Assignment-3

**1. Create a class Publication with data members title(String) and price(int).From this class derive two classes Book and CD.Class Book adds pages(int) and CD adds Size(int).Each of these classes should have constructors and display().Write a java program to implement this using super,this and method overriding concepts.**

**import** java.util.\*;

**class** Publication

{

String title;

**int** price;

Publication(String title,**int** price)

{

**this**.title=title;

**this**.price=price;

}

**void** display()

{

System.***out***.println("The title is "+title+" price"+price);

}

}

**class** Book **extends** Publication

{

**int** page;

Book(String title,int price,**int** page)

{

**super**(title,price);

**this**.page=page;

}

**void** display()

{

System.***out***.println("The book title is "+title+" price "+price+" pages "+page);

}

}

**class** CD **extends** Publication

{

**int** size;

CD(String title,int price,**int** size)

{

**super**(title,price);

**this**.size=size;

}

**void** display()

{

System.***out***.println("The book title is "+title+" price "+price+" size "+size);

}

}

**class** Demo

{

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

{

Publication p=**new** Publication("python",700);

p.display();

Book b=**new** Book("java",1000,200);

b.display();

CD c=**new** CD("c language",2500,200);

c.display();

}

}

**Output:**

The title is JAVA price700

The book title is Introduction to java price 1000 pages 200

The book title is Java CD price 2500 size 200

**2.Write a java program to demonstrate method overriding?**

**class** Parent {

**void** show()

{

System.***out***.println("Parent class");

}

}

**class** Child **extends** Parent {

**void** show()

{

System.***out***.println("Child class");

}

}

**class** Inherit {

**public** **static** **void** main(String[] args) {

Parent object1 = **new** Parent();

object1.show();

Parent object2 = **new** Child();

object2.show();

}

}

Output:

Parent class

Child class

**3.Write a java program to create an interface called Shape with CalculateArea().Create three classes namely Square,Circle,Triangle which implements shape?**

**interface** Shape

{

**void** calArea();

}

**class** Circle **implements** Shape

{

**int** r = 4;

**double** pi = 3.14, Area = 0;

**public** **void** calArea()

{

area = pi \* r \* r;

System.***out***.println("Area of circle is:"+Area);

}

}

**class** Square **implements** Shape

{

**int** S=4;

**double** Area;

**public** **void** calArea()

{

Area = s\*s;

System.***out***.println("Area of square is:"+Area);

}

}

**class** Triangle **implements** Shape

{

**int** h = 3, b = 2;

**double** Area;

**public** **void** calArea()

{

area = 0.5\*b\*h;

System.***out***.println("Area of triangle is:"+Area);

}

}

**public** **class** Demo

{

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

{

Shape a;

s=**new** Circle();

s.calArea();

s=**new** Square();

s.calArea();

s=**new** Triangle();

s.calArea();

}

}

Output:

Area of circle is:50.24

Area of square is:16

Area of triangle is:6.0

**4.Create two packages p1 and p2.The package p1 contains class A which contains one display().Create class B in package 2.The main method of class B invoke A’s display .Write a java program to do this?**

**package p1**;

**public** **class** A

{

**public** **void** display()

{

System.***out***.println("Package A invokes");

}

}

**import** p1.\*;

**class** B{

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

{

A a=**new** A();

a.display();

}

}

Output:

C:\programs\packages>javac B.java

C:\programs\packages>java B

Package A invokes

**5.** **Write a java program to count numbers, characters in the command line arguments using Exception handling mechanism**

**class** Count

{

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

{

**try**

{

String p=args[0];

**int** q=0,r=0;

**for**(int a=0;a<p.length();a++)

{

**if**(!(Character.*isSpaceChar*(p.charAt(a))))

{

**if**(Character.*isDigit*(p.charAt(a)))

r+=1;

**else** **if**(Character.*isLetter*(p.charAt(a)))

q+=1;

}

}

System.***out***.println("The character count is "+qq+" digit count "+qr);

}

**catch**(Exception e)

{

System.***out***.println(e);

}

}

}

Output:

C:\programs>java Count

java.lang.ArrayIndexOutOfBoundsException: 0

C:\programs>java Count nihitha19

The character count is 7 digit count 2

**1.What is Inheritance?**

Inheritance is one of the core concepts of object-oriented programming  languages. It is a mechanism where you can to derive a class from another class for a hierarchy of classes that share a set of attributes and methods

**2 .What is Multiple Inheritance?**

“Multiple Inheritance” refers to the concept of one class extending (Or inherits) more than one base class. The problem with “multiple inheritance” is that the derived class will have to manage the dependency on two base classes.In java ,mulple inheritance can be achieved using interfaces.

**3.What is the use of Super keyword?**

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.

To understand the super keyword, you should have a basic understanding of Inheritance.

**4. What is abstract method?**

**Abstract method:** can only be used in an abstract class, and it does not have a body. The body is provided by the subclass (inherited from).**5.What is abstract class?**

**Abstract class:** is a restricted class that cannot be used to create objects (to access it, it must be inherited from another class).

**6.What is the use of final modifier?**

You can declare some or all of a class's **methods final**.

You **use** the **final** keyword in a **method** declaration to indicate that the **method** cannot be overridden by subclasses.

The Object class does this—a number of its **methods** are **final**

**7.** **What is interface? Write the syntax interface**.

 An **interface in Java** is a blueprint of a class. It has static constants and abstract methods.

The interface in Java is a mechanism to achieve abstraction. There can be only abstract methods in the Java interface, not method body. It is used to achieve abstraction and multiple inheritance in Java

**Syntax:**

**interface** <interface\_name>{

    // declare constant fields

    // declare methods that abstract

    // by default.

}

**8. What is package?**

A package is a namespace that organizes a set of related classes and interfaces. Conceptually you can think of packages as being similar to different folders on your computer. You might keep HTML pages in one folder, images in another, and scripts or applications in yet another. Because software written in the Java programming language can be composed of hundreds or *thousands* of individual classes, it makes sense to keep things organized by placing related classes and interfaces into packages

**9. What is exception?**

 An exception is an event, which occurs during the execution of a program, that disrupts the normal flow of the program's instructions.

When an error occurs within a method, the method creates an object and hands it off to the runtime system. The object, called an exception object, contains information about the error, including its type and the state of the program when the error occurred. Creating an exception object and handing it to the runtime system is called throwing an exception.

**10. What is the use of finally block?**

* **Java** finally block is a block that is used to execute important code such as closing connection, stream etc.
* **Java** finally block is always executed whether exception is handled or not.
* **Java** finally block follows try or catch block.

.