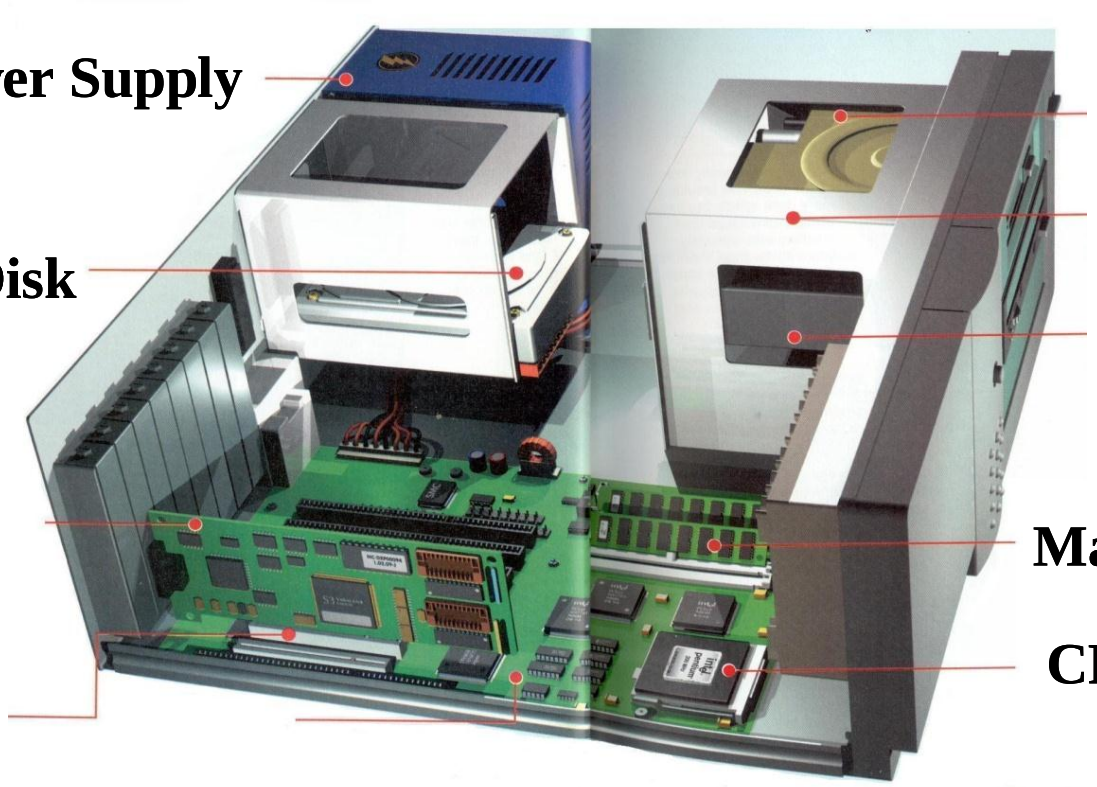
CSIS/COMP 1117B
Computer Programming

Some Basic Computer Concepts

Some Basic Computer Concepts

- A typical desktop computer system
- Programming languages: high and low
- Software development
  - what this course really is about!

A Typical Desktop Computer System



The Central Processing Unit (1)

- The “brain” of the entire system
  - controls and coordinates the operation of the other components: main memory, other storage elements, input/output devices
- Operating under the control of *a* program residing in the main memory
  - a program is a collection of (machine specific) instructions and the associated data
  - instructions are very primitive and very often thousands are needed to be executed to do something as simple as displaying a letter on the screen

The Central Processing Unit (2)

- Has an internal storage, called **registers**, for information that is needed for instruction executions
  - instructions currently being executed
  - data to be operated on
- Mirrors parts of the main memory in **caches** to reduce waiting time for instructions/data to be transferred in or out of the main memory
  - separate caches for data and instruction
  - multiple levels with different cost/performance characteristics

The Main Memory

- A linear array of (***volatile***) storage elements each identified by an address (a number)
- Each element is typically made up of 8 binary digits (**bits**) called a **byte**, and can be used to store a character (letters, digits, punctuation marks, special symbols, etc.)
- Larger data elements are stored in consecutive bytes, e.g., 4 or 8 bytes for an integer
- Typical sizes of the main memory ranges from 4GB to 16GB (with “servers” typically having hundreds)
  - 1G = 1,024M
  - 1M = 1,024K
  - 1K = 1,024

The Hard Disk

- Provides *long term* information storage.
- Information organized as **files**
  - a collection of **records**
  - each record made up of **fields** of (possibly) different types
  - an **abstraction** implemented by the **operating system**
- "Physical" organization of a typical disk
  - a set of one or more (up to 5) disks with one set of read/write heads (1 per recording surface, 2 per disk)
  - concentric circles (**track**s) divided into fixed size blocks (call **sector**s)
  - a collection of tracks on different recording surfaces at the same distance from the centre is called a **cylinder**
- Typical capacity ranges from 500GB to 6TB.

  Why is it that my 2TB disk is reported to have a capacity of 1.81TB (2,000,291,885,056 bytes)?

Other Peripheral Devices

- Other storage devices
  - CD/DVD/BD read/writer
  - other solid-state storage devices, e.g., thumb drive
- Input devices
  - keyboard
  - pointing devices: mouse, track ball, thumb pad, stylus
- Output devices
  - monitor
  - printer
  - speaker
- *Where is the all-important **network**?*

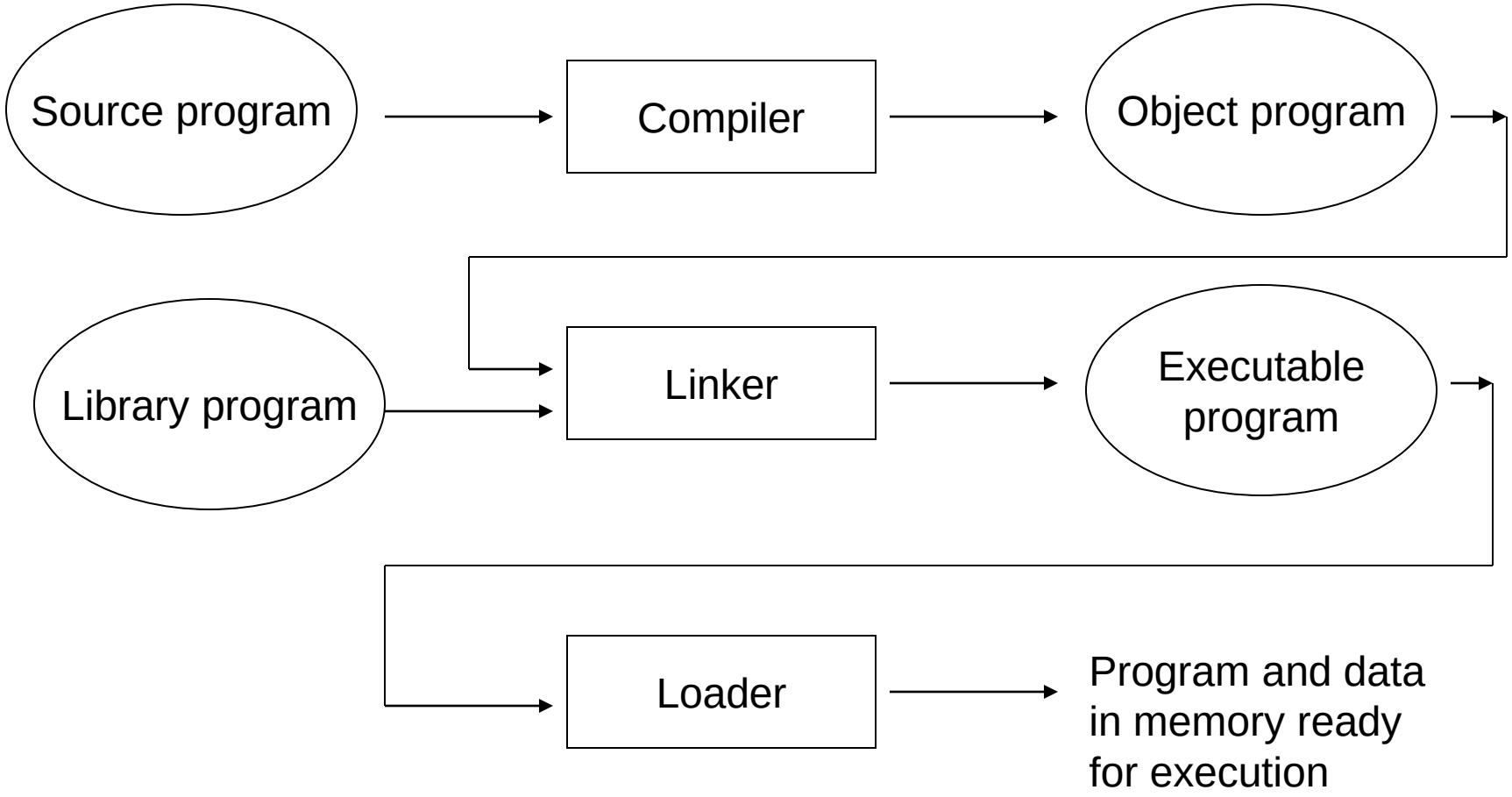
High-Level Programming Languages

- Programming a computer in machine instructions is basically ***impossible*** (*why?*)
- Need languages that are easier on the human programmers
  - more readable and easier to write
  - facilitates adaptation of solutions (**algorithm**s) for computers to execute
  - may be problem domain specific
    - Internet: Java, C#, Python, JavaScript, XHTML
    - scientific: Fortran, C/C++
    - smart phones: Objective-C
    - other niche: Lisp/Scheme, Prolog, ML

A Simple C++ Program

```
// A simple C++ program for temperature conversion
#include <iostream>
using namespace std;
int main()  {
    int  f,  c;     // declarations
    cout << "Enter the temperature in Fahrenheit: ";
    cin >> f;     // accepts input from user and place in f
    c = (f - 32) * 5 / 9;
    cout << "The temperature in Centigrade is: " << c << '\n';
    system("PAUSE");    // waits for user to hit <return>
    return 0;
}
```

The Compilation Process



Syntax

- The **structure** of a language.
- A **context-free grammar** is typically used to specify structural rules of a programming language.
  - consists of a set of terminal symbols and a set of non-terminal symbols together with a set of grammar (production) rules
- A compiler typically performs its translation under the guidance of the grammar for the specific language being translated.
- A grammar rule has two sides:
  - the left-hand-side: the object (non-terminal) being specified
  - the right-hand-side: the composition of the left-hand-side object as a sequence of terminal / non-terminal symbols

A simple Grammar for Statements

<statement> → <assign stmt> | <compound stmt>

<assign stmt> → <id> ‘=‘ <expression> ‘;’

<compound stmt> → ‘{‘ { <statement> } ‘}’

<expression> → <expression > (‘+’ | ‘–’) <term>

<term> → <term> (‘*’ | ‘/’) <factor>

<factor> → <id> | ‘(‘ <expression> ‘)’

Semantics

- **Meaning** of various constructs of a language.
- Typically given (rather informally) in natural language, e.g.,
  - an assignment statement:

    <assign stmt> $\rightarrow$ <id> ‘=‘ <expression> ‘;’
  - its meaning is

    evaluate <expression> and then store the result in the memory location represented by <id>

Software Development

- Software Life Cycle
  - analysis and specification of the task
  - design of the software
  - implementation
  - testing
  - maintenance
  - obsolescence
- We are mainly concerned with the first four.

Design vs. Implementation

- An **algorithm** is a sequence of precise instructions that leads to a solution (similar to a cooking recipe). (**Design phase**)
  - a recipe is a set of instructions for a person to carry out, e.g., a recipe for preparing a fish dish:
    1. Clean a 1-pound fish.
    2. Cut the fish into two halves.
    3. Fry the fillets in boiling oil for 6 minutes.
    4. Add soy sauce and green onions right before serving.
- A computer **program** is a set of precise instructions for a computer to execute. (**Implementation phase**)

Integrated Development Environment

- Software that integrates the steps of developing a program (e.g., editing, compiling, linking, debugging) into a single package.
- Dev-C++
  - an window-base IDE for C++ programming
  - this software is free and can be easily downloaded from the Internet

What about the Operating System?

- It is the program that is executing when you power up a computer (or a smart phone!)
- It manages the various hardware resources and makes them ***more usable***.
- The functions of an operating system that are most relevant to you for this course is:
  - execute programs: such as the IDE, your programs
  - manage files which you used to store your programs