•• R10.1 Suppose an int value a is two billion and b is -a. What is the result of a – b? Of b – a? What is the result of Integer.compare(a, b)? Of Integer.compare(b – a)?

Solution:

The result of a - b is **4,000,000,000**. The result of b - a is **-4,000,000,000**. This is because the subtraction operation is performed by adding the opposite of the second operand to the first operand. For example, a - b is equivalent to a + (-b), which is 2,000,000,000 + 2,000,000,000 = 4,000,000,000. Similarly, b - a is equivalent to b + (-a), which is -2,000,000,000 + (-2,000,000,000) = -4,000,000,000.

But integer data type Maximum value is 2147483647. Minimum value is –2147483648.Since a-b greater than max value, a-b gives a negative arbitrary value and b-a is less minimum value, b-a gives a positive arbitrary value.

Integer.compare(a, b):

This method returns the value 0 if a is equal to b, it returns -1 if a is numerically less than b, and returns 1 if a is numerically greater than b. Since a is greater than b, the result will be 1.

Integer.compare(b – a): Integer.compare() method takes two integer value and returns -1, 0, +1. Since b-a is only one value, it will give compile time error.

•• R10.2 Suppose a double value a is 0.6 and b is 0.3. What is the result of (int)(a - b)? Of (int)(b - a)? What is the result of Double.compare(a, b)? Of Double.compare(b - a)?

Solution:

(int)(a-b) = 0;

(int)b-a) = 0;

Double.compare(a, b) = 1;

Double.compare(b-a) = compile time error;

R10.3 Suppose C is a class that implements the interfaces I and J. Which of the following

assignments require a cast?

C c = . . .;

I i = . . .;

J j = . . .;

a. c = i;

b. j = c;

c. i = j;

Solution: A and C

Required casting:

c = (I)i;

I = (J)j;

N.B. We can convert class to interface to class if the class implements interface and there is no need to casting.

• R10.4Suppose C is a class that implements the interfaces I and J, and suppose i is declared as: I i = new C(); Which of the following statements will throw an exception?

a.C c = (C) i;

b.J j = (J) i;

c.i = (I) null;

Solution: A, B, and C

• R10.5 What does this code fragment print? Why is this an example of polymorphism?

Measurable[] data = { new BankAccount(10000), new Country("Belgium", 30510) };

System.out.println(average(data));

Solution:

This code fragment prints 15005.0. This is the average of the measures of the two objects in the data array. The BankAccount object has a measure of 10000, which is its balance. The Country object has a measure of 30510, which is its area in square kilometers. The average method takes an array of Measurable objects as a parameter and returns the average of their measures. This is an example of polymorphism because the Measurable interface defines a common behavior (getMeasure) for different classes (BankAccount and Country) that implement it. The average method can work with any object that implements the Measurable interface, regardless of its actual class. This allows us to write generic code that can handle different types of objects in a uniform way.

• R10.6 Suppose the class Sandwich implements the Edible interface, and you are given the variable declarations

Sandwich sub = new Sandwich();

Rectangle cerealBox = new Rectangle(5, 10, 20, 30);

Edible e = null;

Which of the following assignment statements are legal?

Solution:

a.e = sub;

Legal

b.sub = e;

Illegal

c.sub = (Sandwich) e;

Legal

d.sub = (Sandwich) cerealBox;

Illegal

e.e = cerealBox;

Illegal

f.e = (Edible) cerealBox;

Legal

g.e = (Rectangle) cerealBox;

Legal

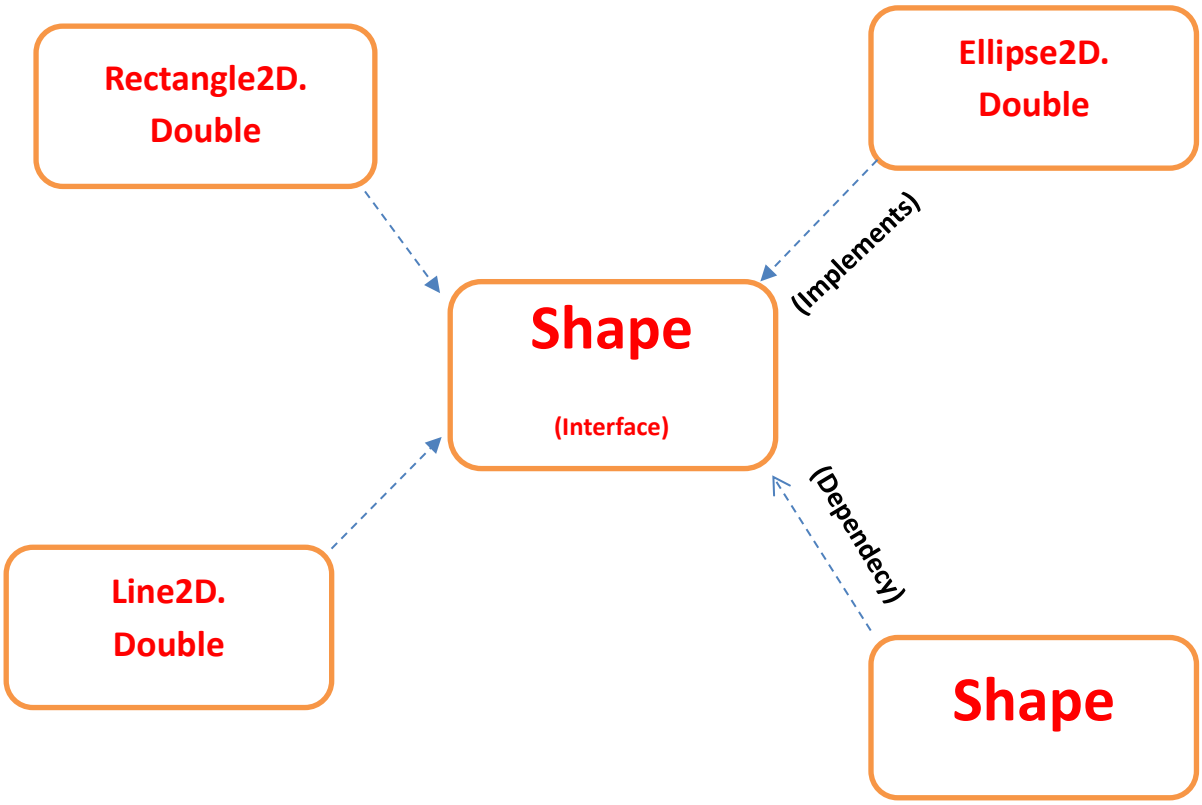
h.e = (Rectangle) null;

Legal

R10.7 The classes Rectangle2D.Double, Ellipse2D.Double, and Line2D.Double implement the Shape

interface. The Graphics2D class depends on the Shape interface but not on the rectangle,

ellipse, and line classes. Draw a UML diagram denoting these facts.



• R10.8 Suppose r contains a reference to a new Rectangle(5, 10, 20, 30). Which of the following assignments is legal? (Look inside the API documentation to check which interfaces the Rectangle class implements.)

a.Rectangle a = r;

b.Shape b = r;

c.String c = r;

d.ActionListener d = r;

e.Measurable e = r;

f.Serializable f = r;

g.Object g = r;

Solution:

A, B, F, and g

• R10.9 Classes such as Rectangle2D.Double, Ellipse2D.Double, and Line2D.Double implement the

Shape interface. The Shape interface has a method

Rectangle getBounds()

that returns a rectangle completely enclosing the shape. Consider the method call:

Shape s = . . .;

Rectangle r = s.getBounds();

Explain why this is an example of polymorphism.

Solution:

Polymorphism refers to the ability of different classes to be treated as instances of the same class through a common interface. In this case:

Classes Implementing the Interface

The classes Rectangle2D.Double, Ellipse2D.Double, and Line2D.Double implement the Shape interface. Each of these classes provides its own implementation of the getBounds() method.

Polymorphic Assignment

The statement Shape s = . . .; declares a variable s of type Shape. This variable can be assigned an instance of any class that implements the Shape interface. It could be an instance of Rectangle2D.Double, Ellipse2D.Double, Line2D.Double, or any other class that implements Shape.

Polymorphic Method Call

The statement Rectangle r = s.getBounds(); calls the getBounds() method on the object referred to by s. Since s is declared as type Shape, this call is polymorphic. At runtime, the actual implementation of getBounds() provided by the specific class that s refers to will be invoked.