Base Conversion

One algorithm for converting a base 10 number to another base \( b \) involves repeatedly dividing by \( b \). Each time a division is performed the remainder and quotient are saved. At each step, the dividend is the quotient from the preceding step; the divisor is always \( b \). The algorithm stops when the quotient is 0. The number in the new base is the sequence of remainders in reverse order (the last one computed goes first; the first one goes last).

In this exercise you will use this algorithm to write a program that converts a base 10 number to a 4-digit number in another base (you don't know enough programming yet to be able to convert any size number). The base 10 number and the new base (between 2 and 9) will be input to the program. The start of the program is in the file `BaseConvert.java`. Save this file to your directory, then modify it one step at a time as follows:

1. The program will only work correctly for base 10 numbers that fit in 4 digits in the new base. We know that in base 2 the maximum unsigned integer that will fit in 4 bits is 11112 which equals 15 in base 10 (or \( 2^4 - 1 \)). In base 8, the maximum number is 77778 which equals 4095 in base 10 (or \( 8^4 - 1 \)). In general, the maximum base 10 number that fits in 4 base \( b \) digits is \( b^4 - 1 \). Add an assignment statement to the program to compute this value for the base that is input and assign it to the variable `maxNumber`. Add a statement that prints out the result (appropriately labeled). Compile and run the program to make sure it is correct so far.

2. Now add the code to do the conversion. The comments below guide you through the calculations—replace them with the appropriate Java statements.

```java
// First compute place0 -- the units place. Remember this comes
// from the first division so it is the remainder when the
// base 10 number is divided by the base (HINT %).
// Then compute the quotient (integer division / will do it!) -
// You can either store the result back in base10Num or declare a
// new variable for the quotient

// Now compute place1 -- this is the remainder when the quotient
// from the preceding step is divided by the base.
// Then compute the new quotient

// Repeat the idea from above to compute place2 and the next quotient

// Repeat again to compute place3
```

3. So far the program does not print out the answer. Recall that the answer is the sequence of remainders written in reverse order—note that this requires concatenating the four digits that have been computed. Since they are each integers, if we just add them the computer will perform arithmetic instead of concatenation. So, we will use a variable of type String. Note near the top of the program a variable named `baseBNum` has been declared as an object of type String and initialized to an empty string. Add statements to the program to concatenate the digits in the new base to `baseBNum` and then print the answer. Compile and run your program. Test it using the following values: Enter 2 for the base and 13 for the base 10 number—the program should print 1101 as the base 2 value; enter 8 for the base and 1878 for the number—the program should print 3526 for the base 8 value; enter 3 for the base and 50 for the number—the program should print 1212.
import java.util.Scanner;

public class BaseConvert {
    public static void main (String[] args) {
        int base; // the new base
        int base10Num; // the number in base 10
        int maxNumber; // the maximum number that will fit in 4 digits in the new base
        int place0; // digit in the 1's (base^0) place
        int place1; // digit in the base^1 place
        int place2; // digit in the base^2 place
        int place3; // digit in the base^3 place
        String baseBNum = new String (""); // the number in the new base
        Scanner scan = new Scanner(System.in);

        // read in the base 10 number and the base
        System.out.println();
        System.out.println("Base Conversion Program");
        System.out.println();
        System.out.print("Please enter a base (2-9): ");
        base = scan.nextInt();

        // Compute the maximum base 10 number that will fit in 4 digits
        // in the new base and tell the user what range the number they
        // want to convert must be in
        System.out.print("Please enter a base 10 number to convert: ");
        base10Num = scan.nextInt();

        // Do the conversion (see notes in lab)
        // Print the result (see notes in lab)
    }
}