**1.program:**

public class Publication

{

public String title;

public int price;

Publication

{

display();

}

display()

{

System.out.println("This is publication class");

}

}

public class Book extends Publication

{

public int pages;

Book()

{

super();

display();

}

add(int pages)

{

this.pages = this.pages + pages;

}

display()

{

System.out.println("This is book class");

}

}

public class CO extends Publication

{

public int size;

CO()

{

super();

display();

}

add(int size)

{

this.size = this.size + size;

}

display()

{

System.out.println("This is CO class");

}

}

**2. program:**

class sum{  
    void display(){  
        int a=10;  
        int b=20;  
        System.out.println("this sum class");  
         
    }  
}  
public class number extends sum  
{  
         
    void display(){  
        System.out.println("number class");  
         
    }  
     
    public static void main(String[] args) {   
    sum s=new sum();  
    s.display();  
    }  
     
 }

**Output**: this sum class

**3.program:**

interface shape{  
   abstract calculateArea();  
     
}  
class square implements shape{  
public void calculateArea(int s){  
    System.out.println(s\*s);    
}  
}  
class circle implements shape{  
public void calculateArea(int r){  
    System.out.println(3.14\*r\*r);    
}  
}  
class triangle implements shape{  
public void calculateArea(int l,int b){  
    System.out.println(0.5\*l\*b);    
}  
}  
public class Program  
{  
    public static void main(String[] args) {  
        circle c=new circle();  
        c.calculateArea(1);  
        square s=new square();  
        s.calculateArea(18);  
        triangle t=new triangle();  
        t.calculateArea(2,20);  
         
    }  
}

**Output:** 3.14

324

20

**4.Program:**

package p1;  
 public class A  
 {  
    public void display(){  
    System.out.print("using package p1");  
         
    }  
}  
package p2;  
import p1.A;  
public class B{  
    public static void main(String args[]){  
        A aobj=new A();  
        aobj.display();  
    }  
}

**Output**: using package p1

**5.Program:**

public class FindNumsAndCharsCount{

public static void main(String[] args) {

try{

char[] ch = args.toCharArray();

int letter = 0;

int num = 0;

for(int i = 0; i < x.length(); i++){

if(Character.isLetter(ch[i])){

letter ++ ;

}

else if(Character.isDigit(ch[i])){

num ++ ;

}

}

System.out.println("The string is : ",args);

System.out.println("The no of characters in the given string is: " + letter);

System.out.println("The no of characters in the given string is: " + num);

}catch(Exception e)

{

Sysytem.out.println(e.printStackTrace());

}

}

}

**Output:**

**input:** The string is : Java412

**Output:** The no of characters in the given string is: 4

The no of characters in the given string is: 3

**1.what is Inheritance?**

Inheritance can be defined as the process where one class acquires the properties of another. With the use of inheritance the information is made manageable in a hierarchical order.

The class which inherits the properties of other is known as subclass (derived class, child class) and the class whose properties are inherited is known as superclass (base class, parent class).

**2.What is Multiple Inheritance?**

 A class can inherit the properties of more than a single parent class. In simpler terms, multiple inheritance means a class extending more than one class.

The programming language of java is unable to utilise this feature directly. It can be achieved indirectly through the usage of interfaces.  A single interface can extend multiple interfaces.

**3.Use of Super Keyword:**

The super keyword is a reference variable which is used to refer immediate parent class object.

Whenever we create the instance of subclass, an instance of parent class is created implicitly which is referred by super reference variable.

super can be used to refer immediate parent class instance variable. It is used to access the data member or field of parent class. It is used if parent class and child class have same fields. It can be used to invoke immediate parent class method. super() can be used to invoke immediate parent class constructor.

**4. What is Abstract Method?**

A method without implementation is known as abstract method. A method must always be declared in an abstract class.

Abstract class doesn’t need to have an abstract method compulsory.  
 If a regular class extends an abstract class, then the class must have to implement all the abstract methods of abstract parent class or it has to be declared abstract as well.

**5. Abstract Class:**

A class which is declared as abstract is known as an abstract class. It can have abstract and non-abstract methods. It needs to be extended and its method implemented. It cannot be instantiated.

\*An abstract class must be declared with an abstract keyword.

\*It can have abstract and non-abstract methods.

\*It cannot be instantiated.

\*It can have constructors and static methods also.

\*It can have final methods which will force the subclass not to

change the body of the method.

**6. Use of Final Modifier:**

The final is a modifier which can be applied to a variable, a method or a class. When a final modifier is used with a class then the class cannot be extended further.

The final keyword is used with a method that it cannot be overridden, which means you cannot override the logic of the method in the subclass. This is also done to protect the original logic of method. Final keyword is used with a variable then its value cannot be changed once assigned.

**7. What is Interface? Write the Syntax of Interface?**

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

Syntax:

Interface <interface\_name>{

//declare constant fields

  //declare methods that abstract

//by default

}

**8. What is Package?**  
  
 Package is a group of similar types of classes, interfaces and sub-packages. It can be categorized in two form, built-in package and user-defined package. Package keyword is used to create a package.

Advantages of Package:

1) package is used to categorize the classes and interfaces so that they can be easily maintained.

2) package provides access protection.

3) package removes naming collision.

**9. What is Exception?**

An exception is a problem that arises during the execution of a program. When an exception occurs the normal flow of the program is disrupted and the program terminates abnormally.

Checked exceptions − A checked exception is an exception that is checked by the compiler at compilation-time, these are also called as compile time exceptions. These exceptions cannot simply be ignored, the programmer should handle these exceptions.

Unchecked exceptions − An unchecked exception is an exception that occurs at the time of execution. These are also called as Runtime exceptions. These include programming bugs, such as logic errors or improper use of an API. Runtime exceptions are ignored at the time of compilation.

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

  finally block is useful in exception handling and always used in conjunction with try block. There should at least be one try block that finally block can be associated to.

When a piece of code throws an exception, it stops processing the remaining code in try block and transfers the control to an appropriate catch block if a catch block exists else control gets transferred to finally block.

finally block is resource deallocation. It means all the resources such as Network Connections, Database Connections, which we opened in try block are needed to be closed. so that we won’t lose our resources as opened. So those resources are needed to be closed in finally block.