

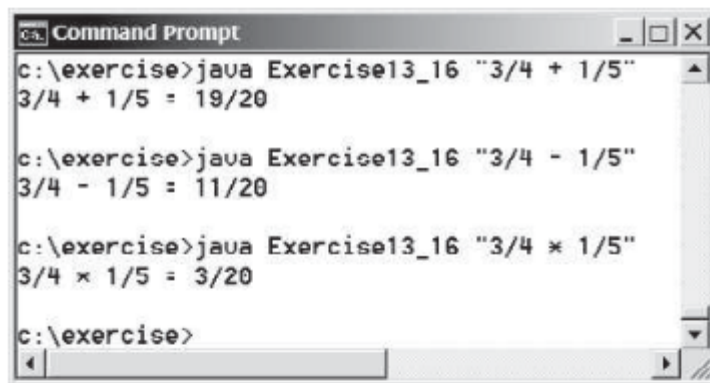
- *13.14** (Demonstrate the benefits of encapsulation) Rewrite the **Rational** class in Listing 13.13 using a new internal representation for the numerator and denominator. Create an array of two integers as follows:

```
private long[] r = new long[2];
```

Use **r[0]** to represent the numerator and **r[1]** to represent the denominator. The signatures of the methods in the **Rational** class are not changed, so a client application that uses the previous **Rational** class can continue to use this new **Rational** class without being recompiled.

- *13.15** (Use **BigInteger** for the **Rational** class) Redesign and implement the **Rational** class in Listing 13.13 using **BigInteger** for the numerator and denominator.

- *13.16** (Create a rational-number calculator) Write a program similar to Listing 7.9, Calculator.java. Instead of using integers, use rationals, as shown in Figure 13.10a. You will need to use the **split** method in the **String** class, introduced in Section 10.10.3, Replacing and Splitting Strings, to retrieve the numerator string and denominator string, and convert strings into integers using the **Integer.parseInt** method.



```

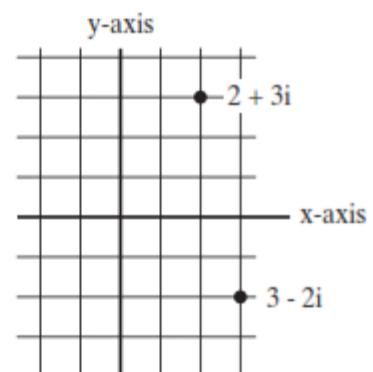
c:\exercise>java Exercise13_16 "3/4 + 1/5"
3/4 + 1/5 = 19/20

c:\exercise>java Exercise13_16 "3/4 - 1/5"
3/4 - 1/5 = 11/20

c:\exercise>java Exercise13_16 "3/4 * 1/5"
3/4 * 1/5 = 3/20

c:\exercise>
  
```

(a)



(b)

FIGURE 13.10 (a) The program takes three arguments (operand1, operator, and operand2) from the command line and displays the expression and the result of the arithmetic operation. (b) A complex number can be interpreted as a point in a plane.

- *13.17** (Math: The **Complex** class) A complex number is a number in the form $a + bi$, where a and b are real numbers and i is $\sqrt{-1}$. The numbers **a** and **b** are known as the real part and imaginary part of the complex number, respectively. You can perform addition, subtraction, multiplication, and division for complex numbers using the following formulas:

$$a + bi + c + di = (a + c) + (b + d)i$$

$$a + bi - (c + di) = (a - c) + (b - d)i$$

$$(a + bi) * (c + di) = (ac - bd) + (bc + ad)i$$

$$(a + bi) / (c + di) = (ac + bd) / (c^2 + d^2) + (bc - ad)i / (c^2 + d^2)$$

You can also obtain the absolute value for a complex number using the following formula:

$$|a + bi| = \sqrt{a^2 + b^2}$$

(A complex number can be interpreted as a point on a plane by identifying the (a, b) values as the coordinates of the point. The absolute value of the complex number corresponds to the distance of the point to the origin, as shown in Figure 13.10b.)

Design a class named **Complex** for representing complex numbers and the methods **add**, **subtract**, **multiply**, **divide**, and **abs** for performing complex-number operations, and override **toString** method for returning a string representation for a complex number. The **toString** method returns **(a + bi)** as a string. If **b** is **0**, it simply returns **a**. Your **Complex** class should also implement the **Cloneable** interface.

Provide three constructors **Complex(a, b)**, **Complex(a)**, and **Complex()**. **Complex()** creates a **Complex** object for number **0** and **Complex(a)** creates a **Complex** object with **0** for **b**. Also provide the **getRealPart()** and **getImaginaryPart()** methods for returning the real and imaginary part of the complex number, respectively.

Write a test program that prompts the user to enter two complex numbers and displays the result of their addition, subtraction, multiplication, division, and absolute value. Here is a sample run:

```
Enter the first complex number: 3.5 5.5 Enter
Enter the second complex number: -3.5 1 Enter
(3.5 + 5.5i) + (-3.5 + 1.0i) = 0.0 + 6.5i
(3.5 + 5.5i) - (-3.5 + 1.0i) = 7.0 + 4.5i
(3.5 + 5.5i) * (-3.5 + 1.0i) = -17.75 + -13.75i
(3.5 + 5.5i) / (-3.5 + 1.0i) = -0.5094 + -1.7i
|(3.5 + 5.5i)| = 6.519202405202649
```

13.19 (*Convert decimals to fractions*) Write a program that prompts the user to enter a decimal number and displays the number in a fraction. Hint: read the decimal number as a string, extract the integer part and fractional part from the string, and use the **BigInteger** implementation of the **Rational** class in Programming Exercise 13.15 to obtain a rational number for the decimal number. Here are some sample runs:

```
Enter a decimal number: 3.25 Enter
The fraction number is 13/4
```

```
Enter a decimal number: -0.45452 Enter
The fraction number is -11363/25000
```