2.7 Numerical Representation extra credit

2.7.1 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:

```
1  # Function definition
2  def ConvertFromDecimal( n, b ):
3      print( "" )
4      print( "n = " + str( n ) + ", b = " + str( b ) )
5
6      number = ""
7
8      return number
9
10 # Program
11 while( True ):
12     n = input( "Enter a base-10 number to convert: " )
13     b = input( "Enter a base to convert it to: " )
14
15     result = ConvertFromDecimal( n, b )
16
17     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.

```
1  def ConvertFromDecimal( n, b ):
2      # Now we're inside the function...
3      print( "" )
4      print( "n = " + str( n ) + ", b = " + str( b ) )
5
6      number = ""
7
8      print( "" )
9      while ( n > 0 ):
10         # 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:

```
1  # Function definition
2  def ConvertFromDecimal( n, b ):
3      print( "" )
4      print( "n = " + str( n ) + ", b = " + str( b ) )
5
6      number = ""
7
8      print( "" )
9      while ( n > 0 ):
10         q = n / b
11         r = n % b
12
13         print( str( n ) + "/" + str( b ) + " = " + str(
       q ) + " " + str( r ) + "/" + str( b ) )
14
15         number = str( r ) + number
16         n = q
17
18     return number
19
20 # Program
21 while( True ):
22     n = input( "Enter a base-10 number to convert: " )
23     b = input( "Enter a base to convert it to: " )
24
25     result = ConvertFromDecimal( n, b )
26
27     print( "" )
28     print( "Result: " + result )
29     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
```