# Exercise 13-3 Parse a name

In this exercise, you'll write an application that parses full names into first and last name or first, middle, and last name, depending on whether the user enters a string consisting of two or three words. The output for the program should look something like this:

```
Welcome to the name parser.

Enter a name: Joel Murach

First name:  Joel
Last name:   Murach
```

1. Open the project named ch13_ex3_NameParser that's in the ex_starts directory. Then, review the code in the NameParserApp class.
2. Add code that separates the name into two or three strings depending on whether the user entered a name with two words or three.
3. Display each word of the name on a separate line. If the user enters fewer than two words or more than three words, display an error message. Also, make sure the application works even if the user enters one or more spaces before or after the name.
4. Test the project to make sure it works correctly.

# Exercise 13-4 Validate a social security number

In this exercise, you'll add a method named getSSN to the Validator class that was presented in chapter 7. Then, you'll use this method in a program to validate a social security number entered by the user.

1. Open the project named ch13_ex4_SSNValidator that's in the ex_starts directory. Then, open the SSNValidatorApp and Validator classes and review the code.
2. Add a method named getSSN to the Validator class that accepts and validates a social security number. This method should accept a Scanner object and a string that will be displayed to the user as a prompt. After it accepts the social security number, this method should check that the entry contains 11 characters. If it does, the method should validate the entry by checking that the first three characters are numeric digits, the fourth character is a hyphen, the fifth and sixth characters are numeric digits, the seventh character is a hyphen, and the eighth through eleventh characters are numeric digits. To check for a numeric digit, you can use a private method that tests if a character is equal to any of the numbers from 0 through 9. If the user's entry doesn't conform to this format, the method should display an error message and ask the user to enter the number again.
3. Modify the SSNValidatorApp class so it uses the getSSN method. Then, compile and run this class to make sure the validation works correctly.