CIS-350
INFRASTRUCTURE TECHNOLOGIES

DATA FORMATS – Chapter 4

Read chapter 4.

We will discuss data compression and explain how

- alphanumeric data
- image data
- sound data
- video data

are represented in the computer for

1. internal processing, and
2. storage and transmission purposes

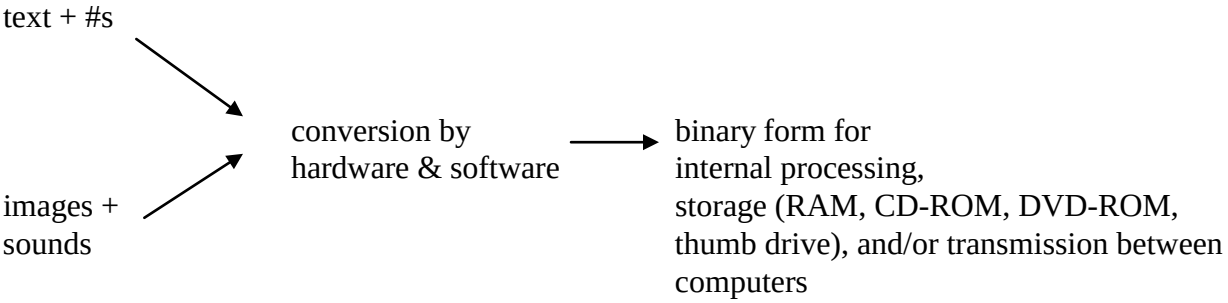
1 and 2 are different.

Alphanumeric (string or character) data (text & numbers)

- are often used in business and data processing
- conversion & processing very straightforward - the keyboard and software generates a binary equivalent for each key you press

Images and sounds

- used in multimedia concepts
- conversion and processing difficult
- may occupy hundreds of MB of space



Formats of data

Proprietary formats

- are used by individual programs such as WordPerfect or Microsoft Word (file name: data1.doc or data1.docx)
- suitable for users working on similar computer systems.

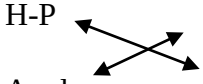
If you type the DOS command:

type data1.docx <E>

it will not possible to read the file as a text file because it has some special characters in it that are understood by Word only.

Standard formats

- make possible for users on different systems (hardware and software) to share and exchange data



- important at the interfaces between different programs or between different computers
- the individual programs/computers can use whatever representation is convenient for the processing

- are arbitrary
- designed for convenience in processing, efficiency in storage and flexibility
- usually the format which is being used often and is widely accepted becomes the standard format

See Fig. 4.2, p. 99

1. Alphanumeric data are represented in EBCDIC (IBM mainframe), ASCII (other computers - IBM/PC, VAX, H-P) or Unicode
2. Image data (bit maps) - GIF (Graphical Image Format) file, TIFF (tagged image file format), PNG (portable network graphics), JPEG
3. Outline graphics and fonts - PostScript notation, TrueType
4. Object images – Postscript, SWF(Macromedia Flash)
5. Sound - WAV sound blaster format, AVI, MP3
6. Video – Quicktime, MPEG-2, or -4, RealVideo, WMV, DivX
7. Page description – pdf (Portable Document Format), HTML, XML

Alphanumeric (string or character) data

- data entered as characters (a-z, A-Z), number digits (1-9), and punctuation (, . ! ? : ;)

Ex.

- names, addresses, product descriptions, etc.
- word processing documents
- keywords, variable names, and formulas that make up a computer program

//pseudocode program segment (Java/C+/C#)

Ex.

```
int a=5;
char b='5';
double c=10.67;
```

- when a program is compiled, the compiler arbitrarily assigns memory locations to variables a, b, and c, and initializes them with the above values
- by specifying the type of values (int, char, double) stored in variables, you tell the computer how many storage locations to reserve and how the value should be stored
  - int: 2 or 4 bytes (depends on the implementation of C++/Java/C#)
  - char - 1 byte (or 2 bytes if Unicode is used)
  - double - 8 bytes

In this case, compiler translates numbers and characters from decimal to binary representation.

The table below shows that the values 5, '5', and 10.67 will be converted from the decimal form to the binary form and how they will be stored internally in memory.

Physical Address (Variable name)	Contents of RAM	Comments
a	00000000 00000101	$(5)_{10}$, occupies 2 bytes
b	00110101 or 11110101	code for '5' in ASCII or EBCDIC, occupies 1 byte
c	10.67	It will occupy 8 bytes. I wrote 10.67 in the decimal form because we do not know exactly how to represent this real # in the binary form. In fact, this number will be converted to its binary equivalent and stored in memory in the binary form. See LN for chapter 3 (the example on conversion of the decimal 0.51 to its binary equivalent and LN for chapter 5 (the section on floating point numbers).

or

```
int a;
char b;
double c;
read a, b, c;      //read from the keyboard
```

Run:
5 5 10.67 <E>

The table below shows that the values 5, '5', and 10.67 will be converted from the decimal form to the binary form and how they will be stored internally in memory.

Physical Address (Variable name)	Contents of RAM	Comments
a	00000000 00000101	$(5)_{10}$, occupies 2 bytes
b	00110101 or 11110101	code for '5' in ASCII or EBCDIC, occupies 1 byte
c	10.67	It will occupy 8 bytes. I wrote 10.67 in the decimal form because we do not know exactly how to represent this real # in the binary form. In fact, this number will be converted to its binary equivalent and stored in memory in the binary form. See LN for chapter 3 (the example on conversion of the decimal 0.51 to its binary equivalent and LN for chapter 5 (the section on floating point numbers).

In this case, software and/or hardware (input device) translates numbers and characters from decimal to binary representation.

To summarize, the conversion routines are (1) built into the compiler or (2) separate software and/or hardware implement them.

Three standards for character data representation

Each character read is converted to its equivalent 1-byte or 2-byte unique code.

- ASCII: IBM/PC, VAX, H-P
- EBCDIC: - IBM mainframe
- Unicode - uses 2 bytes, can represent 65,536 characters

See, Figure 4.3, p. 102 for the ASCII code table.
See, Figure 4.4, p. 103 for the EBCDIC code table.

Example:
In ASCII, the word 'LOPEZ' consumes 5 bytes and is represented by the following bit pattern:

	L	O	P	E	Z
Hexadecimal:	4C	4F	50	45	5A
Binary:	0100 1100	0100 1111	0101 0000	0100 0101	0101 1010
Decimal:	76	79	80	69	90

Alternative sources of alphanumeric input

Optical character recognition (OCR)
- Scan a page of text using an image scanner and convert the image into alphanumeric data form using OCR software

Bar code readers – used at groceries checkout counters

Radio frequency identification (RFID) input – used in RFID tags

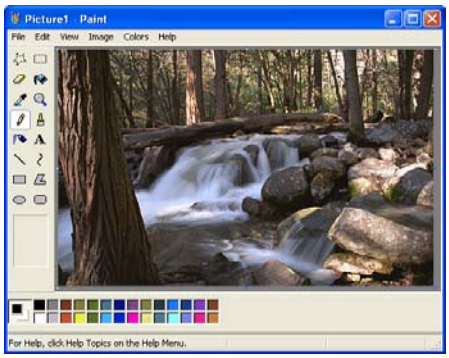
Voice input – the translation process requires the conversion of voice data into sound patterns known as **phonemes**.

Image data - fall into two categories: bit map images and vector images

There are also many useful links on the Internet and WWW to image data. For example,
http://graphicssoft.about.com/od/aboutgraphics/a/bitmapvector.htm

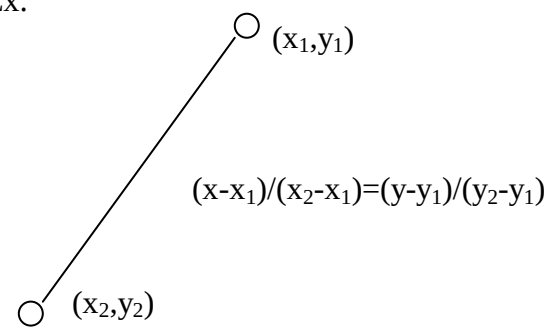
1. bit map images - bmi (raster images)
   - to maintain and reproduce the detail of these images, it is necessary to represent and store each individual point within the image
   - photographs, printings are stored as bmi and read using an image scanner; a video digital camera can be used to capture bmi as well

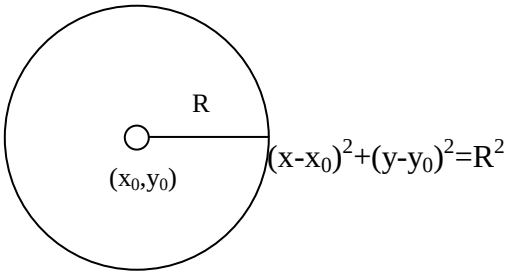
- characterized by continuous variations in shading, color, shape, and texture
- use a great deal of storage, and processing may be time consuming



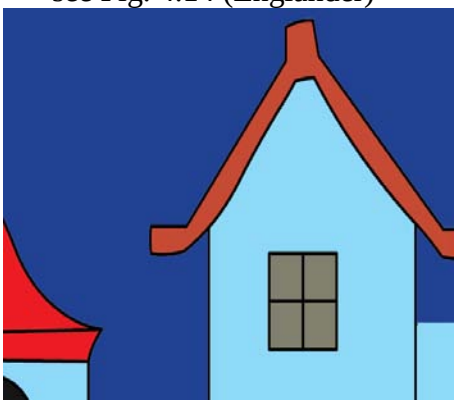
2. <u>object images</u> (vector images or vector graphics)
   - made up of lines and curves that can be <u>defined mathematically</u>

Ex.



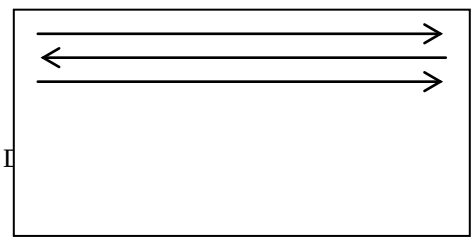


- use less storage than bmi and are easier to manipulate
- usually created by drawing packages
- see Fig. 4.14 (Englander)



Different computer representations and processing techniques are used for each category of the image

<u>Bit map images</u> – bmi



D

A scanner electronically moves over the image (L→R) and then (R→L), converting the image dot by dot, line by line into a stream of binary numbers, each representing a single point in the image, known as pixel. (An image captured from a digital camera is also a bit map image.)

A pixel has the following attributes
- x coordinate
- y coordinate
- level of gray (for B-W images) or color

x-y coordinates do not have to be stored anywhere because a bit map is a 2-dim array

Bit map is a 2-dim array
each element (pixel) of this array may represent a different color

	0	1	2	3	…	…	1021	1022	1023
0	•	•							
1									
2									
…									
…									
…						•			
765									
766									
767	•								•

768 x 1024 pixels array

The row numbers (0 through 767) and column numbers (0 through 1023) represent x and y coordinates, respectively. A "•" represents a pixel.

For B/W images to store 256 different levels of gray, you need 1 byte for each pixel

$(00)_{16}=(00000000)_2$ - white
$(FF)_{16}=(11111111)_2$ - black

Any # in between [0,255] will represent different level of gray.

size of the element (in bytes)	# of unique colors
1	2^8 - 256
2	2^{16} - 64K
3	2^{24} - 16 million - true color system
4	2^{32}

A color pixel is made up of a mixture of intensities of red, blue, and green (RBG)

If each basic color such as red (R) is stored in one byte (8 bits), we have 2^8 - 256 combinations of red. The same refers to blue and green.

$2^8 * 2^8 * 2^8 = 256*256*256 = 2^{24}$ colors

storage requirements for video memory:
768*1024*3 bytes = 2,304KB=2.25MB

The accuracy (faithfulness) of the computer representation of the image depends on the
- size of pixels (# of pixels/inch)
- the # of colors representing each pixel

more pixels - better resolution- greater storage requirements - more processing
Use bmi when:
- a great amount of detail within an image is required
- processing requirements are simple

GIF - Graphics Interchange Format is used as storage format for images. Also, TIFF.

See Fig. 4.13, p. 113 for the GIF file format layout used for transmission.

Sounds

By very nature, sounds are represented by analog (continuous) signals. Sound can be digitized, however, through a sampling process. This is called analog → digital conversion. Samples of tone, pitch, frequency, etc. of the sound are taken thousands of times per second, and then stored/represented as numbers. Later, playing back the sound pulses in the proper order reproduces the original samples, and as long as the time between samples is sufficiently short, a human listener hears continuous sound.

On an audio CD, the sound is sampled 44,100 times/s (Figure 4.17, p. 120). Each sample consumes 2 bytes. One second of sound consumes 2*44,100=88.2KB; One minute: 60*88.2KB=5.3MB. For the WAV and MP3 sound formats, see Figures 4.18 and 4.19 on pp. 121-122. Other sound formats include MIDI, VOC, or MP3.

Video images

Create massive amount of data. A video camera producing full screen 640 x 480 pixel true color images at a frame rate of 30 frames/sec will generate 640 x 480 x 3 x 30 = 27.65MB of data per second. A one-minute clip would consume 60 x 27.65MB = 1.6GB of storage.

Methods to reduce the storage requirements:

(1) reduce the size of the image
(2) limit the number of colors
(3) reduce the frame rate (the image may flicker)
(4) compress the data

Typically the video is stored in a file on a local hard disk or DVD-ROM. It can also be continuously downloaded in real time from a Web or network server. The latter technique is called streaming video and imposes more stringent requirements on the system. The quality of

streaming video is limited by the capability of the network connection and the power of the CPU to decode the data fast enough to keep up with the incoming data stream.

Movie quality video images are typically stored in MPEG-2 format that after compression occupy 30-40MB of data per minute.

Data Compression

It is very impractical to store, transmit, and manipulate the data in its normal uncompressed form.

Two categories of data compression algorithms:

(1) lossless – the compressed data will be restored exactly to its original form
(2) lossy – the compressed data will not be restored exactly to its original form
(some data degradation will occur). Acceptable in multimedia applications.

Examples of lossless data compression algorithms:

(1) Eliminate redundancies in the data

0 5 5 7 3 2 0 0 0 0 1 4 7 3 2 9 1 0 0 0 0 0 6 6 8 2 7 3 2 7 3 2 …

(a) count the number of consecutive 0s and replace the string with the count

0 1 5 5 7 3 2 0 4 1 4 7 3 2 9 1 0 5 6 6 8 2 7 3 2 7 3 2 ….

(b) identify larger sequences within the string

0 1 5 5 7 3 2 0 4 1 4 7 3 2 9 1 0 5 6 6 8 2 7 3 2 7 3 2 ….

and replace each instance of the sequence with the letter, say, “Z”

0 1 5 5 Z 0 4 1 4 Z 9 1 0 5 6 6 8 2 Z Z….

GIF images and zip files are compressed losslessly. JPEG and MPEG-2 are examples of lossy algorithms. The former can reduce the amount of data by a factor 10:1 or more, whereas the latter 100:1 with very little noticeable degradation in image quality. The video/sound/image compression process and restoration process (uncompression) require tremendous amounts of computing power.

Page Description Languages

They describe the layout of objects on a displayed or printed page.

HTML (Hypertext Markup Language) uses tags, which do not have meanings, and describes the appearance of a Web page to the browser. It can be extended by XML in which tags have meanings.

PDF (Portable Document Format) and Postscript offer the ability to recreate sophisticated pages with surprising faithfulness to the intentions of the original page designer.

PDF is a file format and is a subset of Postscript. It is difficult to recreate the original object from PDF. Word or Excel → PDF (easy); PDF → Word or Excel (difficult). Unlike Postscript, PDF does not provide programming language features.