// There are two ways to achieve abstraction in java

// Abstract class (0 to 100%) and Interface (100%)

**interface** Printable{

**void** print();

}

**class** A6 **implements** Printable{

**public** **void** print(){System.***out***.println("Hello");}

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

A6 obj = **new** A6();

obj.print();

}

}

// There are two ways to achieve abstraction in java

// Abstract class (0 to 100%) and Interface (100%)

//Interface declaration: by first user

**interface** Drawable{

**void** draw();

}

//Implementation: by second user

**class** Rectangle **implements** Drawable{

**public** **void** draw(){System.***out***.println("drawing rectangle");}

}

**class** Circle **implements** Drawable{

**public** **void** draw(){System.***out***.println("drawing circle");}

}

//Using interface: by third user

**class** TestInterface1{

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

Drawable d=**new** Circle();//In real scenario, object is provided by method e.g. getDrawable()

d.draw();

}}

// There are two ways to achieve abstraction in java

// Abstract class (0 to 100%) and Interface (100%)

**interface** Bank{

**float** rateOfInterest();

}

**class** SBI **implements** Bank{

**public** **float** rateOfInterest(){**return** 9.15f;}

}

**class** PNB **implements** Bank{

**public** **float** rateOfInterest(){**return** 9.7f;}

}

**class** TestInterface2{

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

Bank b=**new** SBI();

System.***out***.println("ROI: "+b.rateOfInterest());

}}

// There are two ways to achieve abstraction in java

// Abstract class (0 to 100%) and Interface (100%)

// interface\_multiple\_inheritance\_by\_implementing\_multiple\_interfaces

**interface** Printable{

**void** print();

}

**interface** Showable{

**void** show();

}

**class** A7 **implements** Printable,Showable{

**public** **void** print(){System.***out***.println("Hello");}

**public** **void** show(){System.***out***.println("Welcome");}

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

A7 obj = **new** A7();

obj.print();

obj.show();

}

}

// There are two ways to achieve abstraction in java

// Abstract class (0 to 100%) and Interface (100%)

// Inheriting multiple interfaces with same methods has no ambiguity problem

// unlike inheriting multiple classes.

// That is why multiple inheritance is not allowed in Java through classes

**interface** Printable{

**void** print();

}

**interface** Showable{

**void** print();

}

**class** TestInterface3 **implements** Printable, Showable{

**public** **void** print(){System.***out***.println("Hello");}

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

TestInterface3 obj = **new** TestInterface3();

obj.print();

}

}

// There are two ways to achieve abstraction in java

// Abstract class (0 to 100%) and Interface (100%)

// Inheriting multiple interfaces with same methods has no ambiguity problem

// unlike inheriting multiple classes.

// That is why multiple inheritance is not allowed in Java through classes

**interface** Printable{

**void** print();

}

**interface** Showable **extends** Printable{

**void** show();

}

**class** TestInterface4 **implements** Showable{

**public** **void** print(){System.***out***.println("Hello");}

**public** **void** show(){System.***out***.println("Welcome");}

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

TestInterface4 obj = **new** TestInterface4();

obj.print();

obj.show();

}

}

// There are two ways to achieve abstraction in java

// Abstract class (0 to 100%) and Interface (100%)

// Since Java 8, we can have method body in interface.

// But we need to make it default or static method.

**interface** Drawable{

**void** draw();

**default** **void** msg(){System.out.println("default method");}

}

**class** Rectangle **implements** Drawable{

**public** **void** draw(){System.out.println("drawing rectangle");}

}

**class** TestInterfaceDefault{

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

Drawable d=**new** Rectangle();

d.draw();

d.msg();

}}

}

// There are two ways to achieve abstraction in java

// Abstract class (0 to 100%) and Interface (100%)

// Since Java 8, we can have method body in interface.

// But we need to make it default or static method.

**interface** Drawable{

**void** draw();

**static** **int** cube(**int** x){**return** x\*x\*x;}

}

**class** Rectangle **implements** Drawable{

**public** **void** draw(){System.***out***.println("drawing rectangle");}

}

**class** TestInterfaceStatic{

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

Drawable d=**new** Rectangle();

d.draw();

System.***out***.println(Drawable.*cube*(3));

}}

// There are two ways to achieve abstraction in java

// Abstract class (0 to 100%) and Interface (100%)

//Creating interface that has 4 methods

**interface** A{

**void** a();//bydefault, public and abstract

**void** b();

**void** c();

**void** d();

}

//Creating abstract class that provides the implementation of one method of A interface

**abstract** **class** B **implements** A{

**public** **void** c(){System.***out***.println("I am C");}

}

//Creating subclass of abstract class, now we need to provide the implementation of rest of the methods

**class** M **extends** B{

**public** **void** a(){System.***out***.println("I am a");}

**public** **void** b(){System.***out***.println("I am b");}

**public** **void** d(){System.***out***.println("I am d");}

}

//Creating a test class that calls the methods of A interface

**class** Test5{

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

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

a.a();

a.b();

a.c();

a.d();

}}