

# Chapter 1

## Introduction to Computers, Programs, and Python



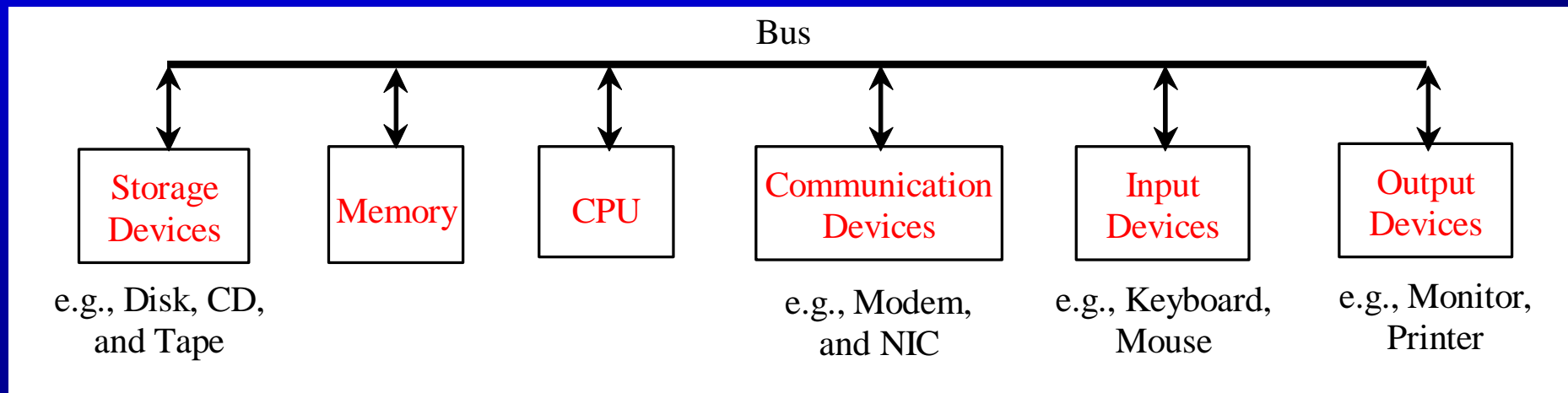
# Objectives

- ❑ To understand computer basics, programs, and operating systems (§§1.2-1.4).
- ❑ To write and run a simple Python program (§1.5).
- ❑ To explain the basic syntax of a Python program (§1.5).
- ❑ To describe the history of Python (§1.6).
- ❑ To explain the importance of, and provide examples of, proper programming style and documentation (§1.7).
- ❑ To explain the differences between syntax errors, runtime errors, and logic errors (§1.8).
- ❑ To create a basic graphics program using Turtle (§1.9).



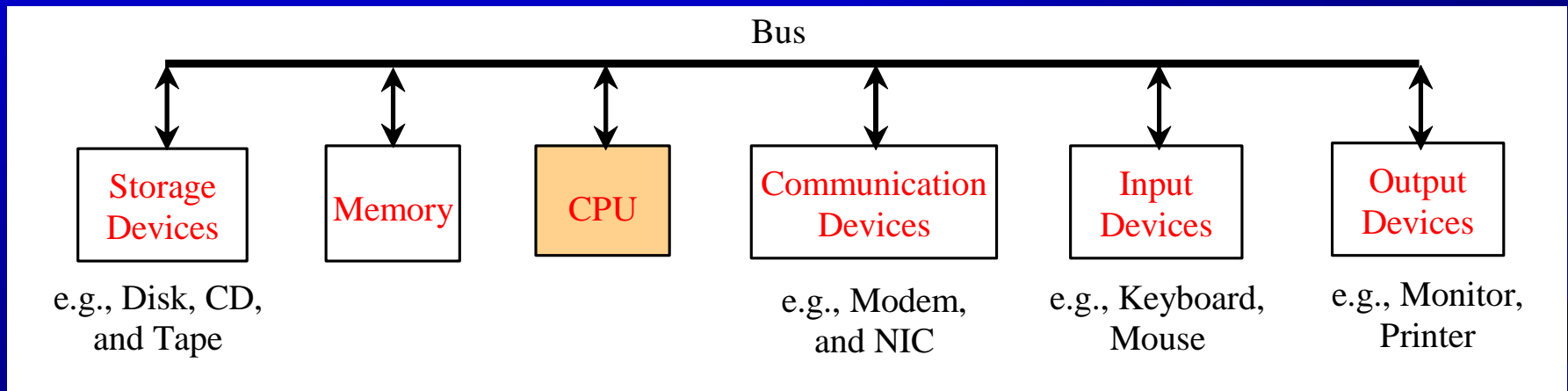
# What is a Computer?

A computer consists of a CPU, memory, hard disk, floppy disk, monitor, printer, and communication devices.



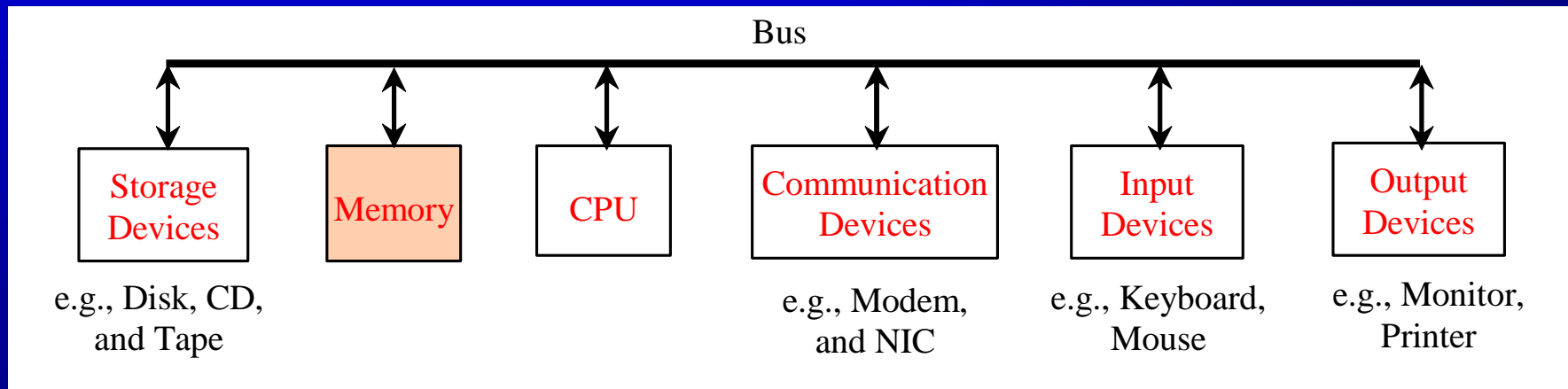
# CPU

The central processing unit (CPU) is the brain of a computer. It retrieves instructions from memory and executes them. The CPU speed is measured in megahertz (MHz), with 1 megahertz equaling 1 million pulses per second. The speed of the CPU has been improved continuously. If you buy a PC now, you can get an Intel Pentium 4 Processor at 3 gigahertz (1 gigahertz is 1000 megahertz).



# Memory

*Memory* is to store data and program instructions for CPU to execute. A memory unit is an ordered sequence of bytes, each holds eight bits. A program and its data must be brought to memory before they can be executed. A memory byte is never empty, but its initial content may be meaningless to your program. The current content of a memory byte is lost whenever new information is placed in it.



# How Data is Stored?

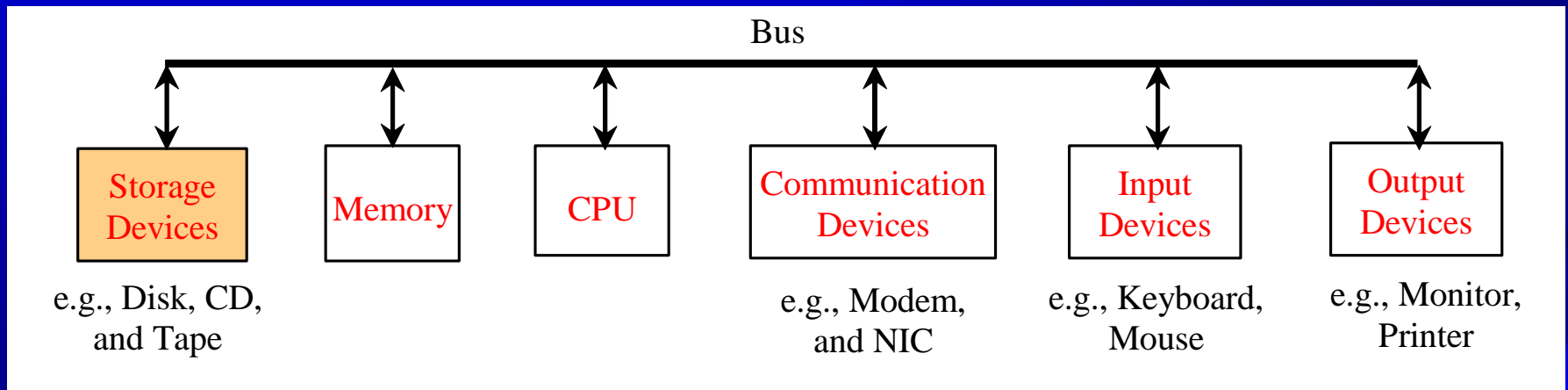
Data of various kinds, such as numbers, characters, and strings, are encoded as a series of bits (zeros and ones). Computers use zeros and ones because digital devices have two stable states, which are referred to as *zero* and *one* by convention. The programmers need not to be concerned about the encoding and decoding of data, which is performed automatically by the system based on the encoding scheme. The encoding scheme varies. For example, character 'J' is represented by 01001010 in one byte. A small number such as three can be stored in a single byte. If computer needs to store a large number that cannot fit into a single byte, it uses a number of adjacent bytes. No two data can share or split a same byte. A byte is the minimum storage unit.

Memory address	Memory content	
.	.	
.	.	
.	.	
2000	01001010	Encoding for character 'J'
2001	01100001	Encoding for character 'a'
2002	01110110	Encoding for character 'v'
2003	01100001	Encoding for character 'a'
2004	00000011	Encoding for number 3



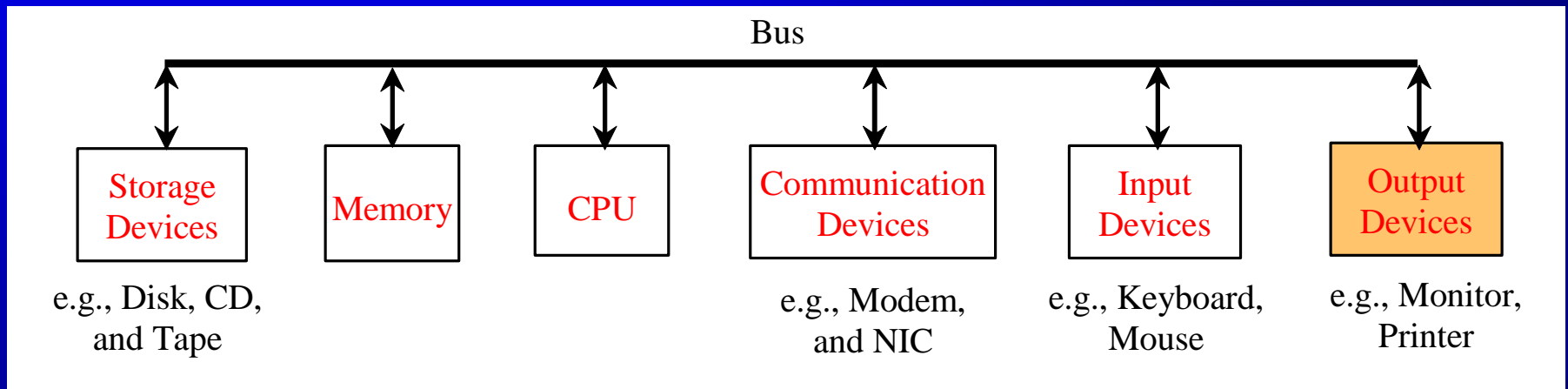
# Storage Devices

Memory is volatile, because information is lost when the power is off. Programs and data are permanently stored on storage devices and are moved to memory when the computer actually uses them. There are three main types of storage devices: Disk drives (hard disks and floppy disks), CD drives (CD-R and CD-RW), and Tape drives.



# Output Devices: Monitor

The monitor displays information (text and graphics). The resolution and dot pitch determine the quality of the display.

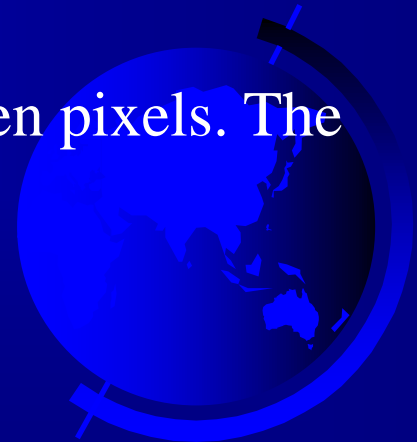




# Monitor Resolution and Dot Pitch

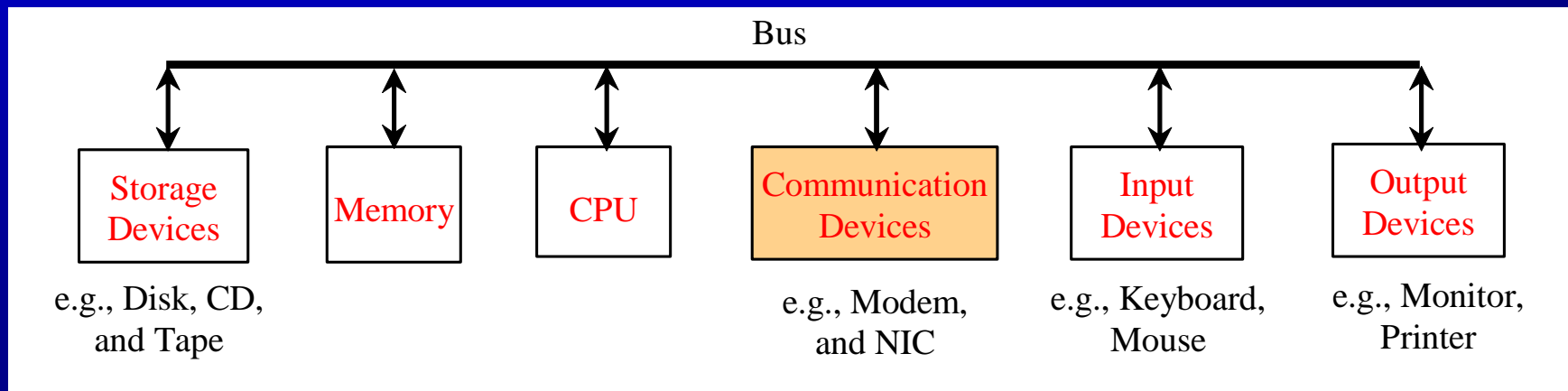
*resolution* The *resolution* specifies the number of pixels per square inch. Pixels (short for “picture elements”) are tiny dots that form an image on the screen. The resolution can be set manually. The higher the resolution, the sharper and clearer the image is. However, the image may be very small if you set high resolution on a small screen monitor. PC monitors are usually 15-inch, 17-inch, 19-inch, or 21-inch. For a 15-inch monitor, a comfortable resolution setting would be 640×480 (307,200 pixels).

*dot pitch* The *dot pitch* is the amount of space between pixels. The smaller the dot pitch, the better the display.



# Communication Devices

A *regular modem* uses a phone line and can transfer data in a speed up to 56,000 bps (bits per second). A *DSL* (digital subscriber line) also uses a phone line and can transfer data in a speed 20 times faster than a regular modem. A *cable modem* uses the TV cable line maintained by the cable company. A cable modem is as fast as a DSL. Network interface card (*NIC*) is a device to connect a computer to a local area network (LAN). The LAN is commonly used in business, universities, and government organizations. A typical type of NIC, called *10BaseT*, can transfer data at 10 mbps (million bits per second).



# Programs

Computer *programs*, known as *software*, are instructions to the computer.

You tell a computer what to do through programs. Without programs, a computer is an empty machine. Computers do not understand human languages, so you need to use computer languages to communicate with them.

Programs are written using programming languages.



# Programming Languages

Machine Language    Assembly Language    High-Level Language

Machine language is a set of primitive instructions built into every computer. The instructions are in the form of binary code, so you have to enter binary codes for various instructions. Program with native machine language is a tedious process. Moreover the programs are highly difficult to read and modify. For example, to add two numbers, you might write an instruction in binary like this:

```
1101101010011010
```

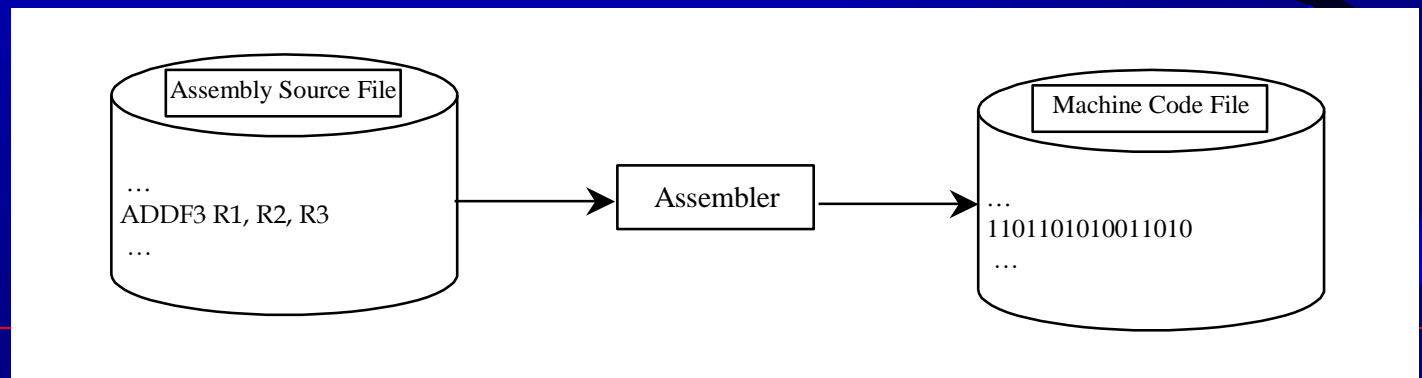


# Programming Languages

Machine Language    Assembly Language    High-Level Language

Assembly languages were developed to make programming easy. Since the computer cannot understand assembly language, however, a program called assembler is used to convert assembly language programs into machine code. For example, to add two numbers, you might write an instruction in assembly code like this:

```
ADDF3 R1, R2, R3
```



# Programming Languages

Machine Language    Assembly Language    High-Level Language

The high-level languages are English-like and easy to learn and program. For example, the following is a high-level language statement that computes the area of a circle with radius 5:

```
area = 5 * 5 * 3.1415;
```



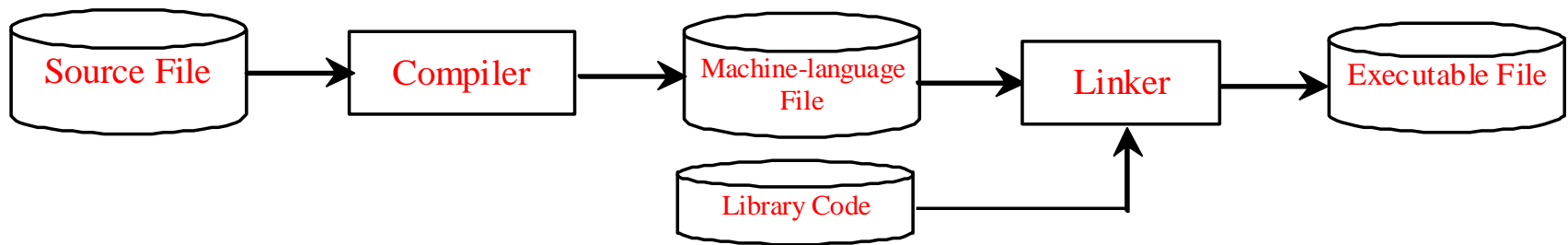
# Popular High-Level Languages

- COBOL (COmmon Business Oriented Language)
- FORTRAN (FORmula TRANslation)
- BASIC (Beginner All-purpose Symbolic Instructional Code)
- Pascal (named for Blaise Pascal)
- Ada (named for Ada Lovelace)
- C (whose developer designed B first)
- Visual Basic (Basic-like visual language developed by Microsoft)
- Delphi (Pascal-like visual language developed by Borland)
- C++ (an object-oriented language, based on C)
- C# (a Python-like language developed by Microsoft)
- Python (We use it in the book)



# Compiling Source Code

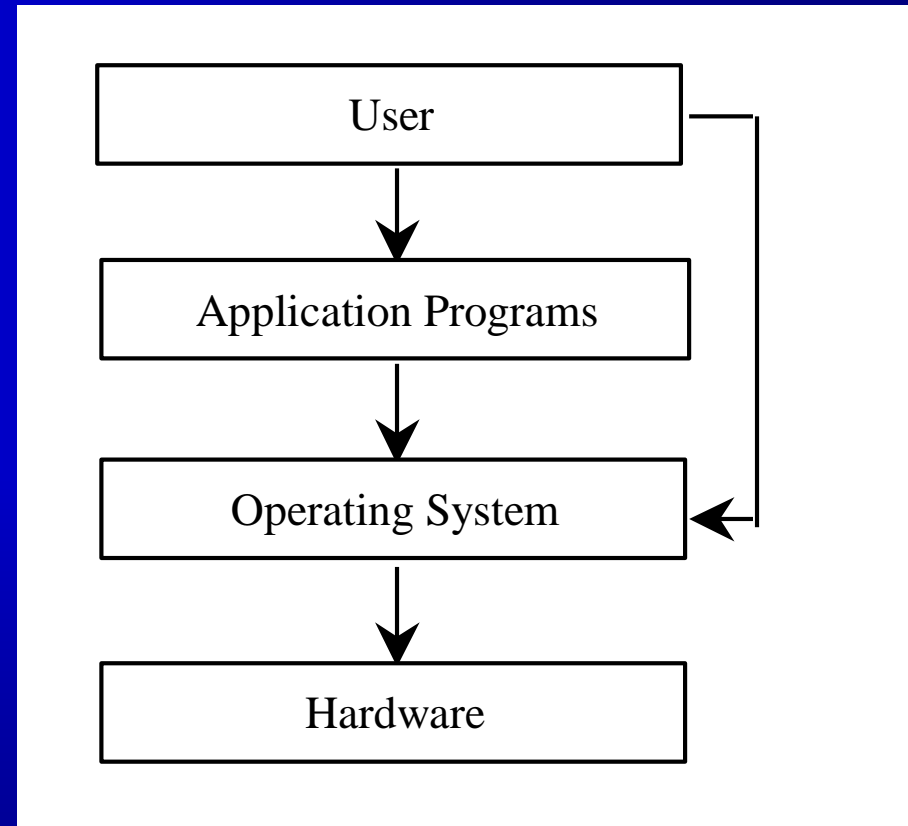
A program written in a high-level language is called a *source program*. Since a computer cannot understand a source program. Program called a *compiler* is used to translate the source program into a machine language program called an *object program*. The object program is often then linked with other supporting library code before the object can be executed on the machine.





# Operating Systems

The *operating system* (OS) is a program that manages and controls a computer's activities. You are probably using Windows 98, NT, 2000, XP, or ME. Windows is currently the most popular PC operating system. Application programs such as an Internet browser and a word processor cannot run without an operating system.



# What is Python?

General Purpose    Interpreted    Object-Oriented

Python is a general purpose programming language. That means you can use Python to write code for any programming tasks. Python are now used in Google search engine, in mission critical projects in NASA, in processing financial transactions at New York Stock Exchange.



# What is Python?

General Purpose    Interpreted    Object-Oriented

Python is interpreted, which means that python code is translated and executed by an interpreter one statement at a time. In a compiled language, the entire source code is compiled and then executed altogether.



# What is Python?

General Purpose    Interpreted    Object-Oriented

Python is an object-oriented programming language. Data in Python are objects created from classes. A class is essentially a type that defines the objects of the same kind with properties and methods for manipulating objects. Object-oriented programming is a powerful tool for developing reusable software.



# Python's History

- ❑ created by Guido van Rossum in Netherlands in 1990
- ❑ Open source



# Python 2 vs. Python 3

Python 3 is a newer version, but it is not backward compatible with Python 2. That means if you write a program using Python 2, it may not work on Python 3.



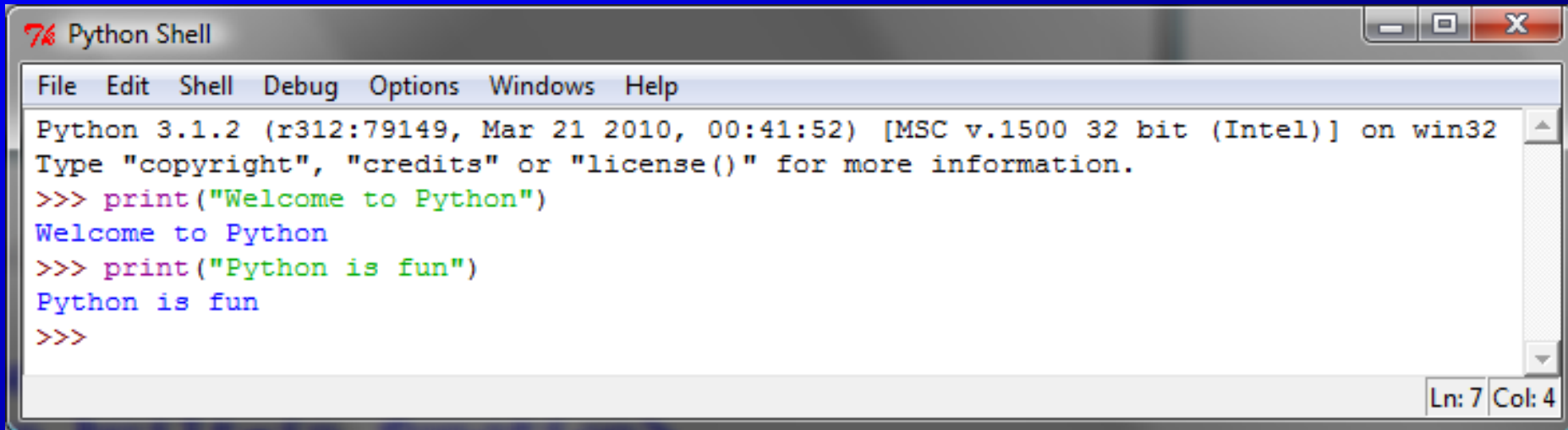
# Launch Python

```
Administrator: Command Prompt
C:\>python
Python 3.1.2 (r312:79149, Mar 21 2010, 00:41:52) [MSC v.1500 32 bit (Intel)] on
win32
Type "help", "copyright", "credits" or "license" for more information.
>>> print("Welcome to Python")
Welcome to Python
>>> print("Python is fun")
Python is fun
>>> ^Z

C:\>
```



# Launch Python IDLE



The screenshot shows a window titled "Python Shell" with a menu bar containing "File", "Edit", "Shell", "Debug", "Options", "Windows", and "Help". The main text area displays the following text:

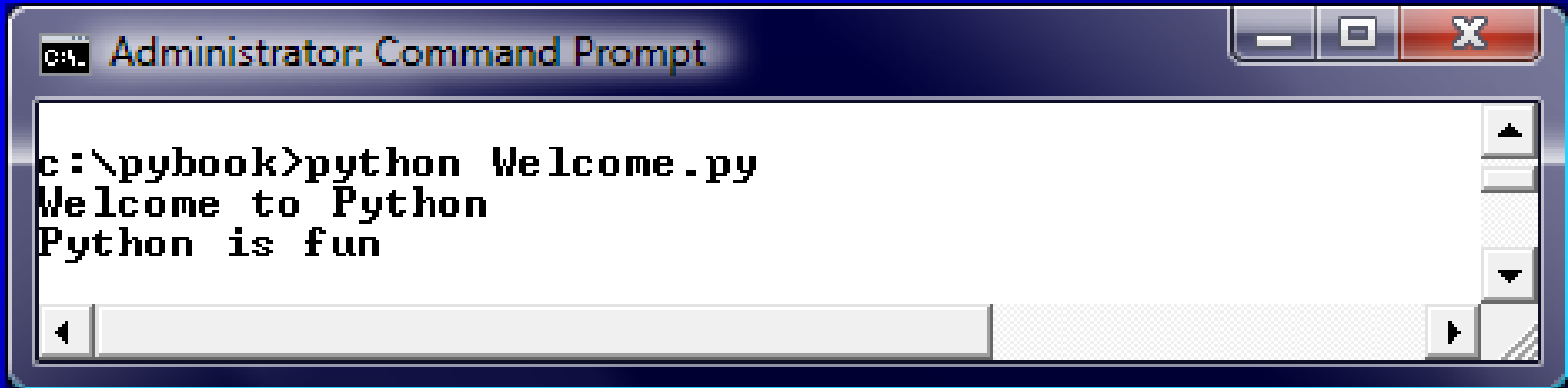
```
Python 3.1.2 (r312:79149, Mar 21 2010, 00:41:52) [MSC v.1500 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> print("Welcome to Python")
Welcome to Python
>>> print("Python is fun")
Python is fun
>>>
```

The status bar at the bottom right of the window shows "Ln: 7 Col: 4".





# Run Python Script



The image shows a screenshot of a Windows Command Prompt window titled "Administrator: Command Prompt". The window has standard Windows window controls (minimize, maximize, close) in the top right corner. The command prompt shows the following text:

```
c:\pybook>python Welcome.py  
Welcome to Python  
Python is fun
```

The text is displayed in a monospaced font. The prompt character is a greater-than sign (>). The output consists of two lines: "Welcome to Python" and "Python is fun". The window has a scroll bar on the right side and a horizontal scroll bar at the bottom.



# A Simple Python Program

## Listing 1.1

```
# Display two messages
print("Welcome to Python")
print("Python is fun")
```

Welcome

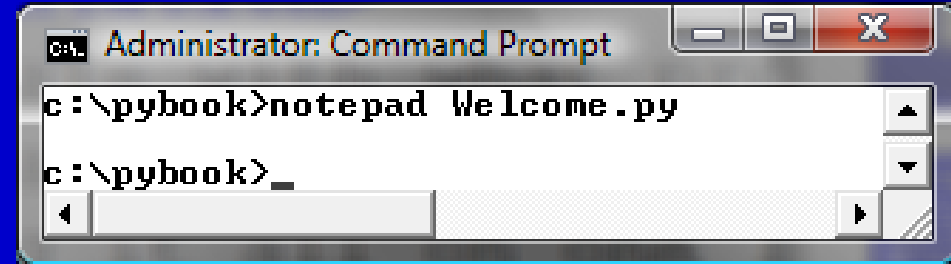
Run

### IMPORTANT NOTE:

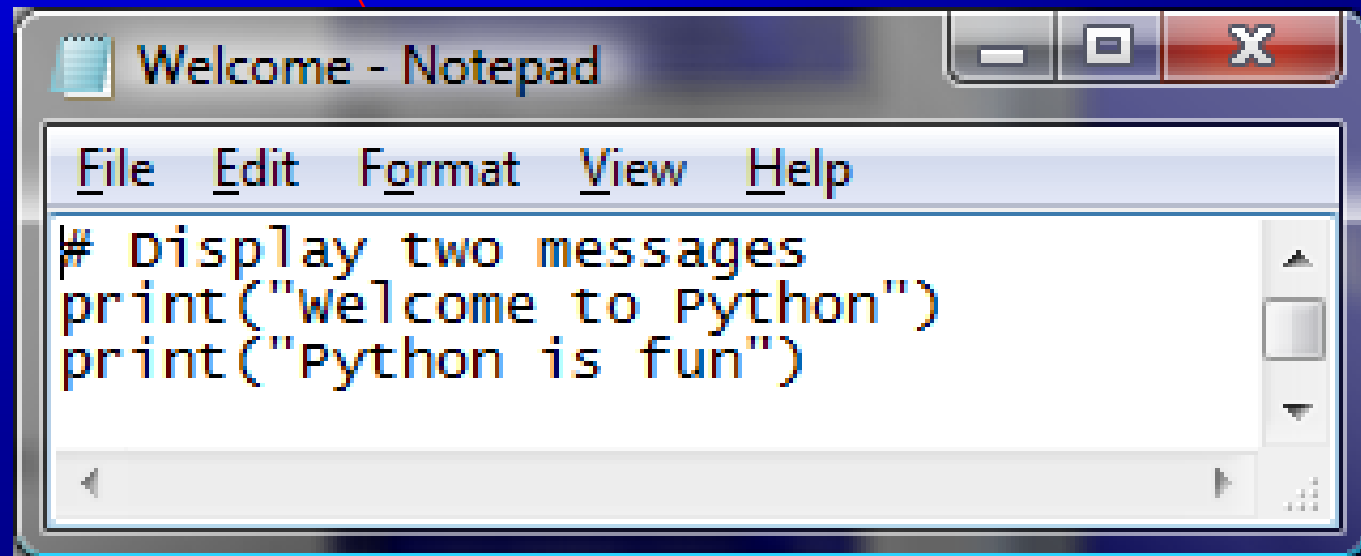
(1) To enable the buttons, you must download the entire slide file slide.zip and unzip the files into a directory (e.g., c:\slide). (2) You must have installed Python and set python bin directory in the environment path. (3) If you are using Office 2010, check PowerPoint2010.doc located in the same folder with this ppt file.

# Creating and Editing Using Notepad

To use Notepad, type  
notepad Welcome.py  
from the DOS prompt.



```
Administrator: Command Prompt
c:\pybook>notepad Welcome.py
c:\pybook>_
```



```
Welcome - Notepad
File Edit Format View Help
# Display two messages
print("welcome to Python")
print("Python is fun")
```

# Trace a Program Execution

Execute a statement

```
# Display two messages  
print("Welcome to Python")  
print("Python is fun")
```

# Trace a Program Execution

Execute a statement

```
# Display two messages  
print("Welcome to Python")  
print("Python is fun")
```

# Two More Simple Examples

WelcomeWithThreeMessages

Run

ComputeExpression

Run



# Supplements on the Companion Website

- See Supplement I.B for installing and configuring Python
- See Supplement I.C for developing Python programs from Eclipse

[www.cs.armstrong.edu/liang/py](http://www.cs.armstrong.edu/liang/py)



# Anatomy of a Python Program

- Statements
- Comments
- Indentation





# Statement

A statement represents an action or a sequence of actions. The statement `print("Welcome to Python")` in the program in Listing 1.1 is a statement to display the greeting "Welcome to Python".

```
# Display two messages  
print("Welcome to Python")  
print("Python is fun")
```

# Indentation

The indentation matters in Python. Note that the statements are entered from the first column in the new line. It would cause an error if the program is typed as follows:

```
# Display two messages
  print("Welcome to Python")
print("Python is fun")
```

# Special Symbols

Character Name	Description
()	Opening and closing parentheses Used with functions.
#	Pound sign Precedes a comment line.
" "	Opening and closing quotation marks Enclosing a string (i.e., sequence of characters).
''' '''	Opening and closing quotation marks Enclosing a paragraph comment.



# Programming Style and Documentation

- Appropriate Comments
- Proper Indentation and Spacing Lines



# Appropriate Comments

Include a summary at the beginning of the program to explain what the program does, its key features, its supporting data structures, and any unique techniques it uses.

Include your name, class section, instructor, date, and a brief description at the beginning of the program.



# Proper Indentation and Spacing

## □ Indentation

- Indent four spaces.
- A consistent spacing style makes programs clear and easy to read, debug, and maintain.

## □ Spacing

- Use blank line to separate segments of the code.



# Programming Errors

- Syntax Errors
  - Error in code construction
- Runtime Errors
  - Causes the program to abort
- Logic Errors
  - Produces incorrect result



# Getting Started with GUI Programming

Why GUI?

Turtle

Tkinter

GUI is a great pedagogical tool to motivate students and stimulate student interests in programming.





# Getting Started with GUI Programming

Why GUI?

Turtle

Tkniter

A simple way to start graphics programming is to use Python built-in Turtle package.

A Turtle Example

Run



# Getting Started with GUI Programming

Why GUI?

Turtle

Tkinter

Later in the book, we will also introduce Tkinter for developing comprehensive GUI applications.

A Tkinter Example

Run

