Representations of Numbers

Written by Rachel J. Morris, last updated 10/5/16

Representations of Numbers

This Chapter:

1) Converting between binary, hexadecimal, decimal, and octal.

Decimal Numbers

Definition: Given a positive integer X, the *decimal representation* for X is a string consisting of digits from {0, 1, 2, 3, 4, 5, 6, 7, 8, 9} that looks like $d_n d_{(n-1)} \ldots d_2 d_1 d_0$, where

$$X = \sum_{i=0}^{n} d_i \cdot 10^i$$

$$= d_n \cdot 10^n + d_{(n-1)} + 10^{(n-1)} + \ldots + d_2 \cdot 10^2 + d_1 \cdot 10^1 + d_0 \cdot 10^0$$

Decimal Numbers

Or, think of $= d_n \cdot 10^n + d_{(n-1)} + 10^{(n-1)} + \ldots + d_2 \cdot 10^2 + d_1 \cdot 10^1 + d_0 \cdot 10^0$

As being the different “places” of a digit – 100s place, 10s place, 1s place, etc.

5	3	2	1
Thousands place	Hundreds place	Tens place	Ones place

Is essentially 5x1000 + 3x100 + 2x10 + 1x1

10^3 10^2 10^1 10^0

Binary Numbers

Definition: The base-two (binary) representation of a positive integer X is a string consisting of the digits from {0, 1} that looks like $b_n b_{(n-1)} \ldots b_2 b_1 b_0$ where

$$X = \sum_{i=0}^{n} b_i \cdot 2^i$$

$$= d_b \cdot 2^n + b_{(n-1)} + 2^{(n-1)} + \ldots + b_2 \cdot 2^2 + b_1 \cdot 2^1 + b_0 \cdot 2^0$$

Binary Numbers

So for binary numbers, the break down is:

1	0	1	1
Eights place	Fours place	Twos place	Ones place

And this is 1x8 + 0x4 + 1x2 + 1x1 = 8 + 2 + 1 = 11

2^3 2^2 2^1 2^0

Numbers

Hexadecimal is base-16

The set of numbers is
{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F}

Decimal is base-10

The set of numbers is
{0, 1, 2, 3, 4, 5, 6, 7, 8, 9}

Octal is base-8

The set of numbers is
{0, 1, 2, 3, 4, 5, 6, 7}

Binary is base-2

The set of numbers is
{0, 1}

Algorithm for writing a number in base-two

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

Algorithm for writing a number in base-two

In programming terms...

```
string DecToBinary( int n )
{
	string binary = "";

	while ( n > 0 )
	{
		int q, r;
		q = n / 2;
		r = n % 2;

		binary = to_string( r ) + binary;
		n = q;
	}

	return binary;
}
```

C++

1)Input a natural number *n*.

2)While *n > 0*, do the following:

1)Divide *n* by 2 and get a quotient *q* and remainder *r*.

2)Write *r* as the next (right-to-left) digit.

3)Replace the value of *n* with *q*, and repeat.

Algorithm for writing a number in base-two

Examples...