Programming Refresher Workshop

Session 6 Exercises

Learning objectives:

- Breaking a number into its smaller parts
- Constructing a number from its smaller parts
- Understanding the method for converting to base 10

Exercise 16 (ex16): Base Conversion

In our customary base-ten system, we use digits "0" through "9" to construct numbers. We do not have a single-digit numeral for "ten". Instead, we use two digits and write "ten" as "10", which stands for "1 tens and 0 ones".

In general, when we need to count one more than nine, we put "0" at the ones position and add one to the tens position. Subsequently, when the digit in the tens position gets too big – when we need one more than nine tens and nine ones ("99"), we put "0" at the tens and the ones positions, and add one to the ten-times-ten, or hundreds, position. The next position is the ten-times-ten-times-ten, or thousands, position. And so forth. This is the base-ten system.

We need not limit ourselves to using base-ten system to represent numbers. Indeed, in Computing, we are familiar with other systems, eg: base-two system (which is called **binary** system, each digit position contains either one or zero), base-eight system (**octal** system, the digits go from 0 through seven), and base-sixteen system (**hexadecimal** system, the digits go from 0 through 9, followed by the letters A through F).

Following table demonstrates how some base-ten numbers can be represented as binaries, octals, or base-5 numbers.)

Decimal (base 10)	Binary (base 2)	Octal (base 8)	Base 5
0	0	0	0
1	1	1	1
2	10	2	2
3	11	3	3
4	100	4	4
5	101	5	10
6	110	6	11
7	111	7	12
8	1000	10	13
9	1001	11	14
10	1010	12	20
11	1011	13	21
12	1100	14	22
13	1101	15	23

In general, it is possible to use any positive integer greater than one as the base for numbers. When we write a base-ten number, for example 456; the number is expressed in terms of powers of ten like this:

458 = 4 * ten^2 + 5 * ten + 8.

The positions of the digits indicate the power of ten multiplied by that digit. In other systems, the bases are no longer ten, but the idea is the same.

In base 8, for example, we can have a base-8 number 712. This number is equivalent to the number 456 in base ten. To see that, we express 710 in terms of powers of **eight** like this:

712 = 7 * eight^2 + 1 * eight + 2

Now, we perform the usual computation:

458 = 4 * 10^2 + 5 * 10 + 8 = 400 + 50 + 8 = 458.

712 = 7 * 8^2 + 1 * 8 + 2 = 448 + 8 + 2 = 458.

Thus, we have 458 (base 10) = 712 (base 8). Or written in a more stylish manner,

$$458_{10} = 712_8$$

For curiosity sake, the following binary number is also equivalent to 458_{10}.

111001010_2 = 1 * 2^8 + 1 * 2^7 + 1 * 2^6 + 0 * 2^5 +

0 * 2^4 + 1 * 2^3 + 0 * 2^2 + 1 * 2^1 + 0

= 256 + 128 + 64 + 8 + 2

= 458

Question: *What is the equivalent base-5 number?*

Converting to base-ten system

Given two numbers of different bases, how do we know if they are actually the same number? The way to do that is to convert them to the same base, such as base 10, and check if the converted results are the same. That is what we did above for 458_{10} and 712_8 . So, to convert a base-8 number such as 76542_8 to a base-ten number, we do the following:

Digit	7	6	5	4	2
Power	4	3	2	1	0
Multiply	7 * 8^4	6 * 8^3	5 * 8^2	4 * 8^1	2 * 8^0
Sum up	7 * 8^4 + 6 * 8^3 + 5 * 8^2 + 4 * 8^1 + 2 * 8^0				

Computationally, we adopt an iterative process. We iteratively obtain the digits of the number one-by-one, from right to left. In each iteration, we obtain a digit, and multiply it with its corresponding power of the base. This result is then accumulated (or summed up) to generate the final answer. Following is a pseudo-code for doing this, assuming that the base-8 number we are trying to convert is “n”. This pseudo-code makes use of two intermediate variables: “pwr” stores the power for the currently operated digit, and “sum” to sum up the result iteratively. At the end of the computation, “sum” contains the desired base-10 number.

```
/* Pseudo-code for conversion to base-10. */
/* pre-condition: n is non-negative */
pwr = 0 ;
sum = 0 ;
While n > 0 do
        d = n mod 8 ;
        n = n / 10 ;
        sum = sum + d * 8^pwr ;
        pwr = pwr + 1 ;
return sum ;
```

Write a program that reads in a base b (which is a number between 2 and 10) and a base-b number num. Your program will first check if num is indeed a base-b number. If num is not a base-b number, the program will abort with an error message. Otherwise, the program will convert num to a base-10 number, and print out the number.

Sample run 1

```
Enter a base b : 8
Enter a base-b number : 712
The corresponding base-10 number is : 458
```

Sample run 2

```
Enter a base b : 2
Enter a base-b number : 111001010
The corresponding base-10 number is : 458
```

Sample run 3

```
Enter a base b : 5
Enter a base-b number : 13612
Error!! 13612 is not a number of base 5.
```

Testing

What are the pre-conditions required for your conversion program?

What are the cases you should test your program with? Can you list out the cases?