Object Oriented Programming

Topic: Inheritance

MCQs BANK No.: 6

Instructions:

This MCQs Bank contains question and solution on adjacent(even-odd) pages. First try to solve the MCQ by yourself, then look for the solution.

Best viewed in “single page view” in PDF viewer.

MCQ No: 1

_______________ allows the creation of hierarchical classifications in object-oriented programming and helps to create a general class that defines traits common to a set of related items. Fill in the blank.

a) Encapsulation
b) Inheritance
c) Data Hiding
d) Polymorphism

MCQ No: 1 (Solution)

Ans: b) Inheritance

Explanation: Inheritance allows the creation of hierarchical classifications in object-oriented programming. Using inheritance, you can create a general class that defines traits common to a set of related items. This class can then be inherited by other, more specific classes, each adding those things that are unique to it. The idea of inheritance is simple but powerful: When you want to create a new class and there is already a class that includes some of the code that you want, you can derive your new class from the existing class. In doing this, you can reuse the fields and methods of the existing class without having to write (and debug!) them yourself..

MCQ No: 2

In the terminology of Java, a class that is inherited is called a _______________. Fill in the blank.

a) superclass
b) subclass
c) constructor class
d) destructor class

MCQ No: 2 (Solution)

Ans: a) superclass

Explanation: In the terminology of Java, a class that is inherited is called a superclass. 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).

MCQ No: 3

In the terminology of Java, a class that does the inheriting is called a __________. Fill in the blank.

a) superclass
b) subclass
c) constructor class
d) destructor class

MCQ No: 3 (Solution)

Ans: b) subclass

Explanation:
In the terminology of Java, a class that does the inheriting is called a subclass. 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).

MCQ No: 4

Excepting Object, which has no superclass, every class has one and only one direct superclass (single inheritance). In the absence of any other explicit superclass, every class is implicitly a subclass of Object. True or False

a) True
b) False

MCQ No: 4 (Solution)

Ans: a) True

Explanation:
Excepting Object, which has no superclass, every class has one and only one direct superclass (single inheritance). In the absence of any other explicit superclass, every class is implicitly a subclass of Object.
Classes can be derived from classes that are derived from classes that are derived from classes, and so on, and ultimately derived from the topmost class, Object. Such a class is said to be descended from all the classes in the inheritance chain stretching back to Object.

MCQ No: 5

The subclass inherits members of the superclass and hence promotes ________. Fill in the blank.

a) polymorphism
b) code reuse
c) code rewrite
d) multiple inheritance

MCQ No: 5 (Solution)

Ans: b) code reuse

Explanation:
The subclass inherits members of the superclass and hence promotes code reuse. The subclass itself can add its own new behavior and properties.

MCQ No: 6

Private members of the superclass are not inherited by the subclass. True or False

a) True
b) False

MCQ No: 6 (Solution)

Ans: a) True

Explanation:
Private members of the superclass are not inherited by the subclass and can only be indirectly accessed(using public methods).
A subclass inherits all of the public and protected members of its parent, no matter what package the subclass is in.

MCQ No: 7

A subclass must use the ____________ keyword to derive from a super class which must be written in the header of the subclass definition. Fill in the blank.

a) extends
b) import
c) static
d) super

MCQ No: 7 (Solution)

Ans: a) extends

Explanation:
A subclass must use the extends keyword to derive from a super class which must be written in the header of the subclass definition.

MCQ No: 8

Since constructors and initializer blocks are not members of a class, they are not inherited by a subclass. True or False

a) True
b) False

MCQ No: 8 (Solution)

Ans: a) True

Explanation:
A subclass inherits all the members (fields, methods, and nested classes) from its superclass. Constructors are not members, so they are not inherited by subclasses, but the constructor of the superclass can be invoked from the subclass.

MCQ No: 9

You can write a subclass constructor that invokes the constructor of the superclass, either implicitly or by using the keyword __________. Fill in the blank.

a) default
b) super
c) extends
d) final

MCQ No: 9 (Solution)

Ans: b) super

Explanation: A parent class constructor is not inherited in child class and this is why super() is added automatically in child class constructor if there is no explicit call to super or this.

MCQ No: 10

You can write a new instance method in the subclass that has the same signature as the one in the superclass, this is called ____________________. Fill in the blank

a) method calling
b) method overriding
c) method overloading
d) method hiding

MCQ No: 10 (Solution)

Ans: b) method overriding

Explanation: You can write a new instance method in the subclass that has the same signature as the one in the superclass, thus overriding it. If subclass (child class) has the same method as declared in the parent class, it is known as method overriding in Java. In other words, If a subclass provides the specific implementation of the instance method that has been declared by one of its parent class, it is known as method overriding.

MCQ No: 11

You can write a new static method in the subclass that has the same signature as the one in the superclass, this is called ________________. Fill in the blank.

a) method calling
b) method overriding
c) method overloading
d) method hiding

MCQ No: 11 (Solution)

Ans: d) method hiding

Explanation:
Method hiding may happen in any hierarchy structure in java. When a child class defines a static method with the same signature as a static method in the parent class, then the child's method hides the one in the parent class.

MCQ No: 12

You can declare new methods in the subclass that are not in the superclass. True or False

a) True
b) False

MCQ No: 12 (Solution)

Ans: a) True

Explanation:
You can declare new methods in the subclass that are not in the superclass. The inherited methods can be used directly as they are. You can write a new instance method in the subclass that has the same signature as the one in the superclass, thus overriding it.

MCQ No: 13

A subclass can call a constructor defined by its superclass by use of ___________. Fill in the blank.

a) super(parameter-list);
b) super class name
c) constructor(parameter-list);
d) default(parameter-list);

MCQ No: 13 (Solution)

Ans: a) super(parameter-list);

Explanation:
A subclass can call a constructor method defined by its superclass by use of the following form of super:
super(parameter-list);
Here, parameter-list specifies any parameters needed by the constructor in the superclass.
super( ) must always be the first statement executed inside a subclass' constructor.

MCQ No: 14

If we want to refer a member of the superclass of a subclass, we have to use ________________. Fill in the blank.

a) super.member
b) default.member
c) parent.member
d) base-class.member

MCQ No: 14 (Solution)

Ans: a) super.member

Explanation: The second form of super acts somewhat like this, except that it always refers to the superclass of the subclass in which
it is used. This usage has the following general form:
super.member
Here, member can be either a method or an instance variable. This second form of super is most applicable to situations in which member names of a subclass hide members by the same name in the superclass.

MCQ No: 15

Which of the following correctly explains multi-level inheritance.

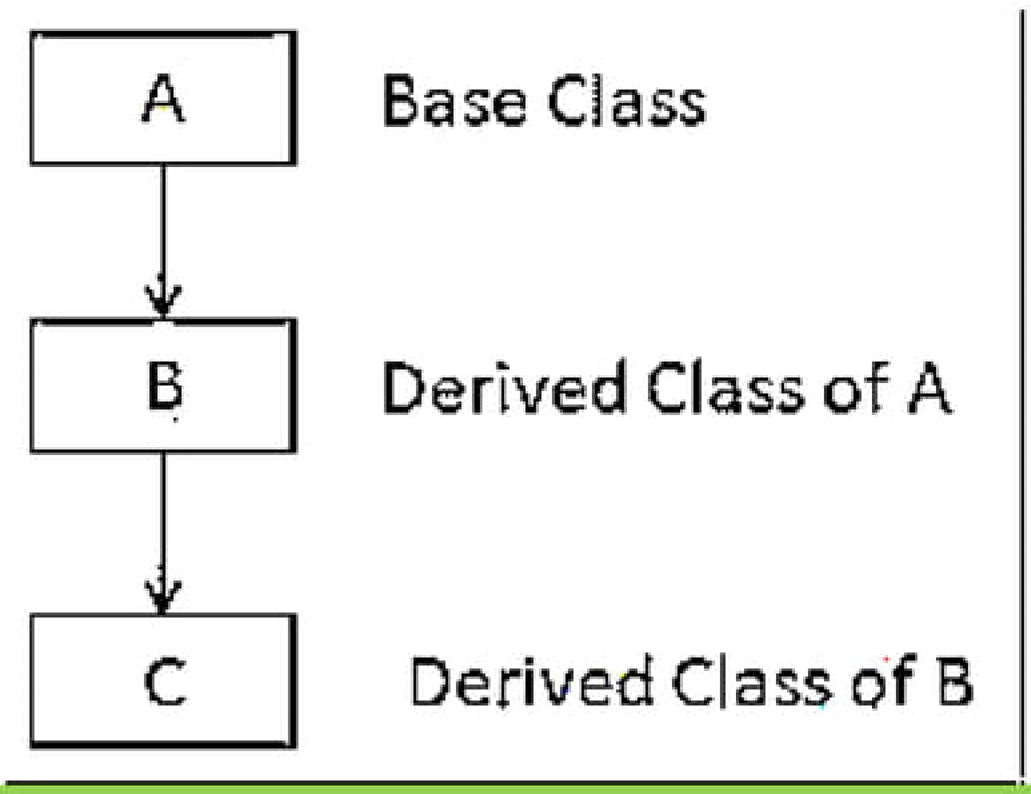
a) one class extends one class only
b) the ladder of single inheritance increases
c) one class directly extends more than one class
d) none of these

MCQ No: 15 (Solution)

Ans: b) the ladder of single inheritance increases

Explanation:

When a class extends a class, which extends another class then this is called multilevel inheritance. For example class C extends class B and class B extends class A then this type of inheritance is known as multilevel inheritance.



MCQ No: 16

Java supports multiple inheritance partially through ________________ . Fill in the blank.

a) inheritance
b) interfaces
c) multiple packages
d) multi-level inheritance

MCQ No: 16 (Solution)

Ans: b) interfaces

Explanation:
In multiple inheritance, one class extends multiple classes. Java does not support multiple inheritance but C++ supports. Java supports multiple inheritance partially through interfaces.

MCQ No: 17

If one class is extended by many subclasses, then the type of inheritance is called ___________ . Fill in the blank.

a) Multi-Level Inheritance
b) Hierarchical Inheritance
c) Multiple Inheritance
d) Single Inheritance

MCQ No: 17 (Solution)

Ans: b) Hierarchical Inheritance

Explanation: In hierarchical type of inheritance, one class is extended by many subclasses. It is one-to-many relationship. A realtime example is available at dynamic binding. For example, Bird class is extended by two classes – Parrot and Vulture. Both classes can make use of the methods of Bird. Even though the Parrot and Vulture are subclasses of the same class Bird, but still they cannot makeuse of each other members. Parrot and Vulture are known as "siblings". Siblings are disjoint and they cannot make use of other members as between them no inheritance is involved (like two sons of a father; one son's property cannot be shared by other but both can share the property of father).

MCQ No: 18

Which of the following are the disadvantages of Inheritance

a) Both classes (super and subclasses) are tightly-coupled.

b) As they are tightly coupled (binded each other strongly with extends keyword), they cannot work independently of each other.

c) Changing the code in super class method also affects the subclass functionality.

d) All of the above.

MCQ No: 18 (Solution)

Ans: d) All of the above.

Explanation:

Disadvantages of Inheritance

1. Both classes (super and subclasses) are tightly-coupled.
2. As they are tightly coupled (binded each other strongly with extends keyword), they cannot work independently of each other.
3. Changing the code in super class method also affects the subclass functionality.
4. If super class method is deleted, the code may not work as subclass may call the super class method with super keyword. Now subclass method behaves independently.

MCQ No: 19

Defining a method in the subclass that has the same name, same arguments and same return type as a method in the superclass and it hides the super class method is called .

a) method rebuilding
b) method overriding
c) method overloading
d) method calling

MCQ No: 19 (Solution)

Ans: b) method overriding

Explanation:
Defining a method in the subclass that has the same name, same arguments and same return type as a method in the superclass and it hides the super class method is called method overriding. Now when the method is called, the method defined in the subclass is invoked and executed instead of the one in the superclass.

MCQ No: 20

Method overriding is run time polymorphism. True or False

a) True
b) False

MCQ No: 20 (Solution)

Ans: a) True

Explanation:
Defining a method in the subclass that has the same name, same arguments and same return type as a method in the superclass is called method overriding.
Method overriding is an example of runtime polymorphism. In method overriding, a subclass overrides a method with the same signature as that of in its superclass. During compile time, the check is made on the reference type.

MCQ No: 21

Method overloading is compile time polymorphism.

a) True
b) False

MCQ No: 21 (Solution)

Ans: a) True

Explanation: Overloading a method is a way to provide more than one method in one class which have same name but different argument to distinguish them. Method overloading is compile time polymorphism.
Method overloading is the compile-time polymorphism where more than one methods share the same name with different parameters or signature and different return type. Method overriding is the runtime polymorphism having same method with same parameters or signature, but associated in different classes.

MCQ No: 22

_________________ is the mechanism by which a call to an overridden method is resolved at run time, rather than compile time. Fill in the blank.

a) Dynamic method dispatch
b) Static class select
c) Dynamic class select
d) Static method dispatch

MCQ No: 22 (Solution)

Ans: a) Dynamic method dispatch

Explanation: Dynamic method dispatch is the mechanism by which a call to an overridden method is resolved at run time, rather than compile time. Dynamic method dispatch is important because this is how Java implements run-time polymorphism.
Overridden method to execution based upon the type of the object being referred to at the time the call occurs. Thus, this determination is made at run time. In other words, it is the type of the object being referred to (not the type of the reference variable) that determines which version of an overridden method will be executed.