// create an interface

interface Language {

void getName(String name);

}

// class implements interface

class ProgrammingLanguage implements Language {

// implementation of abstract method

public void getName(String name) {

System.out.println("Programming Language: " + name);

}

}

class Main {

public static void main(String[] args) {

ProgrammingLanguage language = new ProgrammingLanguage();

language.getName("Java");

}

}

interface A {

   void funcA();

}

interface B extends A {

   void funcB();

}

class C implements B {

   public void funcA() {

      System.out.println("This is funcA");

   }

   public void funcB() {

      System.out.println("This is funcB");

   }

}

public class Demo {

   public static void main(String args[]) {

      C obj = new C();

      obj.funcA();

      obj.funcB();

   }

}

// Java code for using this() to

// invoke current class constructor

class Test

{

int a;

int b;

//Default constructor

Test()

{

this(10, 20);

System.out.println("Inside default constructor \n");

}

//Parameterized constructor

Test(int a, int b)

{

this.a = a;

this.b = b;

System.out.println("Inside parameterized constructor");

}

public static void main(String[] args)

{

Test object = new Test();

}

}

//Java code for using 'this' keyword

//to return the current class instance

class Test

{

int a;

int b;

//Default constructor

Test()

{

a = 10;

b = 20;

}

//Method that returns current class instance

Test get()

{

return this;

}

//Displaying value of variables a and b

void display()

{

System.out.println("a = " + a + " b = " + b);

}

public static void main(String[] args)

{

Test object = new Test();

object.get().display();

}

}

class Main {

public static void main(String[] args) {

// create primitive types

int a = 5;

double b = 5.65;

//converts into wrapper objects

Integer aObj = Integer.valueOf(a);

Double bObj = Double.valueOf(b);

if(aObj instanceof Integer) {

System.out.println("An object of Integer is created.");

}

if(bObj instanceof Double) {

System.out.println("An object of Double is created.");

}

}

}

// Java program to illustrate the Concept

// of Autoboxing and Unboxing

// Importing required classes

import java.io.\*;

// Main class

class Main {

// Main driver method

public static void main(String[] args)

{

// Creating an Integer Object

// with custom value say it be 10

Integer i = new Integer(10);

// Unboxing the Object

int i1 = i;

// Print statements

System.out.println("Value of i:" + i);

System.out.println("Value of i1: " + i1);

// Autoboxing of character

Character X = 'a';

// Auto-unboxing of Character

char ch = X;

// Print statements

System.out.println("Value of ch: " + ch);

System.out.println(" Value of X: " + X);

}

}

interface Polygon {

void getArea();

// default method

default void getSides() {

System.out.println("I can get sides of a polygon.");

}

}

// implements the interface

class Rectangle implements Polygon {

public void getArea() {

int length = 6;

int breadth = 5;

int area = length \* breadth;

System.out.println("The area of the rectangle is " + area);

}

// overrides the getSides()

public void getSides() {

System.out.println("I have 4 sides.");

}

}

// implements the interface

class Square implements Polygon {

public void getArea() {

int length = 5;

int area = length \* length;

System.out.println("The area of the square is " + area);

}

}

class Main {

public static void main(String[] args) {

// create an object of Rectangle

Rectangle r1 = new Rectangle();

r1.getArea();

r1.getSides();

// create an object of Square

Square s1 = new Square();

s1.getArea();

s1.getSides();

}

}