

*LABORATORY* **6**

*Zip Codes and  
Postal Bar Codes*

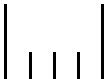
The main purpose of this laboratory is to gain some practice writing methods. These methods are in the context of a class that allows the scanning of a postal bar code to detect a zip code and the generation and display of a postal bar code given a zip code. The preliminary reading and questions are designed to provide you some background on the problem.

# Pre-Laboratory Reading

---

## Introduction to Postal Bar Codes

Bar codes are graphical patterns that represent numbers. They are read by scanners. On packages like cereal boxes, the codes are composed of thick and thin bars and are part of the Universal Product Code system. In this lab we will study programs that process postal bar codes. Postal bar codes use bars that are either full or half height. Each digit requires a five-bar pattern. We will display the codes graphically, but internally we will use the characters | for a full-height bar and : for a half-height bar. For example, the digit 7 is


       or      | : : : |

We will show both the graphical form (although with more spacing than normal) and the form we will use internally.

## The Encoding

Each digit, 0 through 9, has a code. The table below shows the digit and String pairs.

Digit	String
1	":::   "
2	":: : "
3	"::   : "
4	":: :: "
5	":: : : "
6	"::   : : "
7	" ::: "
8	" :: : "
9	" : : : "
0	"   : : : "

## The Rule

Every five-digit zip code, when represented using a postal bar code, has a single full-height bar at the beginning and another at the end, enclosing the pattern for the five digits plus the “check digit.” Thus the code

- begins with a full-height bar,
- is followed by the codes for each of the five digits in the zip code,
- which is followed by the code for the check digit (described below),
- which is followed by a full-height bar.

The pattern goes from left to right.

## The Check Digit

The check digit is calculated by adding up the digits of the five-digit zip code and seeing what would have to be added to the sum to get a multiple of 10. For example, the check digit for the zip code 51220 would be 0 (as  $5 + 1 + 2 + 2 + 0 = 10$ ), while the check digit for 82125 would be 2 (as  $8 + 2 + 1 + 2 + 5 = 18$ ).

## Quick Self-Test

Give the check digit for each of the following zip codes:

- 23173
- 10064
- 00999
- 67023

The answers are 4, 9, 3, and 2, respectively.

### A Basic Problem— Finding the Individual Digits of a Whole Number

When creating a postal bar code for a particular zip code, you must be able to look at each digit in the zip code individually so (1) the digits can be added to determine the check digit and (2) each digit can be replaced by its String code from the table above. The solution is to use the / and % operations. There are several possible approaches, but here are some hints.

Given the number 56734, we can get the right-most, or “tens,” digit using  $56734 \% 10$ . We can get the left-most digit using  $56734 / 10000$ .

How would you get the second digit from the right (the 3 in our example)?

There are two solutions:

- $(56734 \% 100) / 10$
- $(56734 / 10) \% 10$

## Your Tasks

Write out your answers to these three questions PRIOR TO THE LABORATORY:

1. Consider  $x$  to be a five-digit integer. (Note that 2 is viewed as 00002 in this case.) Write Java statements that assign the five digits to D1, D2, D3, D4, and D5, where D1 is the right-most digit, D2 is the second from the right, and so on. We assume that D1 through D5 are declared to be of type int. You will use this information in the lab.
2. Suppose you have the five single digits D1 through D5 as described above, but don't have the original value of  $x$ . What Java statement would assign to  $x$  the value used to create the D1–D5 values?
3. Supposing that you have the five single digits D1 through D5 as described above, how would you compute the check digit's value?

## Substrings

Task 5 of the laboratory will require that you take a string that is a bar code and select particular substrings so you can find the digits of the zip code. For example, if  $b$  is the string holding the bar code, then

```
b.substring(1,6)
```

is the block of characters in  $b$  from position 1 to position 5. Notice that we said 5, not 6! The second parameter in the substring method is the position immediately after the last position that is part of the substring. The first parameter is just the first position for the substring.

## Review Questions

1. What character is first and last in the postal bar code pattern?
2. What is the check digit for 22334?
3. If the zip code is scanned as 00721 and the check digit is scanned as 0, is the pattern likely to be correct or not? Why?

4. Given a three-digit value  $N$ , what arithmetic expression in Java will give
  - a. the right-most digit?
  - b. the left-most digit?
  - c. the middle digit?
5. If  $N$  is an integer, 23 for example, and you want to print it as a three-digit number like 023, how can you create the string "023" from the number 23? Give the answer in terms of  $N$ . (Does your solution work if the integer is 5, when the desired result is "005"?)
6. Give the postal bar code for the zip code 10012.

## The Laboratory

---

### Objectives

- To be able to write simple methods.
  - To recognize that the use of methods can save effort and make code easier to understand.
  - To gain practice in the proper development and testing of your code.
  - To read and modify a larger program.
  - To complete and use methods involving integers and strings.
- 

### A Guide to the Laboratory

You will need to download the two files used in the laboratory from the Web page for this lab. The files are

- `BarCodeCanvas.java`, a class used to graphically display a postal bar code along with some text.
- `BarCodeZipCodeConverter.java`, a class used to "scan" bar codes and produce the zip code or to produce the bar code given a zip code. It contains a simple main method that creates an instance of the class.

You will be creating a project that will contain these two files. You will not change the `BarCodeCanvas.java` file, but the other file has five methods that are incomplete.

Tasks 1 through 3 require that you complete methods and then test them. They can be tested independently of the methods of tasks 4 and 5.

After all the methods are completed and the program is tested, you are to print out a copy of `BarCodeZipCodeConverter.java` to turn in.

---

### A Guide to the Program Structure

The main class is `BarCodeZipCodeConverter` which has a `ViewFrame` with two additional buttons—one for getting the zip code from a bar code string and the other for getting the bar code string from a zip code. A special type of `Canvas` was created that can display a simple text message and a graphical version of the bar code string. It is called `BarCodeCanvas`. There are three inner classes in the `BarCodeZipCodeConverter` class:

- `Scanner`—a class to obtain a bar code string from the user
- `Bar2ZipAction`—an action button that, when pressed, causes the scanner to obtain a bar code string, decodes that string, and produces the zip code



is used to obtain the bar code string and store it into the variable `code`. The guts of the conversion is done by a private method called `convertBarCodeToZipCode`. You will be writing this method in the lab. The last two lines call on methods of the `BarCodeCanvas` object `c` to display the zip code and the bar code.

The `actionPerformed` method for the other button is similar.

```
public void actionPerformed(ActionEvent e)
{
    vf.println("Convert zip code button pressed"); // debugging
    c.clearMessage();
    c.clearCode();
    int zip;
    zip = vf.readInt("Enter 5-digit zip code");
    if (!(zip >= 0 && zip <= 99999))
    {
        vf.showWarningMsg("Not valid zip code");
        return;
    }
    String code;
    code = convertZipCodeToBarCodeString(zip);
    c.setMessage("Zip code: " + zipCodeString(zip));
    c.setCode(code);
}
```

In this case we see an explicit request for the zip code followed by a check that it is legal. Again, the heart of the conversion to a postal bar code string is in the method `convertZipCodeToBarCodeString`. You will be writing this in the lab.

There is another method used in these two examples—`zipCodeString`. It writes the zip code integer as a string with any necessary leading zeroes. The zip code 123 is displayed as 00123. This is already written for you.

#### The Incomplete Methods

The following methods are listed in the order you need to complete them.

- `convertDigitToCode`
- `getCheckDigit`
- `convertZipCodeToBarCodeString`
- `convertCodeToDigit`
- `convertBarCodeToZipCode`

---

#### Tasks 1–3

Complete the first three methods, carefully reading the comments, and then test the conversion of a zip code to a bar code.

---

#### Tasks 4–5 (may be done as a post-lab exercise if time is short)

When the first three methods are working, complete the final two methods and test the conversion of a bar code string to its zip code integer.

---

#### Test Data

You will need to know some zip codes and bar code strings so you can test. You should also include some illegal bar code strings and illegal zip codes in

your testing to be sure they are detected. The following are some sample zip codes and corresponding bar code strings (which include the check digit code).

Zip Code	Bar Code String
23173	" ::: ::: ::: ::: ::: ::: ::: ::: "
00000	"  ::: ::: ::: ::: ::: ::: ::: ::: "
00077	"  ::: ::: ::: ::: ::: ::: ::: ::: "
99999	" : ::: : ::: : ::: : ::: : ::: : ::: : "
22903	" : ::: : ::: : ::: : ::: : ::: : "
50005	" : ::: : ::: : ::: : ::: : ::: : "

## Hints

The first and fourth methods are just lookups in the table at the bottom of the first page of the lab. For example, in `convertDigitToCode`, if the parameter `d` were the digit 7 you would return the string `"|:::|"`. In the method `convertCodeToDigit`, if the string `c` were to equal `"|:::|"`, the method would return a 7. Both methods use an if-else-if structure.

The `getCheckDigit` method requires that you add the individual digits of the number `n` and determine and return the required check digit. **Be careful of the special case where the sum is a multiple of 10 and the check digit would be 0.**

The methods `convertZipCodeToBarCodeString` and `convertBarCodeToZipCode` can use the methods you have already written. This will reduce their complexity greatly.

## Post-Laboratory Exercises

### Exercise 1

Find a piece of bulk mail with a nine-digit zip code such as 23173-0481, and look at the postal bar code. Decode each digit. Can you figure out the coding system? If not, try to find out the system using the Internet or the library.

### Exercise 2

In binary code strings of the symbols '0' and '1', a parity symbol is often added. Even parity is the most common. It works in this way: The number of '1' symbols in a string is counted. If that number is odd, the parity symbol is '1'. Otherwise, it is '0'. The goal is to have the total number of '1' symbols, with the parity symbol included, be an even number.

Write a program `Parity.java` that requests a binary string from the user and displays the string with the parity symbol added to the end.

Write another program called `CheckParity.java` that requests a binary string from the user and, assuming that a parity symbol is part of the string, displays one of the two messages "Parity Error" or "No Parity Error."