2.6 Numerical Representation

2.6.1 Intro practice

Question 1

The set of digits in base-10 (decimal) number system is $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$. Use this information to help you figure out the following.

a. Write out the set of digits in the octal (base-8) number system. $\{0, 1, 2, 3, 4, 5, 6, 7\}$

b. Write out the set of digits in the binary (base-2) number system. $\{0, 1\}$

The hexadecimal (base-16) number system is

$$\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F\}.$$

Why do we use letters? To keep numbers 10 through 15 as one-character representations.

Specifying base

When we need to specify the base when writing out numbers, write it within parentheses, with a subscript of its base number.

- $(123)_{10} = 123$, base-10
- $(1337)_8 = 1337$, base-8
- $(C47)_{16} =$ C47, base-16
- $(1011)_2 = 1011$, base-2

2.6.2 Digits

For the decimal number $2,368$, we can write this as its individual digits:

Thousands (10^3)	**Hundreds** (10^2)	**Tens** (10^1)	**Ones** (10^0)
2	3	6	8

And then we can build out $2,368$ as the mathematical equation:

$$2 \cdot 10^3 + 3 \cdot 10^2 + 6 \cdot 10^1 + 8 \cdot 10^0$$

Likewise, for the binary number 0101 1001, we can write it as:

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
0	1	0	1	1	0	0	1

And into the equation:

$$1 \cdot 2^6 + 1 \cdot 2^4 + 1 \cdot 2^3 + 1 \cdot 2^0$$

Question 2

Expand each of the following numbers as a mathematical equation. Make sure to pay attention to the *base* value.

a. Write out the equation for $(19)_{10}$

10^1	10^0
1	9

b. Write out the equation for $(0010\ 1101)_2$

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
0	0	1	0	1	1	0	1

c. Write out the equation for $(FFAA66)_{16}$

16^5	16^4	16^3	16^2	16^1	16^0
F	F	A	A	6	6

2.6.3 Converting between bases

Algorithm for converting a decimal number to base b:

1. Input a natural number n
2. While $n > 0$, do the following:
   (a) Divide n by b and get a quotient q and remainder r.
   (b) Write r as the next (right-to-left) digit.
   (c) Replace the value of n with q, and repeat.

Question 3
Convert the following between bases:

a. Convert $(35)_{10}$ to binary (base-2) $n = 35$, $b = 2$
$35/2 = 17 + 1/2$ $(a/b = q + r/b)$ $q = 17$, $r = 1$
$17/2 = 8 + 1/2$ $q = 8$, $r = 1$
$8/2 = 4 + 0/2$ $q = 4$, $r = 0$
$4/2 = 2 + 0/2$ $q = 2$, $r = 0$
$2/2 = 1 + 0/2$ $q = 1$, $r = 0$
$1/2 = 0 + 1/2$ $q = 0$, $r = 1$
n = 0

= 0010 0011

b. Convert $(125)_{10}$ to binary (base-2) $n = 26$, $b = 16$
$125/2 = 62 + 1/2$ $(a/b = q + r/b)$ $q = 62$, $r = 1$
$62/2 = 31 + 0/2$ $q = 31$, $r = 0$
$31/2 = 15 + 1/2$ $q = 15$, $r = 1$
$15/2 = 7 + 1/2$ $q = 7$, $r = 1$
$7/2 = 3 + 1/2$ $q = 3$, $r = 1$
$3/2 = 1 + 1/2$ $q = 1$, $r = 1$
$1/2 = 0 + 1/2$ $q = 0$, $r = 1$
n = 0

= 0111 1101

Hexadecimal to Binary

Often in computers, we write binary strings as hexadecimal to save space and make it easier to read.

Hex	0	1	2	3	4	5	6	7
Binary	0000	0001	0010	0011	0100	0101	0110	0111
Hex	8	9	A	B	C	D	E	F
Binary	1000	1001	1010	1011	1100	1101	1110	1111

Example: Convert 11001 from binary to hexadecimal

1. Write out in chunks of four. Add leading 0's to the left side.
   0001 1001
2. Swap out each "nibble" with hexadecimal
   0001 = 1 1001 = 9

So, $(0001\ 1001)_2 = (19)_{16}$

Example: Convert *DAD* from hexadecimal to binary

1. Convert each digit back to binary.
   D = 1101 A = 1010 D = 1101

So, $(\text{DAD})_{16} = (1101\ 1010\ 1101)_2$

Question 4

Do the following conversions

a. Convert $(1F0B)_{16}$ to binary:

1 = 0001 F = 1111 0 = 0000 B = 1011

= 0001 1111 0000 1011

b. Convert $(0100\ 0110)_2$ to hexadecimal:

0100 = 4 0110 = 6

= 46

2.6.4 Programming a converter

Let's take the algorithm given by the textbook and write a program to do our conversions for us.

Algorithm for converting a decimal number to base b:

1. Input a natural number n
2. While $n > 0$, do the following:
   (a) Divide n by b and get a quotient q and remainder r.
   (b) Write r as the next (right-to-left) digit.
   (c) Replace the value of n with q, and repeat.

Open up a Python IDE (e.g., IDLE, Wing) and start with the following code, which includes a function definition and the main program loop:

```
# Function definition
def ConvertFromDecimal( n, b ):
    print( "" )
    print( "n = " + str( n ) + ", b = " + str( b ) )

    number = ""

    return number

# Program
while( True ):
    n = input( "Enter a base-10 number to convert: " )
    b = input( "Enter a base to convert it to: " )

    result = ConvertFromDecimal( n, b )

    print( "Result: " + result )
```

We are going to update the `ConvertFromDecimal` function to follow the algorithm above.

We need to begin implementing the algorithm from step 2. For the step "While $n > 0$, do the following:", write the Python code:

```
while ( n > 0 ):
```

Note that in Python, the inside of a while loop is specified by indenting all inner code forward one level; Python doesn't use curly braces like C++, Java, or C# does.

```
def ConvertFromDecimal( n, b ):
    # Now we're inside the function...
    print( "" )
    print( "n = " + str( n ) + ", b = " + str( b ) )

    number = ""

    print( "" )
    while ( n > 0 ):
        # Now we're inside the while loop...
```

Next, within the while loop, we need to calculate the quotient q and the remainder r, which we can use with division and modulus. This is step 2-a.

```
q = n / b
r = n % b
```

How does a normal division give us the correct value? Because we are treating n and b as integers (not floats or decimals), so it is **integer division**. In programming, this means it truncates any remainder.

We can print out the results like this:

```
print( str( n ) + "/" + str( b ) + " = "
        + str( q ) + " + " + str( r ) + "/" + str( b ) )
```

Now we add r onto our number string, following step 2-b:

```
number = str( r ) + number
```

And, finally, we replace n with q - step 2-c:

```
n = q
```

At the return of the function, the number is returned.

Full code:

```python
# Function definition
def ConvertFromDecimal( n, b ):
    print( "" )
    print( "n = " + str( n ) + ", b = " + str( b ) )

    number = ""

    print( "" )
    while ( n > 0 ):
        q = n / b
        r = n % b

        print( str( n ) + "/" + str( b ) + " = " + str( q ) + " + " + str( r ) + "/" + str( b ) )

        number = str( r ) + number
        n = q

    return number

# Program
while( True ):
    n = input( "Enter a base-10 number to convert: " )
    b = input( "Enter a base to convert it to: " )

    result = ConvertFromDecimal( n, b )

    print( "" )
    print( "Result: " + result )
    print( "\n" )
```

Example output:

```
Enter a base-10 number to convert: 23
Enter a base to convert it to: 2

n = 23, b = 2

23/2 = 11 + 1/2
11/2 = 5 + 1/2
5/2 = 2 + 1/2
2/2 = 1 + 0/2
1/2 = 0 + 1/2

Result: 10111


Enter a base-10 number to convert: 65
Enter a base to convert it to: 16

n = 65, b = 16

65/16 = 4 + 1/16
4/16 = 0 + 4/16

Result: 41
```