritance	- Allows one class to inherit properties and methods from another class. - allows a software developer to derive a new class from an existing one
parent class	The class that is being inherited from is called the ___ class
child class	The class that is inheriting the properties and methods is called the ___ class.
- Superclass (formal) - Parent Class (informal) - Base Class (Synonym)	What is the formal, informal, and synonym form of abstract class?
- Subclass (formal) - Child Class (informal) - Derived Class (Synonym)	What is the formal, informal, and synonym form of concrete class?
Base class	- A class that is used as a starting point for creating derived classes. It is sometimes referred to as a superclass or parent class. - defines a set of common properties and methods that can be inherited by its derived classes.
Derived class	- A class that is created by inheriting properties and methods from a base class. It is sometimes referred to as a subclass or child class. - can add its own unique properties and methods to the ones it inherits from its base class.
Code reuse	importance of inheritance that can save time and effort by reducing the need to write redundant code.
Hierarchical relationships	importance of inheritance that can help to organize and structure code, making it easier to understand and maintain.
Polymorphism	importance of inheritance that allows objects of different classes to be treated as if they were of the same class, which can lead to more flexible and extensible code.
True	True or False? - Inheritance can make code easier to maintain and update. It can help to create a better design for software systems by promoting modularity and encapsulation.
Robust software	- refers to software that is able to function properly and effectively even when faced with unexpected or challenging situations. - It can handle errors, exceptions, and unexpected inputs without crashing or causing other problems. It is also able to recover from failures quickly and seamlessly.
Scalable software	- refers to software that can handle increasing amounts of work or data without sacrificing performance or functionality - It can be easily expanded or adapted to meet changing needs and requirements. It is also able to distribute workloads across multiple resources or servers, allowing it to handle large volumes of traffic or data.
False (is not)*	True or False? - the Protected access modifier is accessible from objects outside class
True	True or False? - the Public access modifier is accessible from own class
True	True or False? - the Private access modifier is not accessible from derived class
extends	keyword that is used to create a subclass (also known as a child class) that inherits properties and methods from a superclass (also known as a parent class)

super	In Java, the '___' keyword is a reference variable that is used to refer to the immediate parent class of a subclass. It is often used to call a method in the superclass that has been overridden by the subclass.
1. To call a constructor of the parent class 2. To call a method in the parent class 3. To refer to a variable in the parent class	Give some ways in which you can use the super keyword in Java.
Method overriding	- is a feature of object-oriented programming where a subclass provides its own implementation of a method that is already defined in its parent class. - In other words, the subclass provides a different implementation of the method, but with the same signature (name, return type, and parameters) as the parent class method.
Method overloading	is another feature of object-oriented programming where multiple methods can have the same name, but with different parameters.

(Finals) Chapter 8: Implementation of Inheritance

Study online at https://quizlet.com/_eyb2es

1. Single inheritance 2. Multiple inheritance 3. Multilevel inheritance 4. Hierarchical inheritance 5. Hybrid inheritance	Enumerate the 5 Types of Inheritance.
Single inheritance	- A type of inheritance which is the simplest form of inheritance, where a derived class inherits properties and methods from a single base class.
Multiple inheritance	- In this type of inheritance, a derived class can inherit properties and methods from multiple base classes. This can lead to complex class hierarchies and potential conflicts, but it can also be very useful in certain situations.
Multilevel inheritance	- In this type of inheritance, a derived class inherits properties and methods from a base class, which in turn inherits from another base class. This creates a hierarchy of classes, with each class building on the properties and methods of its parent class.
Hierarchical inheritance	- In this type of inheritance, multiple derived classes inherit properties and methods from a single base class. Each derived class can add its own unique properties and methods to the ones it inherits from the base class.
Hybrid inheritance	- In this type of inheritance, a combination of multiple inheritance and hierarchical inheritance is present. In this type of inheritance, a class inherits properties and methods from multiple base classes, and those base classes themselves inherit from a single base class.

(Finals) Chapter 9: Abstract class and Interface

Study online at https://quizlet.com/_eyb3bn

Abstract methods	___ methods are mostly declared where two or more subclasses are also doing the same thing in different ways through different implementations. It also extends the same Abstract class and offers different implementations of the abstract methods.
Abstraction	___ is a principle in Java and refers to the process of hiding complex implementation details and only exposing the necessary features of a class to the user. - It is achieved in Java through the use of abstract classes and interfaces.
Abstract classes	- are classes that cannot be instantiated, meaning you cannot create objects directly from them. These classes are typically used to define a common interface or set of behaviors that multiple related classes should implement. - are classes that can have abstract methods.
True	True or False? - Abstract classes are meant to be extended by other classes, which can then provide their own implementation for any abstract methods declared in the abstract class.
abstract	An abstract class is declared using the ___ keyword before the class definition.

abstract class [className] {
//class body
}
	What is the syntax in making an abstract class?

[accessModifer] abstract [returnType] [methodName]( );	What is the syntax in making an abstract method?
concrete classes	- are classes that can be instantiated, meaning you can create objects directly from them. They can also provide their own implementation for any methods declared in the class. - are typically used to represent specific objects or concepts in a program.
False (does not need)*	True or False? - A concrete class does need the abstract keyword.
Non-abstract method	is a method that has a complete implementation in the class where it is declared. This means that the method provides a specific set of instructions or code that is executed when the method is called.
concrete methods	Non-abstract methods are also known as ___ methods.
Abstract method	is a method declared without an implementation. An abstract class is a class that can have abstract methods. Abstract methods are meant to be overridden by concrete (non-abstract) subclasses.
True	True or False? - Abstract methods are meant to be overridden by concrete (non-abstract) subclasses.
Interface	- is a collection of abstract methods and constants that can be implemented by a class. It defines a set of methods that a class implementing that interface must provide. - is a blueprint that can be used to implement a class.
True	True or False? - The interface does not contain any concrete methods (methods that have code). All the methods of an interface are abstract methods.
True	True or False? - An interface cannot be instantiated. However, classes that implement interfaces can be instantiated.
- Abstraction - Multiple inheritance - loose coupling	Interfaces are used for a number of reasons: (Enumerate)

(Finals) Chapter 9: Abstract class and Interface

Study online at https://quizlet.com/_eyb3bn

True	True or False? - A class can only extend one other class, but it can implement any number of interfaces.
True	True or False? - loose coupling in our code means that our classes are less dependent on each other, and our code is easier to maintain.
implements	To implement an interface in Java, a class must use the ___ keyword.
- Flexibility - Maintainability - Testability	There are a number of benefits to using interfaces in Java: (Enumerate)

Chapter 11 - Polymorphism

Study online at https://quizlet.com/_f3wn55

Polymorphism	is a concept in object-oriented programming that allows objects of different classes to be used interchangeably, while still maintaining their own unique behavior.
True	True or False? - Polymorphism allows to use a single interface to represent different objects that share a common behavior.
"many"	What does Poly mean?
"Take different forms"	What does Morph mean?
Static polymorphism	- also known as compile-time polymorphism, refers to the ability of the compiler to determine which method to call at compile time based on the arguments passed to the method. - is achieved through method overloading
Dynamic polymorphism	- also known as runtime polymorphism, refers to the ability of objects to respond to the same message with the appropriate method based on their class definition at runtime. - •is achieved through method overriding
Compile-time polymorphism / Static Polymorphism	___ polymorphism allows us to use many methods with the same name but differing signatures and return types.
Run-time polymorphism / Dynamic Polymorphism	___ polymorphism is associated with different classes, but it allows us to use the same method with different signature names.
- Method overloading - Constructor overloading - Method hiding	What are the examples of Static Polymorphism?
- Method overriding	Give an example of Dynamic Polymorphism
Binding	- refers to the process of associating a method or function call with its implementation at runtime. - The connecting (linking) between a method call and method body/definition.
1. Static binding 2. Dynamic binding	What are the two types of binding in Java?
Static binding	- Occurs when the method to be called is determined at compile time. - This binding is also known as early binding because it takes place before the program actually runs.
Dynamic binding	- Also known as late binding or runtime binding, occurs when the method to be called is determined at runtime.

Chapter 12 - Modularity

Study online at https://quizlet.com/_f3wpio

package	A ___ in Java is a collection of related classes and interfaces that are grouped together. It helps to organize code and prevent naming conflicts. They provide a hierarchical naming structure that can be used to group related functionality together.
interface	An ___ is a collection of abstract methods that define a contract for a class to implement
Classes	___ in Java are the building blocks of OOP. They encapsulate data and behavior into a single unit, providing a blueprint for creating objects.
Modularity	is a fundamental concept in object-oriented programming (OOP) that refers to the practice of breaking down a large program into smaller, more manageable pieces, or modules.
1. Built-in Packages 2. User-defined Packages	What are the two types of packages in Java?
Built-in Packages	These packages are provided by the Java language itself and contain classes and interfaces that are fundamental to the Java platform.
User-defined Packages	These packages are created by developers to organize their classes and interfaces into logical groups. These packages can contain other packages, as well as classes and interfaces
reverse domain name convention	User-defined packages are typically organized using the ___ name convention, such as com.example.mypackage.
Single Type Import	- a type of import which imports a single class or interface
Type on Demand Import	- a type of import which imports all the accessible classes and interfaces present in a package
Static Import	- a type of import which is used to import static members of a class.
Static Type on Demand Import	- a type of import which imports all the static members of a class or interface
Single Static Import	- a type of import which imports a single static member of a class or interface
Encapsulation	is one of the four fundamental principles of object-oriented programming (OOP) in Java. It refers to the concept of hiding the internal details of an object from the outside world and allowing access to it only through a set of public methods.
access modifiers	Encapsulation is achieved through the use of ___ such as private, public, and protected. These keywords control the visibility of the variables and methods within a class.
Private	___ variables and methods can only be accessed within the same class.
Public	___ variables and methods, on the other hand, can be accessed by any other class.
Protected	___ variables and methods can be accessed within the same package or by any subclass of the class that defines the protected member.