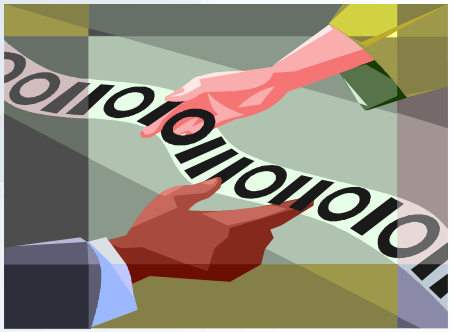
Data Representation



Why Study Data Representation?

- Computers process and store information in binary format
- For many aspects of programming and networking, the details of data representation must be understood
  - **C Programming** – sending information over networks, files
  - **Unix / Linux** – setting permissions for files and directories
  - **Web Pages** – setting color codes

Data Representation

- In terms of this course, we will learn how a simple decimal number (integer) is stored into the computer system as a binary number.
- We will also learn other numbering systems (octal and hexadecimal) that can be used as a “short-cut” to represent binary numbers.

Data Representation

- Before we learn numbering systems, we have to “go-back in time” to see how we learned the decimal numbering system.

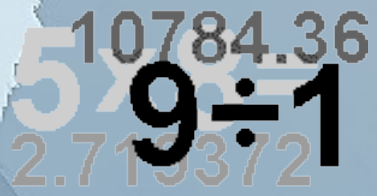
- The decimal numbering system (base 10) uses 10 symbols for each digit (0, 1, 2, … 9). Since most humans have 10 extensions on their hands (2 thumbs, 8 fingers), many suspect that is why humans work with decimal numbers.

Data Representation

Decimal Numbers

- Back in grade school we learn how to understand decimal numbers. For example, take the decimal number 3572. In grade school, we probably learned to break-down this number as follows:

  3 thousands
  5 hundreds
  7 tens
  2 ones



Data Representation

Decimal Numbers

- Another way to look at this number is multiplying the digit by 10 (the numbering base) raised to increasing powers (starting at 0 from the “ones” and moving towards the higher digits)

3 thousands = $3 \times 10^3 = 3 \times 1000$
5 hundreds = $5 \times 10^2 = 5 \times 100$
7 tens = $7 \times 10^1 = 7 \times 10$
2 ones = $2 \times 10^0 = 2 \times 1$

This way of understanding decimal numbers is the basis for math operations such as addition, subtraction, multiplication, decimal numbers, etc!

Data Representation

Binary Numbers

- We can use a similar method to convert a binary number to a decimal number. We do the same thing in the previous slide, but we multiply by base 2 instead of base 10. Take the binary number 1101:

$1 \times 2^3 = 1 \times 8 = 8$
$1 \times 2^2 = 1 \times 4 = 4$
$0 \times 2^1 = 0 \times 2 = 0$
$1 \times 2^0 = 1 \times 1 = 1$

\+ ——

13

Remember, start from the right-hand-side and move to the left.

Therefore, 1101 in binary is 13 in decimal . For programmers, the 8-bit binary number 00001101 can represent the unsigned integer 13!

Data Representation

Octal Numbers

- The octal numbering system (base 8) uses 8 symbols for each digit (0, 1, 2, … 7). We can use the same process in the previous slide to convert an octal number to a decimal number (but use base 8 instead!) . Convert the octal number 2741 to decimal:

$2 \times 8^3 = 2 \times 512 = 1024$
$7 \times 8^2 = 7 \times 64 = 448$
$4 \times 8^1 = 4 \times 8 = 32$
$1 \times 8^0 = 1 \times 1 = 1$
$+$ ———
1505

Remember, start from the right-hand-side and move to the left.

Therefore, 2741 in octal is 1505 in decimal.

Data Representation

Hexadecimal Numbers

- The hexadecimal numbering system (base 16) uses 16 symbols for each digit (0, 1, 2, … 9, A, B, C, D, E, F). Why use letters? Because we are only human and we need to use letters to represent higher digits 10 – 15 as a single digit! Let's convert the hexadecimal number F2A to decimal:

$F \times 16^2 = 15 \times 16^2 = 15 \times 256 = 3840$
$2 \times 16^1 = 2 \times 16^1 = 2 \times 16 = 32$
$A \times 16^0 = 10 \times 16^0 = 10 \times 1 = 10$

\+ ——

3882

Therefore, F2A in Hexadecimal is 3882 in decimal.

Data Representation

- I can understand now how decimal numbers can be stored in the computers as binary numbers, but why are we learning Octal and Hexadecimal numbers?

- As computers and computer programming languages evolved, octal and hexadecimal numbers were considered “short-hand” a short-cut to represent binary numbers.

  For example:
  - Each octal digit represents 3 binary digits.
  - Each hexadecimal digit represents 4 binary digits.

Data Representation

- Linux/Unix operating system commands, networking specialists, programming analysts as well as car-crash investigators use these types of shortcuts which help save space and time issuing a command.

```
chmod 700 secretfile
```

Unix/Linux command to allow file read, write and execute access to the file's owner only!

Hexadecimal numbers can refer to memory addresses which point to incorrect programming procedure!

Cars provide hexadecimal codes to record info prior to impact!

Data Representation

- You will be converting between any number system whether it is from binary to decimal, binary to octal, decimal to binary, octal to hexadecimal, etc.

- The next series of slides provide interesting shortcut how to perform these numbering system conversions. The symbol ^ is used to represent “raised to the power of..”.

  For Example: 10^3 = 10^3

Converting Binary to Octal

Convert the binary number 111110000 to an octal number:

=	1	1	1	1	1	0	0	0	0
x	2^2	2^1	2^0	2^2	2^1	2^0	2^2	2^1	2^0
i.e.	(4)	(2)	(1)	(4)	(2)	(1)	(4)	(2)	(1)
	1x4+	1x2+	1x1	1X4+	1x2+	0x1	0X4+	0x2+	0x1
=	7			6			0		

Remember:

1 octal digit is equal to 3 binary digits. Group binary digits into groups of 3 starting from the right. Add leading zeros if left-most group has less then 3 digits. Convert each group of 3 digits to an octal digit.

Therefore, the binary number 111110000 represents 760 as an octal number. This code can be used to represent directory and file permissions (you will learn how to set permissions soon)

Converting Octal to Binary

Similar to previous calculation, but in reverse:

Convert octal number 760 to binary.

7 6 0

“Spread-out” octal number to make room for binary number result.

(4)(2)(1) (4)(2)(1) (4)(2)(1)

1 1 1 1 1 0 0 0 0

= 111110000

Determine digits (0’s or 1’s) that are required when multiplied by appropriate power of 2 to add up to octal digit.

Converting Binary to Hex

Convert the binary number 111110000 to a hexadecimal number:

=	0	0	0	1		1	1	1	1		0	0	0	0	
	(8)	(4)	(2)	(1)		(8)	(4)	(2)	(1)		(8)	(4)	(2)	(1)	
	1					15					0				
	1					F					0				

Note:

1 hexadecimal digit is equal to 4 binary digits. Group binary digits into groups of 4 starting from the right. Add leading zeros if last group of digits is less than 4 digits. Convert each group of 4 digits to a hexadecimal digit.

Therefore, the binary number 111110000 represents 1F0 as a hexadecimal number.

Converting Hex to Binary

Similar to previous calculation, but in reverse:

Convert hexadecimal number 1F0 to binary.

1	F	0
1	15	0

"Spread-out" hex number to make room for binary number result.

(8)(4)(2)(1) (8)(4)(2)(1) (8)(4)(2)(1)

0 0 0 1 1 1 1 1 0 0 0 0

= 000111110000 = 111110000

Determine digits (0's or 1's) that are required when multiplied by appropriate power of 2 to add up to hexadecimal digit.

Data Representation

Converting decimal to binary

Example: Convert 78 to a binary number

- List the powers of 2 (until greater than or equal to 78) Start with the highest number equal or just less than 78. Put a binary digit “1” below that number and subtract that decimal equivalent from 78 (eg. 78 – 64 = 14). Repeat the same step for the remainder until result is zero. Any numbers NOT used become binary digit “0”

64	32	16	8	4	2	1
1	0	0	1	1	1	0
78-64=14			14-8=6	6-4=2	2-2=0	