Lecture #3
CSC 200-04L Fall 2018

Ambrose Lewis
tjl274@email.vccs.edu

Agenda

- Data Representation
  - Decimal Representations
  - Binary Representations
    - Conversions Decimal to Binary
    - Conversions Binary to Decimals
  - Hexadecimal Representations
  - Character Representations
    - ASCII
    - Unicode
- Using the Command Prompt in Windows

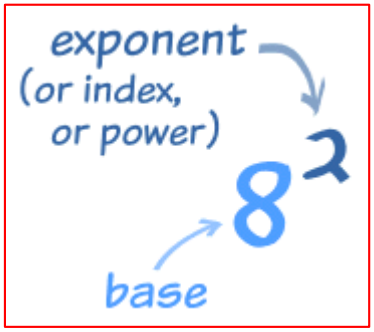
Data Representation…

- Data Representation refers to the format used internally to represent information stored in a computer.
- Computers store many different types of information:
  - numbers
  - Text
  - graphics files, sound files, etc.
- ALL data stored in the computer is stored internally in the same simple format: a sequence of 0's and 1's.
- ***How can a sequence of 0's and 1's represent things as diverse as a photograph, your favorite song, or your resume?***

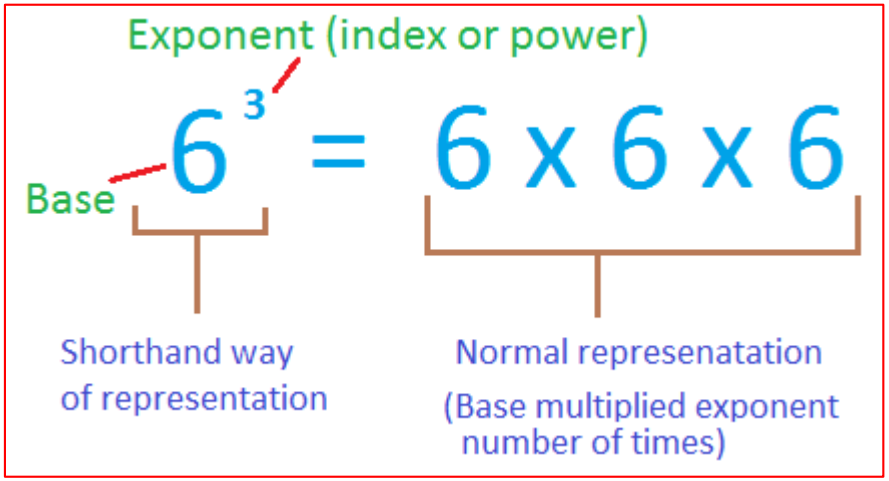
Memory Structure in a Computer

- Memory consists of bits (either a 0 or 1)
- a single bit all by itself can represent two pieces of information (either 0 or 1)
- Remember that 1 byte =8 bits
- a single byte can represent 256 = 2*2*2*2*2*2*2*2 = 2^8 pieces of information or values (0 through 255)
- Word =2,4, or 8 bytes
- a 2 byte word can represent 256^2 pieces of information (approximately 65 thousand).
- Computer memory is byte addressable - each byte has its own address.

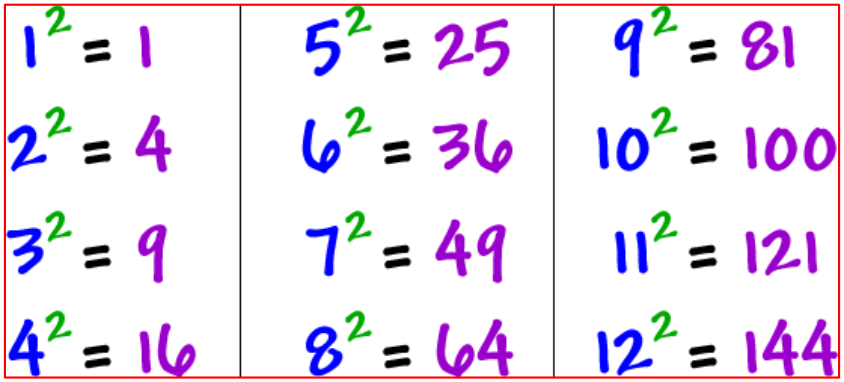
Exponents Refresher…



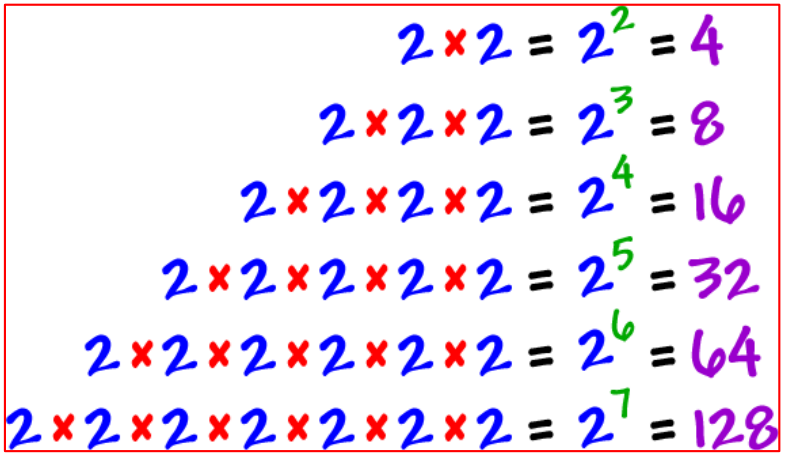
http://www.mathsisfun.com/images/8-squared.gif



http://images.tutorvista.com/cms/images/47/exponents.png



http://coolmath.com/prealgebra/04-exponents/images/02-exponents-definition-01.gif



http://coolmath.com/prealgebra/04-exponents/images/01-exponents-definition-03.gif

If the exponent is 0, then you get 1 (example $9^0 = 1$)

Binary Numbers (1 of 3)

- Normally we write numbers using digits 0 to 9. This is called base 10 or decimal numbers. You can indicate base 10 by putting a 10 subscript after the number (for example 231_{10})
- However, any positive integer (whole number) can be easily represented by a sequence of 0's and 1's.
- Numbers in this form are said to be in base 2 and they are called binary numbers. You can indicate base 2 by putting a two subscript after the number (for example 101_2)
- Base 10 numbers use a positional system based on powers of 10 to indicate their value.
- The rightmost digit is raised to the 10^0 power (ones place)
- The digit just to the left of the ones place is raised to the 10^1 power (tens place)
- The digit just to the left of the tens place is raised to the 10^2 power (hundreds place)
- And so on…

Binary Numbers (2 of 3)

- Decimal (base 10) Example:
  - One hundred twenty three is written 123 in base 10
  - 123 in base 10 is really 1 hundred + 2 tens + 3 ones.
  - 123 in base 10 is really $1*10^2 + 2*10^1 + 3*10^0$
  - 123 in base 10 is really 1*100 + 2*10 + 3*1
- Sometimes you'll see the "^" character used to indicate an exponent.
  - 10^2 = 10^2 = 100
  - This is done in most computer languages, including Java

Binary Numbers (3 of 3)

- Base 2 works the same way, just with powers of two.
- For example, what is the base 10 version of the number 101 in base 2???
- 101 in base 2 = $1*2^2 + 0*2^1 + 1*2^0$
- 101 in base 2 = 1*2*2 + 0*2 + 1*1 = 4 + 0 + 1 = 5 in base 10
- What is the base 10 version of 111 in base 2?

LSB & MSB

- The right-hand most bit of a binary number is defined as the Least Significant Bit or LSB
- The left-hand most bit of a binary number is defined as the Most Significant Bit or MSB
- In the binary number 10, 1 is the MSB and 0 is the LSB
- In a byte (8 bits), we have the following:

Binary Digit							
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
128	64	32	16	8	4	2	1

MSB LSB

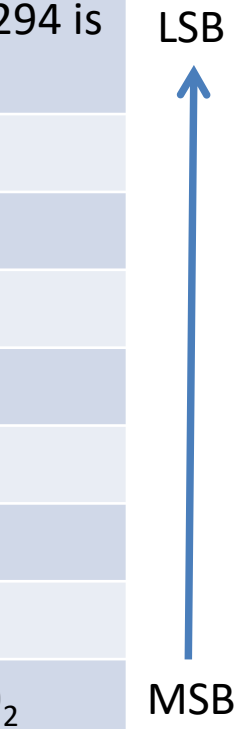
Converting Decimal to Binary

- Keep dividing the decimal number by 2 and write down the remainders
- The remainder will be either 0 or 1
  - If the dividend is even, the remainder is 0
  - If the dividend is odd, the remainder is 1
- Keep dividing until the result is 0
- Example on next page…

Example Converting Decimal to Binary

Goal: Convert 294_{10} to binary

Action	Result	Remainder
Step 1: Divide 294 by 2	147	0 (this is the LSB! Remainder is 0 since 294 is even)
Step 2: Divide 147 by 2	73	1 (since 147 is odd)
Step 3: Divide 73 by 2	36	1
Step 4: Divide 36 by 2	18	0
Step 5: Divide 18 by 2	9	0
Step 6: Divide 9 by 2	4	1
Step 7: Divide 4 by 2	2	0
Step 8: Divide 2 by 2	1	0
Step 9: Divide 1 by 2	0	1 (this is the MSB! Result is 100100110_2



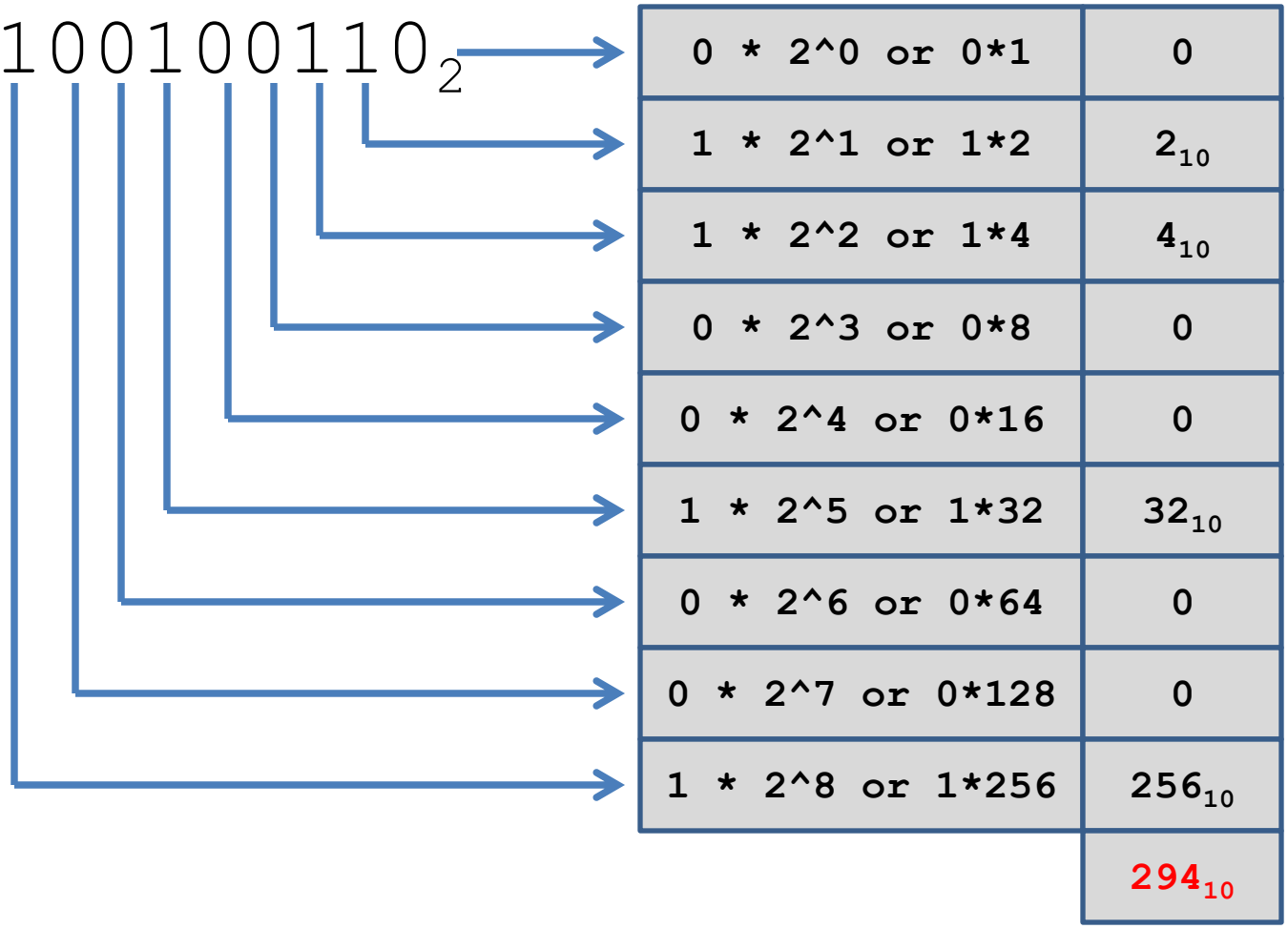
$294_{10} = 100100110_2$

Converting from Binary to Decimal

- The LSB corresponds to 2^0 (o 1s place)
- The place just next to the LSB corresponds to 2^1 (or 2s place)
- Etc…
- So, let's convert 1001001110_2

Converting from Binary to Decimal

- 100100110_2



0 * 2^0 or 0*1	0
1 * 2^1 or 1*2	2_{10}
1 * 2^2 or 1*4	4_{10}
0 * 2^3 or 0*8	0
0 * 2^4 or 0*16	0
1 * 2^5 or 1*32	32_{10}
0 * 2^6 or 0*64	0
0 * 2^7 or 0*128	0
1 * 2^8 or 1*256	256_{10}
	294_{10}

Converting from Binary to Decimal

- Two examples to try:
  - 1101_2 to decimal = _____
  - 6_{10} to binary= _____

Hexadecimal Numbers

- Hexadecimal Numbers are base 16
- Convenient “shorthand” for writing/working with binary numbers in computing
  - We’ll see how you can represent a 4 digit long binary number as a single hexadecimal number
  - Therefore, you can represent an 8 bit byte as a two digit long hexadecimal number

Hexadecimal Numbers

- Hex numbers use the first few letters of the alphabet to represent decimal values 10 through 15 (A = 10, B = 11, …, F=15)
- Single Digit Symbols:

Value	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Binary	0	1														
Decimal	0	1	2	3	4	5	6	7	8	9						
Hexadeci	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

Hexadecimal Numbers

- The rightmost digit in a hexadecimal number corresponds to 16^0
- The place just to the left of the LSB corresponds to 16^1
- The next place to the left of corresponds to 16^2
- Etc…
- Convert to decimal using powers of 16!

$$4_{16} = 4 * 16^0 = 4 * 1 = 4_{10}$$

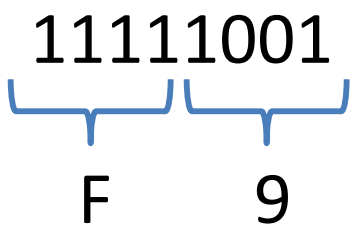
$$40_{16} = 4 * 16^1 = 4 * 16 = 64_{10}$$

$$\begin{aligned} 42_{16} \\ &= 4 * 16^1 + 2 * 16^0 \\ &= 4 * 16 + 2 * 0 \\ &= 64 + 2 \\ &= 66_{10} \end{aligned}$$

Byte Value in Hex

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

- Remember, a byte is 8 bits (or binary digits) long
- The "biggest" value possible to stuff into a byte is $11111111_2 = 255_{10}$
- 4 binary digits (or bits) can be represented as a single digit hex number
  - For example, $1010_2 = 10_{10} = A_{16}$
- You use a two digit hex number as shorthand to represent the value of a byte
- Suppose a byte contained the following bits, what is its value in hex? In decimal?



$F9_{16} = 15*16^1 + 9 * 16^0 = 240 + 9 = 249_{10}$

Text in Computers

- So far, we've dug into how to represent numerical values in the computer
- What about text characters?
  - Symbols that can be typed on the keyboard
  - Warning: There are characters for the digits
    - For example, there is a 1 character
    - Like numbers in a street address
    - You can't perform asthmatic directly on characters

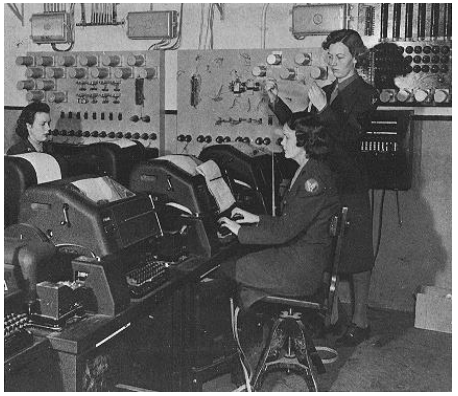
ASCII

- ASCII = American Standard Code for Information Interchange
- Standard for character representation
- First used in 1963
- Think of it as a look up table
- 7 bits long (developed before 8 bit bytes were used everywhere in computing), extended to 8 bits over time
  - How many characters can be represented by 7 bits?

http://en.wikipedia.org/wiki/ASCII

ASCII Characters:

- Blank space
- Digits, 0-9
- Letters, a-z and A-Z
- Basic punctuation symbols like = or , or ;
- Control Characters
  - Originally from teletype machines
  - Does not represent a written symbol
  - Designed to control a printer or display device
  - For example:
    - ASCII 13, CR represents a carriage return
    - ASCII 7, BEL to an audible warning (alert operator of message)



http://en.wikipedia.org/wiki/Teleprinter#mediaviewer/File:WACsOperateTeletype.jpg

http://en.wikipedia.org/wiki/ASCII

ASCII Table

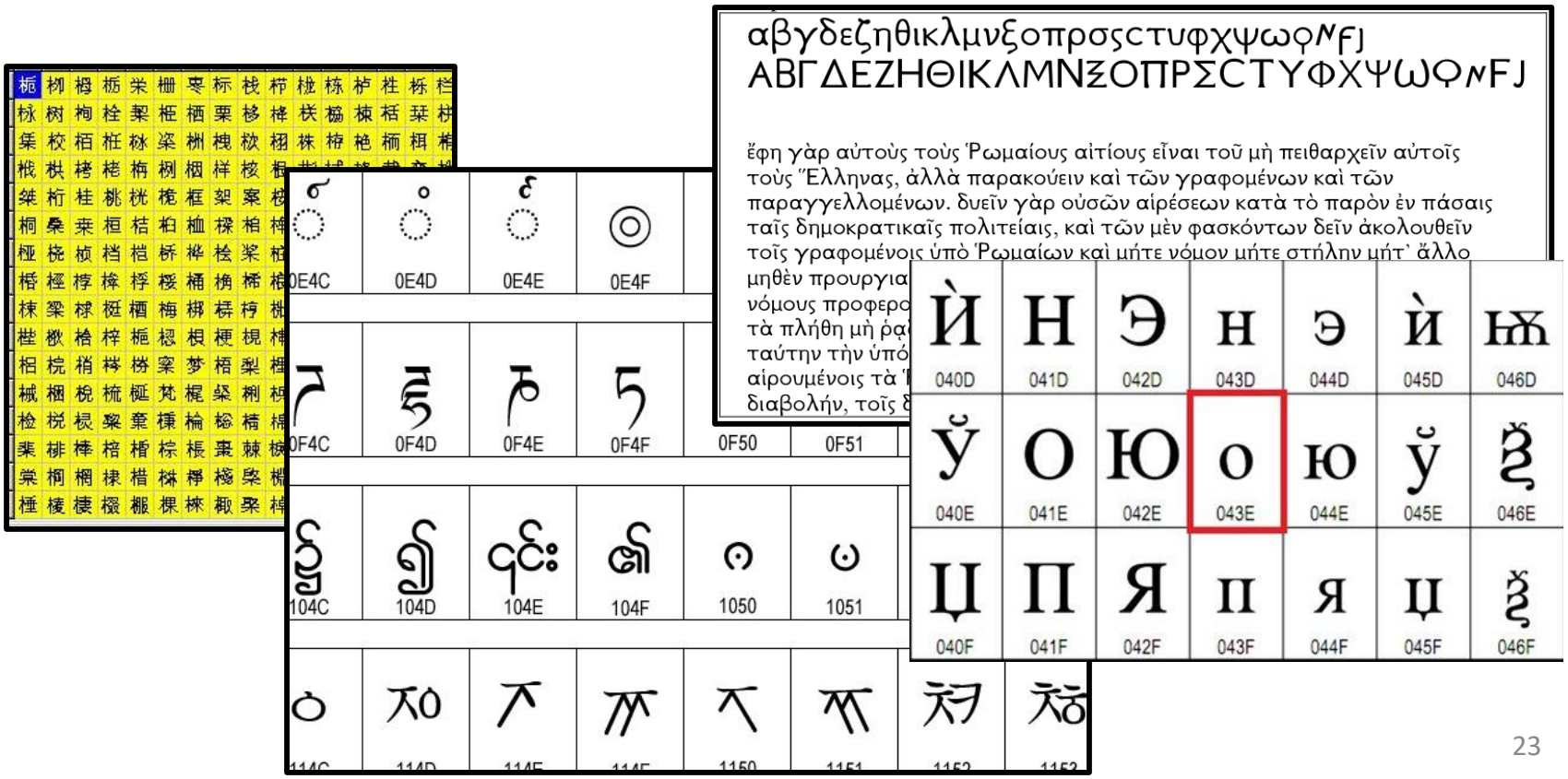
- The number is the ASCII code for the symbol
- For example:
  - The character 2 is ASCII 50, ASCII 66 is B, etc
- Think of the code as an offset into the table (sorting!)

0	*NUL*	16	*DLE*	32	SPC	48	0	64	@	80	P	96	`	112	p
1	*SOH*	17	*DC1*	33	!	49	1	65	A	81	Q	97	a	113	q
2	*STX*	18	*DC2*	34	"	50	2	66	B	82	R	98	b	114	r
3	*ETX*	19	*DC3*	35	#	51	3	67	C	83	S	99	c	115	s
4	*EOT*	20	*DC4*	36	$	52	4	68	D	84	T	100	d	116	t
5	*ENQ*	21	*NAK*	37	%	53	5	69	E	85	U	101	e	117	u
6	*ACK*	22	*SYN*	38	&	54	6	70	F	86	V	102	f	118	v
7	*BEL*	23	*ETB*	39	'	55	7	71	G	87	W	103	g	119	w
8	*BS*	24	*CAN*	40	(	56	8	72	H	88	X	104	h	120	x
9	*HT*	25	*EM*	41	)	57	9	73	I	89	Y	105	i	121	y
10	*LF*	26	*SUB*	42	*	58	:	74	J	90	Z	106	j	122	z
11	*VT*	27	*ESC*	43	+	59	;	75	K	91	[	107	k	123	{
12	*FF*	28	*FS*	44	,	60	<	76	L	92	\	108	l	124	|
13	*CR*	29	*GS*	45	-	61	=	77	M	93	]	109	m	125	}
14	*SO*	30	*RS*	46	.	62	>	78	N	94	^	110	n	126	~
15	*SI*	31	*US*	47	/	63	?	79	O	95	_	111	o	127	DEL

http://www.gjszlin.cz/ivt/esf/ostatni-sin/images/01_ascii.png

Unicode

- Works great for English, but what about other languages?



Unicode

- ASCII is “too small” to represent all the symbols in all of the World’s languages
  - Unicode came out of desire to have a universal character set
- Supports $10FFFF_{16}$ symbols (or “code points”, think of them as indices into the Unicode table)
  - $10FFFF_{16} = 1{,}114{,}111_{10}$ symbols
- The first 256 code points were made identical to the content of ISO-8859-1 (basically 8-bit ASCII as an international standard)
  - makes it trivial to convert existing western text
  - many essentially identical characters were encoded multiple times at different code points to preserve distinctions used by legacy encodings
- Normally a Unicode code point is referred to by writing "U+" followed by its hexadecimal number.

The Unicode code point for "€" is U+20AC.

http://en.wikipedia.org/wiki/Unicode

Agenda

- Data Representation
  - Decimal Representations
  - Binary Representations
    - Conversions Decimal to Binary
    - Conversions Binary to Decimals
  - Hexadecimal Representations
  - Character Representations
    - ASCII
    - Unicode
- Using the Command Prompt in Windows

Command Prompt Intro

- The Command Prompt program allows you to send command directly to the operating system.
- In Command Prompt, you will use your keyboard. You won't use your mouse at all.
- Command Prompt works at a lower level than the GUI. This means that you will have more control over the machine. The disadvantage is that it is less user-friendly.

Getting Started…

- To launch Command Prompt:
  - Select Start, All Programs, Accessories, Command Prompt
  - In older versions of Windows, you can select run and then type cmd in the box
  - The Command Prompt shows up as a black terminal window.

Some Useful Commands…

- **javac:** To compile a Java program, use the javac command. Your program should compile without any errors or warnings (or if there are warnings be absolutely sure that they do not indicate a flaw in your program).
  - C:\>**javac HelloWorld.java**
- **java:** To run a successfully compiled Java program, use the java command.
  - C:\>**java HelloWorld**
- **more:** Display the contents of a file one screen-ful at a time.
  - C:\>**more HelloWorld.java**
- **exit:** Exit the Command Prompt program and close the terminal window.
  - C:\>**exit**
  - **Or, click on the X in the upper right**

WARNING: javac and java aren't in the path on the lab computers, so you can't run them as shown in the lab! To run from the command prompt we need to provide the absolute path to the program (def coming up!)

From http://www.cs.princeton.edu/courses/archive/spr05/cos126/cmd-prompt.html

What's a Path???

- When you type a command in, Windows looks for the program you are trying to run:
  - First, it looks in the current directory
  - Second, it searches in the directories listed in the PATH Environment Variable
    - Type "set" to see all environment variables
  - The command "where" tells you which program will be run when you execute a command
    - Let's do an example with more.exe and more.bat

Files & Directories

- Windows organizes files into a directory hierarchy
- These commands are equivalent to corresponding commands that you access via the Windows point-and-click interface.
- It is useful to be familiar with both interfaces for managing files.
- **dir:** To view the contents of a directory, type dir. This command will list all the files and directories within the current directory. It is analogous to clicking on a Windows folder to see what's inside.

```
C:\> dir
 Volume in drive C has no label.
 Volume Serial Number is C8C7-BDCD

 Directory of C:\

10/26/2004  01:36 PM         0 AUTOEXEC.BAT
10/26/2004  01:36 PM         0 CONFIG.SYS
02/10/2005  01:36 PM       126 HelloWorld.java
12/09/2004  12:11 AM    DIR    Documents and Settings
02/10/2005  08:59 PM    DIR    introcs
11/02/2004  08:31 PM    DIR    j2sdk1.4.2_06
12/29/2004  07:15 PM    DIR    Program Files
01/13/2005  07:33 AM    DIR    WINDOWS
               3 File(s)            126 bytes
               5 Dir(s)  32,551,940,096 bytes free
```

There are 8 items in this directory.
Some of them are files, like "HelloWorld.java"
Others are directories, like "introcs"

more on directories….cd & dir

- **cd:** It is frequently useful to know in which directory you are currently working. In order to find out, type `cd` at the command prompt.

```
C:\> cd
C:\
```

To change directories, use the `cd` command with the name of a directory.

```
C:\> cd introcs
```

Now, the command prompt will be:

```
C:\introcs>
```

To see what is in this directory type:

```
C:\introcs> dir
 Volume in drive C has no label.
 Volume Serial Number is C8C7-BDCD

 Directory of C:\introcs

02/10/2005  08:59 PM    DIR            .
02/10/2005  08:59 PM    DIR            ..
02/03/2005  11:53 PM               126 HelloWorld.java
01/17/2005  01:16 AM               256 readme.txt
               2 File(s)            382 bytes
               2 Dir(s)
```

To return to the previous directory, use the `cd` command, but this time followed by a space and two periods.

```
C:\introcs> cd ..
C:\>
```

more on directories…cd & dir

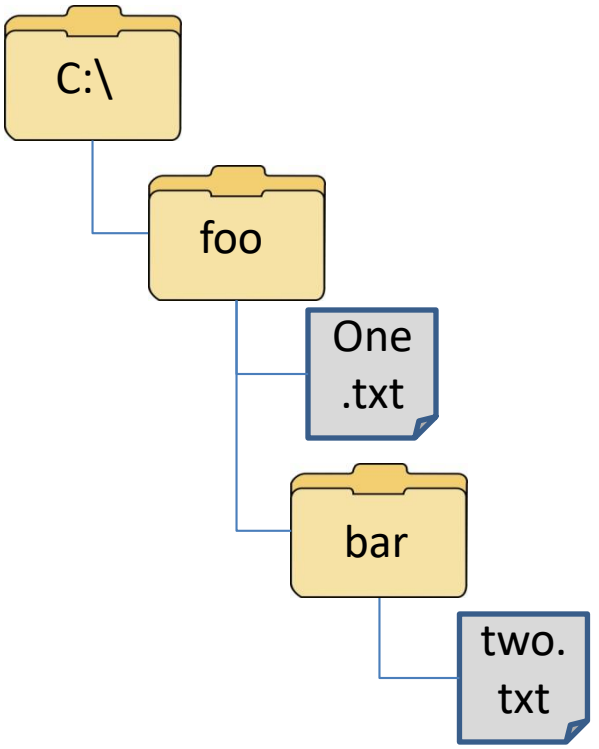
- Some more on cd:
  - Change into parent directory with “cd ..”
  - Change into “grandparent” directory (the parent directory’s parent) with “cd ..\..”
  - Change into a directory named foo with “cd foo”
  - More examples coming…
- Some more on dir:
  - * is a wildcard character, you can use this to filter the files shown by the dir command:
    - To show all java files in the current directory: dir *.java
    - To show all files that start with Q: dir Q*
    - To show all files that have “boo” in their name: dir *boo*
  - You can use .. to search the parent directory with dir:
    - dir ..*.java will list all java files in the parent directory

File Systems

- File systems are arranged in an hierarchy:
  - The root of the hierarchy is a device name in Windows file systems (something like “C:\” or “D:\”)
  - To switch to a different file system, type the drive letter & colon (like d:)
  - Directories can be nested inside the root or other directories
    - For example C:\foo, C:\foo\bar, C:\foo\bar\next, etc.
  - All directories that are not the root have exactly one parent,
    - in the previous example, “C:\” is the parent directory of “foo” , “foo” is the parent of “bar”, “bar” is the parent of “next”, etc
  - A directory can have zero or more directories inside it
    - A directory inside another directory is called a subdirectory (“foo” is a subdirectory of “c:\”, “bar” is a subdirectory of “foo”, etc
    - Each file in the file system resides in exactly one directory
  - see next slide!

File Systems Example

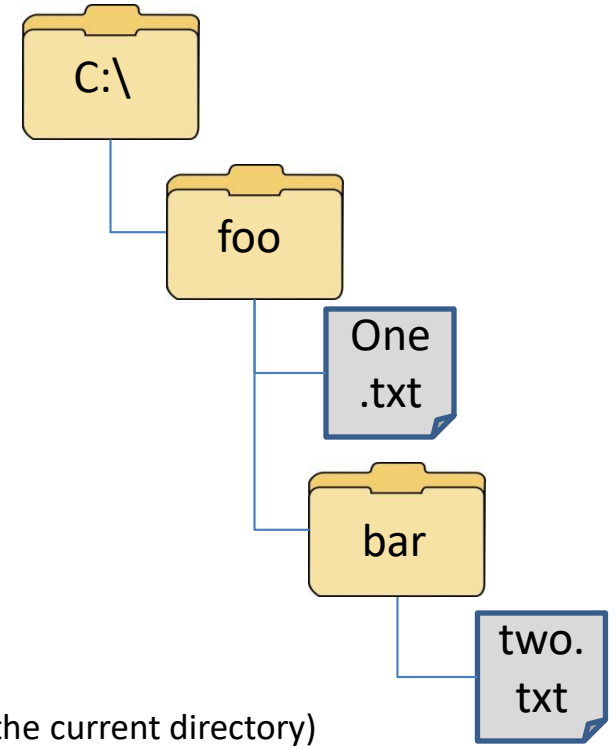
- “C:\” is the root directory
- “C:\” is the parent directory of “foo”
- “foo” is a subdirectory of “C:\”
- The file “one.txt” is in directory “foo”
- “foo” is the parent directory of “bar”
- “bar” is a subdirectory of “foo”
- The file “two.txt” is in directory “bar”



Relative & Absolute Paths

- Know the difference between:
  - Relative path (location of a file or subdirectory based on the directory you are currently in)
    - For example "..\bar\one.txt" refers to a file named "one.txt" in the subdirectory "bar" of the parent directory
      - The ".." part says go up to the parent
      - The "\bar" part says go into the subdirectory named "bar" off of the parent
      - The "one.txt" part refers to the file's name
  - Absolute path (location of a file or subdirectory all the way from the root directory)
    - For example "c:\Users\amb\bar\one.txt"

Relative & Absolute Paths



- The absolute path to “two.txt” is:
  - C:\foo\bar\two.txt
- The absolute path to “bar” is C:\foo\bar

- From “foo”, the relative path to “two.txt” is:
  - bar\two.txt
- From “bar”, the relative path to “one.txt” is:
  - ..\one.txt
- From “bar”, the relative path to “two.txt” is:
  - Two.txt (or, you could say .\two.txt, since “.” is shorthand for the current directory)
- From “bar”, the relative path to the “C:\” is:
  - ..\..

Copy & Move…

- The **Copy** command - This command allows you to copy files from one location to another.
- To use this command you would type
  copy filetocopy copiedfile
- For example if you have the file c:\test\test.txt and would like to copy it to c:\windows\test.txt you would type **copy c:\test\test.txt c:\windows\test.txt** and press enter.
  - If the copy is successful it will tell you so and give you back the prompt.
  - If you are copying within the same directory you do not have to use the path.
- Here are some examples and what they would do:
  - **copy test.txt test.bak** Copies the test.txt file to a new file called test.bak in the same directory
  - **copy test.txt \windows** Copies the test.txt file to the \windows directory
  - **copy * \windows**Copies all the files in the current directory to the \windows directory.
- The **Move** command - This command allows you to move a file from one location to another.
- For example:
  - **move test.txt test.bak** Moves the test.txt file to a new file renaming it to test.bak in the same directory.
  - **move test.txt \windows** Moves the test.txt file to the \windows directory.
  - **move * \windows** Moves all the files in the current directory to the \windows directory.

From http://www.bleepingcomputer.com/tutorials/windows-command-prompt-introduction/

Redirectors

- Redirectors are an important part to using the command prompt as they allow you to manipulate how the output or input of a program is displayed or used.
- Redirectors are used by appending them to the end of a command followed by what you are redirecting to
  - For example: **dir > dir.txt**.
- **>** This redirector will take the output of a program and store it in a file. If the file exists, it will be overwritten. If it does not exist it will create a new file.
  - For example the command **dir > dir.txt** will take the output of the dir command and place it in the dir.txt file. If dir.txt exists, it will overwrite it, otherwise it will create it.
- **>>** This redirector will take the output of a program and store it in a file. If the file exists, the data will be appended to the current data in the file rather than overwriting it. If it does not exist it will create a new file.
  - For example the command **dir >> dir.txt** will take the output of the dir command and appends it to the existing data in the dir.txt file if the file exists. If dir.txt does not exist, it will create the file first.

Some more simple commands…

- To edit a file:
  - Notepad is a basic text editor
  - Type notepad to launch
  - Type notepad filename to open editor and load file
- Sort
  - Sorts the input file alphabetically
- Help
  - get help on commands by typing help followed by a command name
  - Help by itself lists all the commands
  - You can also type command followed by /? To get help (for example “dir /?”
- Date
  - Display & set the date/time
- Type
  - Display file contents
- More
  - Display file a page at a time