

Unit -2 Two Marks Questions with Answers

Q.1 Define the term inheritance.

Ans. Inheritance is a process of deriving a new class from existing class, also called as “extending a class”. When an existing class is extended, the new (inherited) class has all the properties and methods of the existing class and also possesses its own characteristics.

Q.2 Explain the use of extend keyword with suitable example.

Ans;

- The extend keyword is used in inheritance.
- When the child class is derived from its parent class then the keyword extends is used.

Q.3 What is the difference between superclass and subclass?

Ans. :

- The superclass is a parent class or base class from which another class can be derived.
- The subclass is always a derived class which inherits the properties of the base class or superclass.
- The superclass is normally a generalized class whereas the subclass is normally for specific purpose.

Q.4 Enlist various forms of inheritance.

Ans. Various forms of inheritance are -

1. Single level inheritance
2. Multiple inheritance
3. Multi-level inheritance
4. Hierarchical inheritance
5. Hybrid inheritance

Q.5 What is the use of super keyword ?

Ans. The super class is used to access immediate parent class from the subclass. It is used to

1. access parent's variable,
2. access parent's method
3. access parent's constructor invocation

Q.6 Differentiate between inheritance and polymorphism.

Sr. No.	Inheritance	Polymorphism
1.	Inheritance is a property in which some of the properties and methods of base class can be derived by the derived class.	Polymorphism is ability for an object to used different forms. The name of the function remains the same but it can perform different tasks.
2.	Various types of inheritance can be single inheritance, multiple inheritance, multilevel inheritance and hybrid inheritance.	Various types of polymorphism are compile time polymorphism and run time polymorphism. In compile time polymorphism there are two types of overloading possible. - Functional overloading and operator overloading. In run time polymorphism there is a use of virtual function.

Q.7 Distinguish between static and dynamic binding.

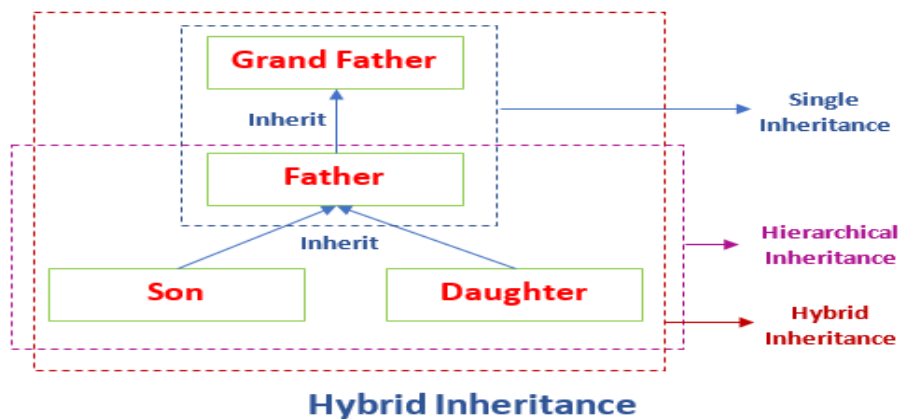
Ans. Static binding is a type of binding in which the function call is resolved at compile time. The dynamic binding is a type of binding in which the function call is resolved at run time. The static binding is called the early binding and the dynamic binding is called late binding.

Q.8 What is the purpose of 'final' keyword ?

Ans. The keyword 'final' is used to avoid further modification of a variable, method or a class. For instance if a variable is declared as final then it can not be modified further, similarly, if a method is declared as final then the method overriding is avoided.

Q.9 Define inheritance hierarchy.

Ans. The inheritance hierarchy represents the collection of classes that inherit from their base classes and thereby make use of the code present in the base class. The data reusability can be achieved by inheritance hierarchy. For example



Q.10 How to prevent inheritance?

Ans. If we declare particular class as final then no class can be derived from it. For example

```
final class Test
{
.....
.....
.....
}
class Test1 extends Test
{
.....
.....
.....
}
```

The above code will produce an error "cannot inherit from final Test".

Thus the use of keyword final for the class prevents the inheritance.

Q.11 Write a class declarations for the following relationship, assuming that all classes are public: a Bulldog is a kind of dog, and a Dog is a kind of Animal.

Ans;

```
class Animal
{
.....
}
class dog extends Animal
{
.....
}
class Bulldog extends dog
{
.....
}
```

Q.12 What is meant by dynamic method dispatch?

Ans. The dynamic method dispatch is run time polymorphism in which a call to overridden function is resolved at runtime instead of at compile time. Hence is the name. For example

```
class Test1
{
public void method1()
{ }
}
class Test2 extends Test1
{
public void method1()
{ }
}
void main()
{
Test1 obj=new Test2();
obj.method1();
}
```

Q.13 Can an abstract class be final? Why?

Ans. The abstract class cannot be final because once we declared the abstract base class with the keyword final then it can not be inherited.

Q.14 Can an abstract class in Java be instantiated? Give the reason.

Ans. The abstract class can not be instantiated (i.e. we can not create the object of this class using new operator) because the abstract class is very much general and less specific. It does nothing and simply lists out only common features of other classes.

Q.15 Why is multiple inheritance using classes a disadvantage in Java?

Ans. In multiple inheritance, the child class is derived from two or more parent classes. It can lead to ambiguity when two base classes implement a method with the same name.

Q.16 State the condition for method overriding in Java

Ans. Method overriding occurs only when the name of the two methods(method in super class and method in subclass) are same and their type signature is same.

Q.17 What is package?

Ans.: Package represent a collection of classes, methods and interfaces. The name of the package must be written as the first statement in the Java source program. The syntax of specifying the package in the Java program is
package name_of_package

Due to package the systematic arrangement of classes is possible. All standard classes in Java are stored in named packages.

Q.18 Mention the necessity for import statement.

Ans. The import statement is used to refer the classes and methods that are present in particular package. This statement is written at the beginning of any Java program. For example -

```
import java.io.*
```

This statement allows to use the useful functionalities for performing input and output operations.

Q.19 List any four packages in java and highlight their features.

Ans. :

Package	Feature
java.io	This package provide useful functionalities for performing input and output operations through data streams and through file system.
java.net	This package is for providing useful classes and interfaces for networking applications which are used in sockets and URL.
java.lang	This package provides core classes and interfaces used in design of Java language.
java.util	Some commonly used utilities such as random number generation, event model, date, time and some of the system properties are represented using java util package.

Q.20 What is API package?

Ans.: • The API packages are application programming interface packages used by java.

- These packages include important classes and interfaces which are used by the programmer while developing the java applications.
- For example java.io, java.util, java.net and so on are the commonly used API.

Q.21 Explain any three uses of package.

Ans. :

- 1.The classes defined in the packages of other program can be easily reused.
2. Two classes from two different packages can have the same name. By using the package name the particular class can be referred.
3. Packages provide the complete separation between the two phases- design and coding.

Q.22 Explain the term CLASSPATH.

Ans. The packages are nothing but the directories. For locating the specified package the java run time system makes use of current working directory as its starting point. This directory path is called CLASSPATH.

Q.23 What are the ways of importing the java packages? OR Write the syntax for importing packages in a Java source file and give an example.

Ans. The import statement can be written at the beginning of the Java program, using the keyword import. For example -

import java.io.File

or

*import java.io.**

Either a class name can be specified explicitly or a * can be used which indicated that the Java compiler should import the entire package.

Q.24 In Java what is the use of interface?

Ans. In java, the interface is used to specify the behaviour of a group of classes. Using interfaces the concept of multiple inheritance can be achieved.

Q.25 Define Interface and write the syntax of the Interface.

Ans. An interface is a collection of method definitions (without implementations) and constant values. It is a blueprint of a class. It has static constants and abstract methods.

Syntax

```
access_modifier interface name_of_interface
{
return_type method_name1(parameter1,parameter2,...parameterN);
.....
type static final variable_name=value;
.....
}
```

Q.26 What modifiers may be used with an interface declaration ?

Ans. : An interface may be declared as public or abstract.

Q.27 Why the variables in interface static and final ?

Ans. The members of interface are static and final because -

- 1) The reason for being static - The members of interface belong to interface only and not object.
- 2) The reason for being final - Any implementation can change value of fields if they are not defined as final. Then these members would become part of the implementation. An interface is pure specification without any implementation.

Q.28 What is the purpose of nested interface?

Ans. The nested interfaces are used to group related interfaces so that they can be easy to maintain. The nested interface must be referred by the outer interface or class. It can't be accessed directly.

Q.29 What are the properties of nested interface?

Ans;

- 1) Nested interfaces are static by default. You don't have to mark them static explicitly.
- 2) Nested interfaces declared inside class can take any access modifier.
- 3) Nested interface declared inside interface is public implicitly.

Q.30 If a class B is derived from class A and extends the interface myinterface, then how will you write this statement for class B definition?

class B extends class A implements myinterface

```
{  
....//body of class B  
}
```

Q. 31 Brief Inner class in Java with its syntax.

Java inner class or nested class is a class which is declared inside the class or interface.

We use inner classes to logically group classes and interfaces in one place so that it can be more readable and maintainable.

Additionally, it can access all the members of outer class including private data members and methods.

Syntax of Inner class

```
class Java_Outer_class{  
//code  
class Java_Inner_class{  
//code  
}  
}
```

Q.32. What is Method overloading?

- **Method Overloading** is a feature in Java that allows a class to have more than one method having same name, but with different number of parameters or parameters having different types and orders
- **Advantages:**
 - increases the readability of the program.
 - Provides the flexibility to use similar methods

Q. 33. What is Type Promotion?

When a data type of smaller size is promoted to the data type of bigger size than it is called type promotion

byte → short → int → long → double
short → int → long → float → double
int → long → float → double
float → double
long → float → double
char → int → long → float → double

Q.34. What is the difference between abstract class and interface?

Abstract Class	Interface
Multiple inheritance is not possible; the class can inherit only one abstract class	Multiple inheritance is possible; The class can implement more than one interfaces
The methods in abstract class may be abstract method or concrete method	The methods in interfaces are abstract by default
Java abstract class is extended using the keyword extends	Java interface can be implemented by using the keyword implements

Q.35 Difference between class and interface

Class	Interface
The class is denoted by a keyword class	The interface is denoted by a keyword interface
By creating an instance of a class the class members can be accessed	you cannot create an instance of an interface
The members of a class can be constant or final	The members of interfaces are always declared as final

Q.36. What is the difference between Method Overloading and Method Overriding

S.No	Method Overloading	Method Overriding
1	Method Overloading means more than one method shares the same name in the class but having different signature.	Method Overriding means method of base class is re-defined in the derived class having same signature.
2	Method Overloading is to “add” or “extend” more to method’s behavior.	Method Overriding is to “Change” existing behavior of method.
3	It is a compile time polymorphism .	It is a run time polymorphism .

Q. 37. What is the difference between this () and super ()?

In java, this keyword is a reference variable that refers to the current object. Super is a special keyword that directs the compiler to invoke the superclass members

Q. 38 . What is the difference between a static and a non-static inner class?

A non-static inner class may have object instances that are associated with instances of the class's outer class. A static inner class does not have any object instances.

Q.39 .Difference between implementing interface and extending interface?

Implements keyword is used for a class to implement a certain interface , while Extends keyword is used for a subclass to extend from a super class.

Q.40List some properties of Abstract classes ?

- The abstract keyword is used to declare a method as abstract.
- An abstract method consists of a method signature, but no method body.

UNIT 2 - PART – B

1. i. Describe in detail about inheritance. (7)
ii. Write a program for inheriting a class. (6)
- 2.i. Illustrate what is super and subclass in Java. (7)
ii. With an example, illustrate how the objects from sub class are inherited by the super class. (6)
3. Examine how to control top level and member level access for the members of the class. (13)
4. Illustrate with an example how passing objects as parameters to methods and returning objects from methods in Java. (13)
5. Describe in brief about object class and its methods in Java with suitable example. (13)
- 6.i. Discuss the concept of abstract class. (7)
ii. Give a program for abstract class with example. (6)
- 7.i. Explain briefly on final keyword. (7)
ii. Explain the concept of abstract class with an example. (6)
- 8.i. Describe what is meant by interface. (7)
ii. How is interface declared and implemented in Java. Give example. (6)
- 9.i. Differentiate classes with interface with suitable examples. (7)
ii. Express a Java program for extending interfaces. (6)
10. Define inner classes. How to access object state using inner classes? Give an example. (13)
- 11.i. Summarize in detail about inner class with its usefulness. (6)
12. Analyse and write a Java program using ArrayList classes and object for the following operations.
 - i. Push (7)
 - ii. Pop (6)
13. Explain in detail about Package with an Example Program. (13)
14. What is meant by constructor? Describe the types of constructors supported by Java with example?
15. Explain Method Overloading and Method Overriding in Java
16. Define Super Keyword and Explain 3 ways of using Super Keyword

UNIT -2 : PART – C

1. Print the sum, difference and product of two complex numbers by creating a class named 'Complex' with separate methods for each operation whose real and imaginary parts are entered by user
2. Assess and write an inheritance hierarchy for classes Quadrilateral, Trapezoid, Parallelogram, Rectangle and Square. Use Quadrilateral as the superclass of the hierarchy. Specify the instance variable and methods for each class. The private instance variables of Quadrilateral should be the x-y coordinate pairs for the four end points of the quadrilateral. Write a program that instances objects of your classes and outputs each objects area (except Quadrilateral) (15)

3. Consider a class student .Inherit this class in UG Student and PG Student. Also inherit students into local and non-local students. Define five Local UG Students with a constructor assuming all classes have a constructor. (15)
- 4.Express a Java Program to create an abstract class named Shape that contains two integers and an empty method named print Area (). Provide three classes named Rectangle, Triangle and Circle such that each one of the classes extends the class Shape. Each one of the classes contains only the method print Area () that prints the area of the given shape. (15)
- 5) An abstract class has a construtor which prints "This is constructor of abstract class", an abstract method named 'a_method' and a non-abstract method which prints "This is a normal method of abstract class". A class 'SubClass' inherits the abstract class and has a method named 'a_method' which prints "This is abstract method". Now create an object of 'SubClass' and call the abstract method and the non-abstract method. (Analyse the result)
- 6.Create an abstract class 'Animals' with two abstract methods 'cats' and 'dogs'. Now create a class 'Cats' with a method 'cats' which prints "Cats meow" and a class 'Dogs' with a method 'dogs' which prints "Dogs bark", both inheriting the class 'Animals'. Now create an object for each of the subclasses and call their respective methods.
7. We have to calculate the area of a rectangle, a square and a circle. Create an abstract class 'Shape' with three abstract methods namely 'RectangleArea' taking two parameters, 'SquareArea' and 'CircleArea' taking one parameter each. The parameters of 'RectangleArea' are its length and breadth, that of 'SquareArea' is its side and that of 'CircleArea' is its radius. Now create another class 'Area' containing all the three methods 'RectangleArea', 'SquareArea' and 'CircleArea' for printing the area of rectangle, square and circle respectively. Create an object of class 'Area' and call all the three methods
8. Describe the advantages of passing objects as parameters in Java methods. Provide an example of a method that accepts objects as arguments and explains how it works. Write a Java method that accepts two objects of the same class as parameters and returns an object representing the combination or result of the two input objects. Explain the logic behind this method
9. Explain the concept of method overloading in Java. Provide an example of a class with overloaded methods, and explain how the Java compiler determines which method to call based on the arguments. What are the rules and restrictions for method overloading in Java? Discuss any potential pitfalls or ambiguities when overloading methods
10. Explain the concept of static members in Java and provide an example of a static variable and a static method