

**Directions: SHOW ALL YOUR WORK. REMEMBER THAT PROGRAM SEGMENTS ARE TO BE WRITTEN IN JAVA.**

Notes:

- Assume that the classes listed in the Quick Reference found in the Appendix have been imported where appropriate.
- Unless otherwise noted in the question, assume that parameters in method calls are not `null` and that methods are called only when their preconditions are satisfied.
- In writing solutions for each question, you may use any of the accessible methods that are listed in classes defined in that question. Writing significant amounts of code that can be replaced by a call to one of these methods may not receive full credit.
- The type **long** is not part of the normal AP Test. Mr. Fahrenbacher used it in question 1 because an **int** can't hold most 10 digit phone numbers. Everything you can do with an **int**, you can also do with a **long** (it's just a big **int**).

1. A telephone number is made of three parts: the area code, the local prefix, and the line number. For example, the phone number 847-647-7744 has an area code of 847, a local prefix of 647, and a line number of 7744. The area code and local prefix are always three digits, and the line number is always four digits. An area code may not begin with the number 0, but the local prefix and line number may begin with any number of zeros.

```
public class TelephoneNumber
{
    private long area, prefix, line;

    /** @param full the 10 digit number that represents a phone number
     *      Precondition: full does not begin with 0 and has exactly 10 digits
     */
    public TelephoneNumber(long full)
    { /* to be implemented in part (c) */

        /** @param full the 10 digit number that represents a phone number
         *      Precondition: full does not begin with 0 and has exactly 10 digits
         *      @return the three digit area code that is at the beginning of the full phone number
         */
        public long parseAreaCode(long full)
        { /* to be implemented in part (a) */ }

        /** @param full the 10 digit number that represents a phone number
         *      Precondition: full does not begin with 0 and has exactly 10 digits
         *      @return the three digit local prefix that is in the middle of the full phone number
         */
        public long parsePrefixNumber(long full)
        { /* to be implemented in part (b) */ }

        /** @param full the 10 digit number that represents a phone number
         *      Precondition: full does not begin with 0 and has exactly 10 digits
         *      @return the four digit line number that is at the end of the full phone number
         */
        public long parseLineNumber(long full)
        { /* implementation not shown */ }
    }
}
```

(a) Write method `parseAreaCode`, which takes a positive integer as its parameter. This method returns the area code for the 10 digit phone number. For example, the result of calling `parseAreaCode(8476477744)` should be 847.

Complete method `parseAreaCode` below.

```
/** @param full the 10 digit number that represents a phone number
 *      Precondition: full does not begin with 0 and has exactly 10 digits
 *      @return the three digit area code that is at the beginning of the full phone number
 */
public long parseAreaCode(long full)
```

(b) Write method `parsePrefixNumber`, which takes a positive integer as its parameter. This method returns the local prefix for the 10 digit phone number. For example, the result of calling `parsePrefixNumber(8476477744)` should be 647.

Complete method `parsePrefixNumber` below.

```
/** @param full the 10 digit number that represents a phone number
 *      Precondition: full does not begin with 0 and has exactly 10 digits
 *      @return the three digit local prefix that is in the middle of the full phone number
 */
public long parsePrefixNumber(long full)
```

(c) Write constructor `TelephoneNumber`, which takes a positive integer as its parameter. This constructor sets up the three different parts of the phone number based on the full 10 digit number passed into the method.

In writing `TelephoneNumber`, assume that `parseAreaCode` and `parsePrefixNumber` works as specified, regardless of what you wrote in part (a) or part (b).

Complete constructor `TelephoneNumber` below.

```
/** @param full the 10 digit number that represents a phone number
 *      Precondition: full does not begin with 0 and has exactly 10 digits
 */
public TelephoneNumber(long full)
```

**Extra** (There is is never an extra part on the real AP test): It's important that a phone number does not contain the sequence 911. As soon as any number is dialed that contains that sequence, the emergency contact center for the area is connected to. Finish this method that has just been added to the `Telephone` class that checks if the four digit line number contains the digit sequence 911. For example:

```
Telephone t = new Telephone(8476479112L);
t.lineContains911(); //true

Telephone t2 = new Telephone(8476789121L);
t2.lineContains911(); //false

public boolean lineContains911()
```