CSC180: Lecture 2

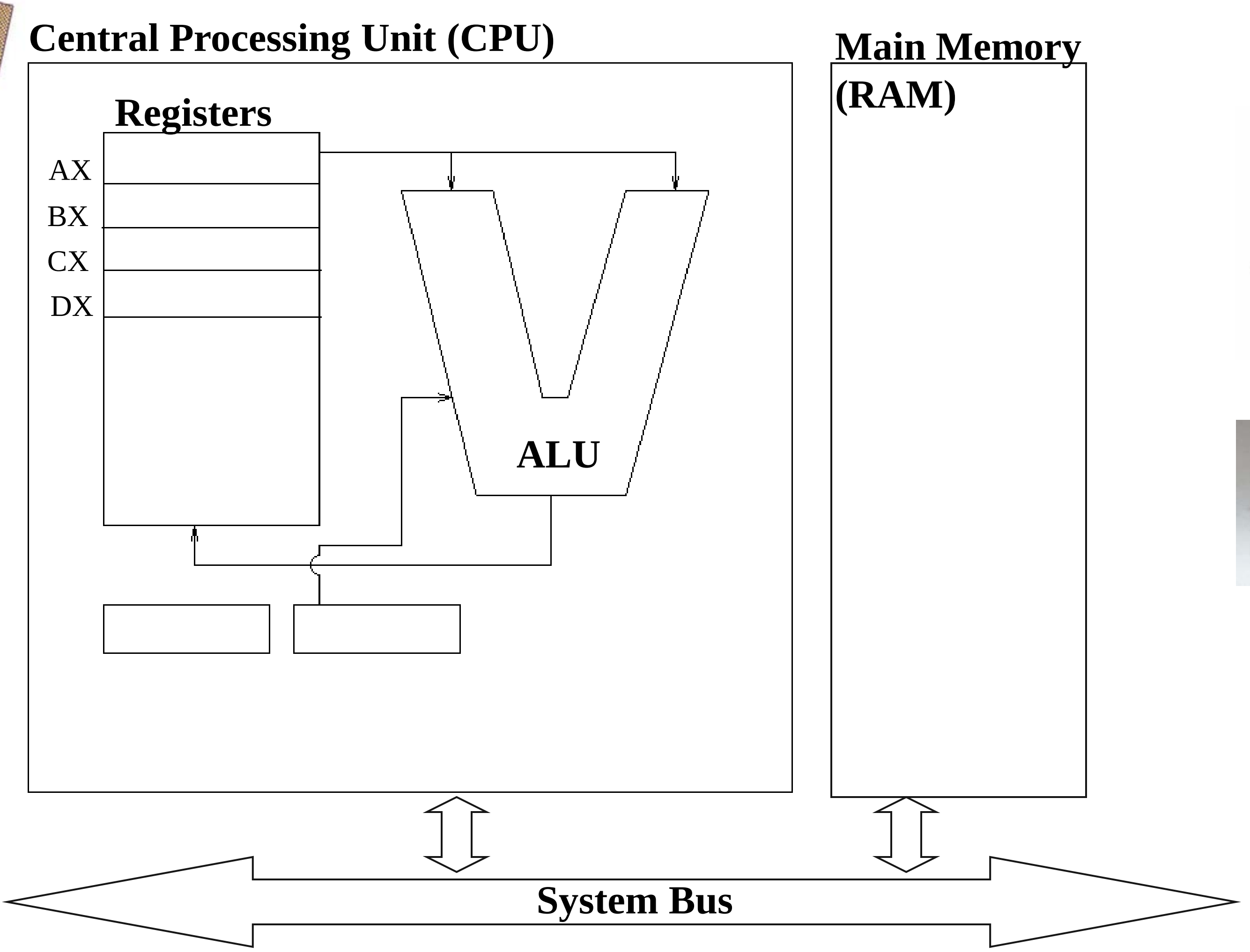
Wael Aboulsaadat

wael@cs.toronto.edu

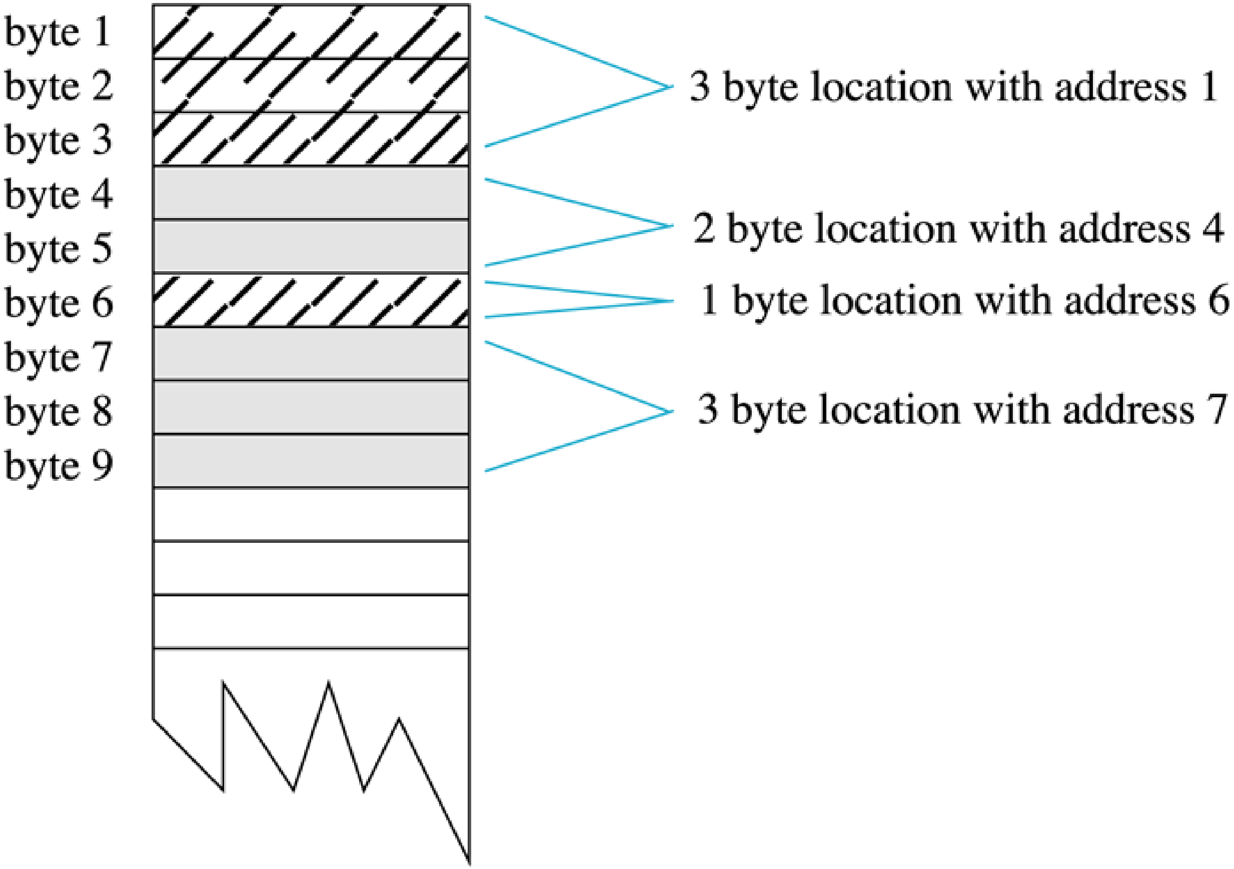
http://portal.utoronto.ca/

Acknowledgement: These slides are partially based on the slides supplied with Prof. Savitch book: Problem Solving with C

The Processor



Memory Locations and Bytes


byte 1
byte 2
byte 3
byte 4
byte 5
byte 6
byte 7
byte 8
byte 9
3 byte location with address 1
2 byte location with address 4
1 byte location with address 6
3 byte location with address 7

Secondary Memory

- Main memory stores instructions and data while a program is running.

- Secondary memory
  - Stores instructions and data between sessions
  - A file stores data or instructions in secondary memory

Memory Access

- Random Access
  - Usually called RAM
    - Computer can directly access any memory location
- Sequential Access
  - Data is generally found by searching through other items first
    - More common in secondary memory

Computer Software

- The operating system
  - Allows us to communicate with the computer
  - Is a program
  - Allocates the computer’s resources
  - Responds to user requests to run other programs

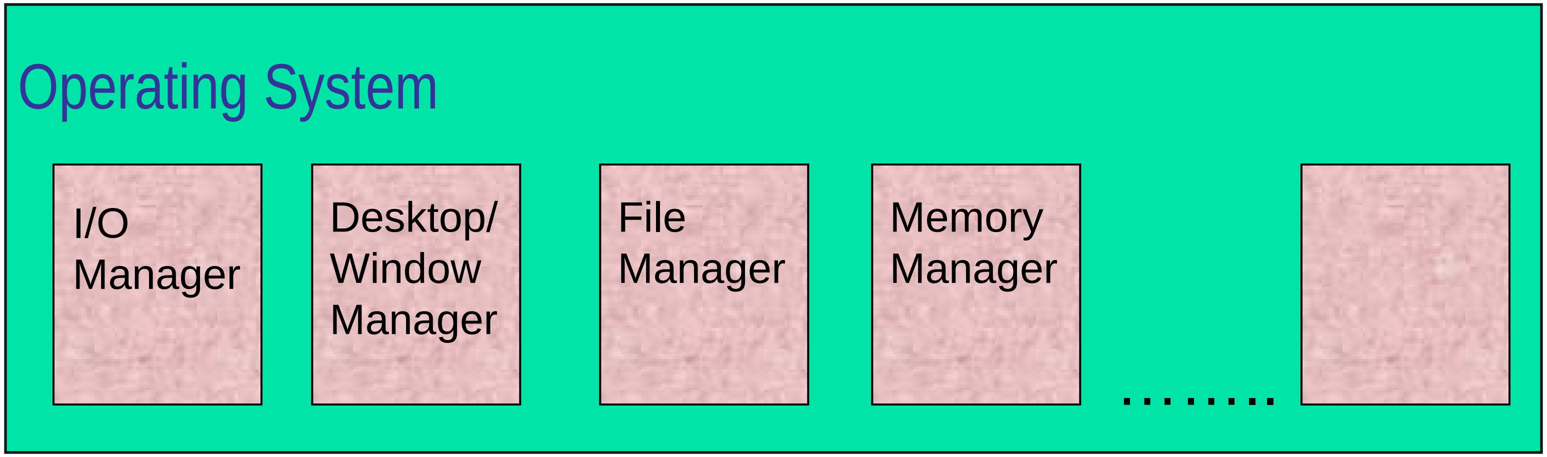
- Common operating systems include…
  - UNIX Linux DOS
Windows Macintosh VMS

Computer Software is of 2 types

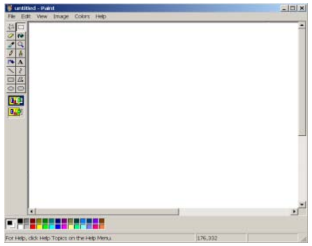
- Independent programs
  - Used by humans

- Libraries
  - Used by programs

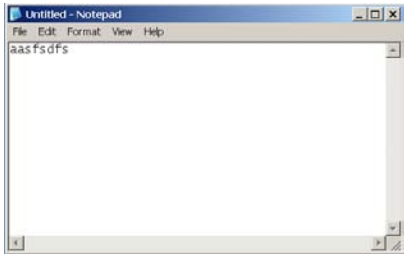
Operating System



Operating System


untitled - Paint
File Edit View Image Colors Help
For Help, click Help Topics on the Help Menu.
176,332

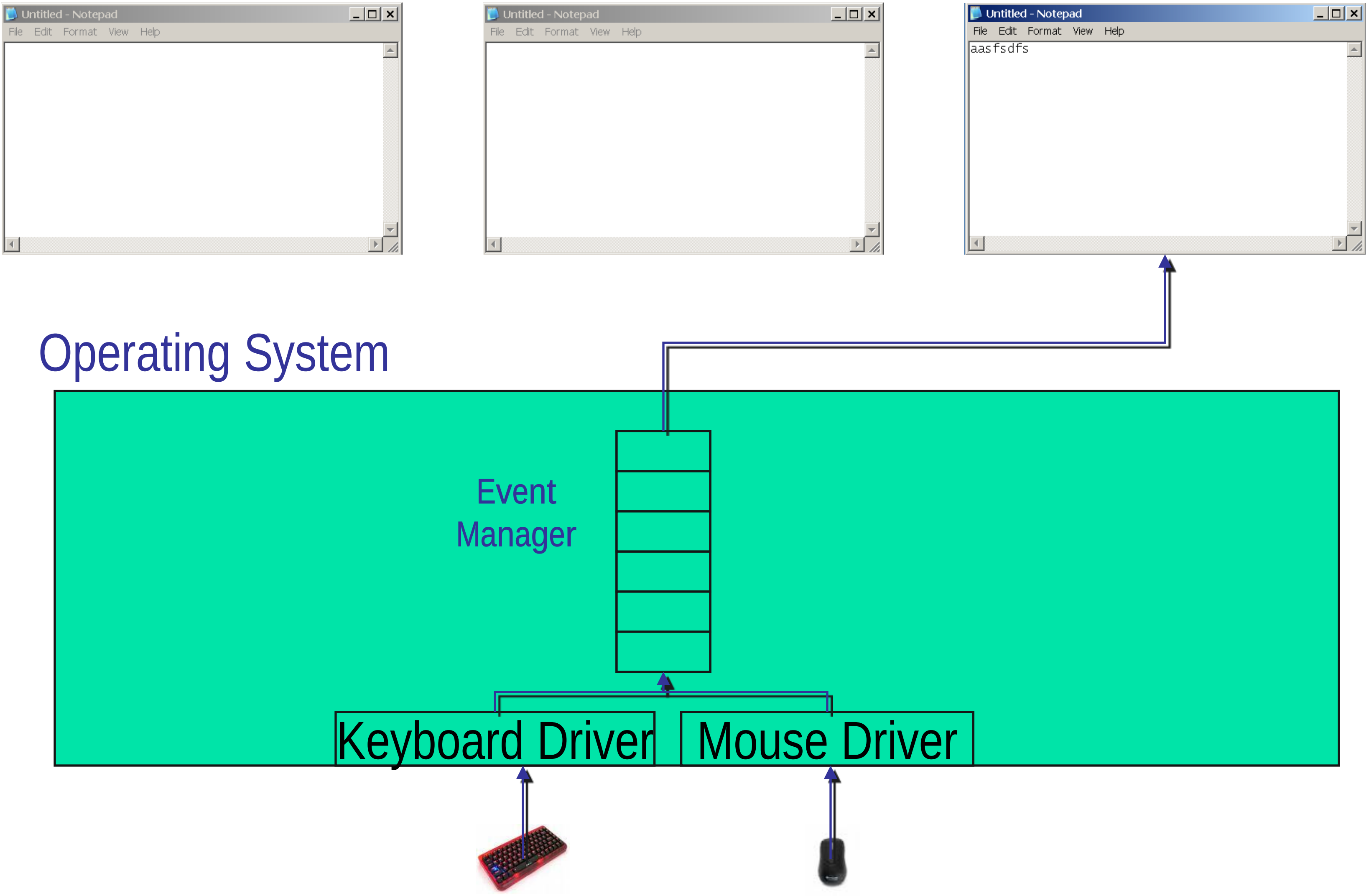
Google - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://www.google.ca/
Getting Started
Latest Headlines
iGoogle | Sign in
Google Canada
Web Images Groups News Maps Scholar more »
Google Search
I'm Feeling Lucky
Advanced Search
Preferences
Language Tools
Search: the web pages from Canada
Google.ca offered in: Français
Advertising Programs - Business Solutions - About Google - Go to Google.com
©2007 Google
Done



aasfsdfs

Operating System

Operating System



Untitled - Notepad
File Edit Format View Help
Untitled - Notepad
File Edit Format View Help
Untitled - Notepad
File Edit Format View Help

aasfsdfs

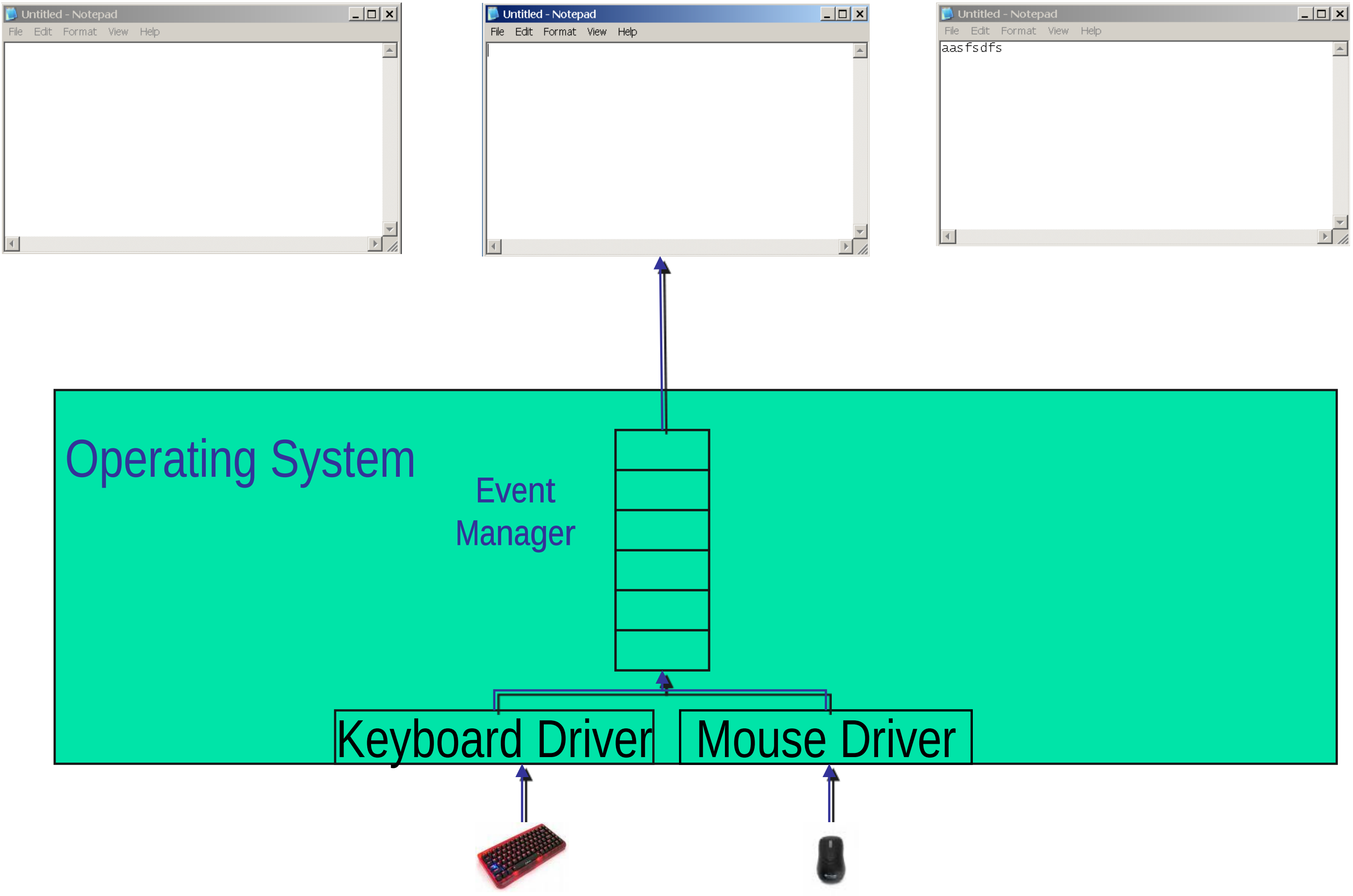
Operating System

Event Manager

Keyboard Driver

Mouse Driver

Operating System


Untitled - Notepad
File Edit Format View Help
Untitled - Notepad
File Edit Format View Help
Untitled - Notepad
File Edit Format View Help
aasfsdfs

Operating System

Event Manager

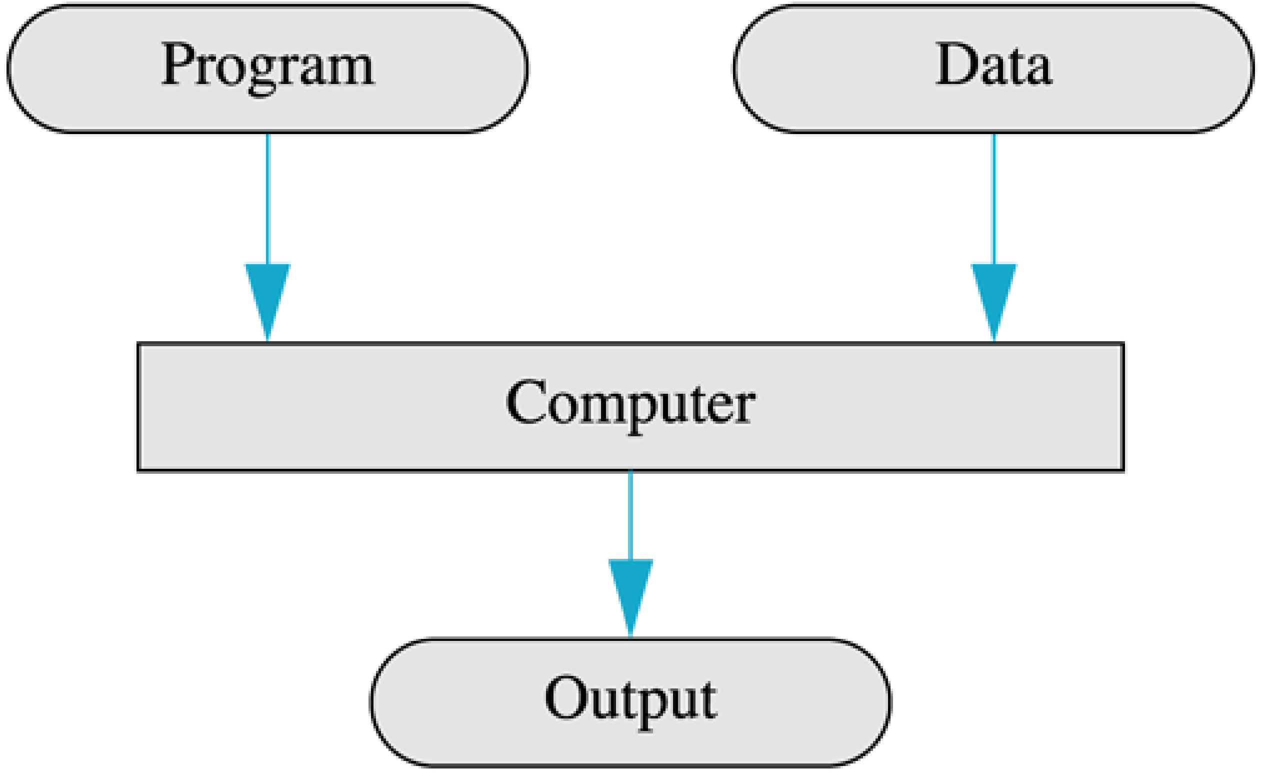
Keyboard Driver

Mouse Driver

Computer Input

- Computer input consists of
  - A program
  - Some data

Simple View of Running a Program



High-level Languages

- Common programming languages include …

  C C++ Java Pascal Visual Basic FORTRAN
  COBOL Lisp Scheme Ada

- These high – level languages
  - Resemble human languages
  - Are designed to be easy to read and write
  - Use more complicated instructions than the CPU can follow
  - Must be translated to zeros and ones for the CPU to execute a program

Low-level Languages

- An assembly language command such as

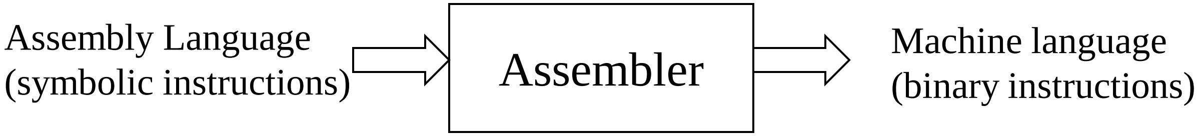
  ADD X Y Z

  might mean add the values found at x and y in memory, and store the result in location z.

- Assembly language must be translated to machine language (zeros and ones)

  0110 1001 1010 1011

- The CPU can follow machine language



Assembly Language
(symbolic instructions)

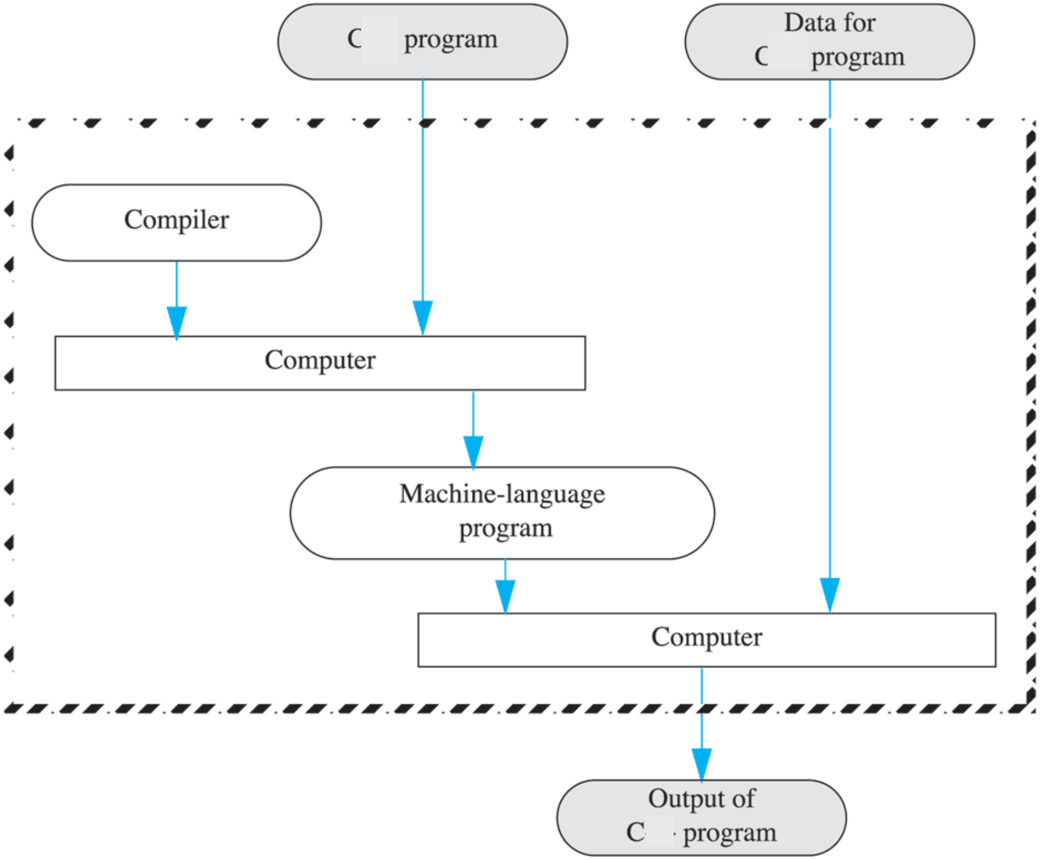
Assembler

Machine language
(binary instructions)

Compilers

- Translate high-level language to machine language
  - Source code
    - The original program in a high level language
  - Object code
    - The translated version in machine language

Compiling and Running a C Program (Basic Outline)



C program

Data for C program

Compiler

Computer

Machine-language program

Computer

Output of C program

Why do we need a Compiler ?

C program

```
#include<stdio.h>

main()
{
    printf("Hello World");
}
```

Assembly

```
 .file   "foo.c"
                .text
                .p2align 4,,15
.globl main
                .type   main, @function
main:           push   BP
                mov    $9, AX
                mov    SP, BP
                sub    $8, SP
                and    $-16, SP
                .p2align 4,,15
.L6:            dec    AX
                jns    .L6
                mov    BP, SP
                pop    BP
                ret
                .size   main, .-main
                .ident  "GCC: (GNU) 3.3.1"
```

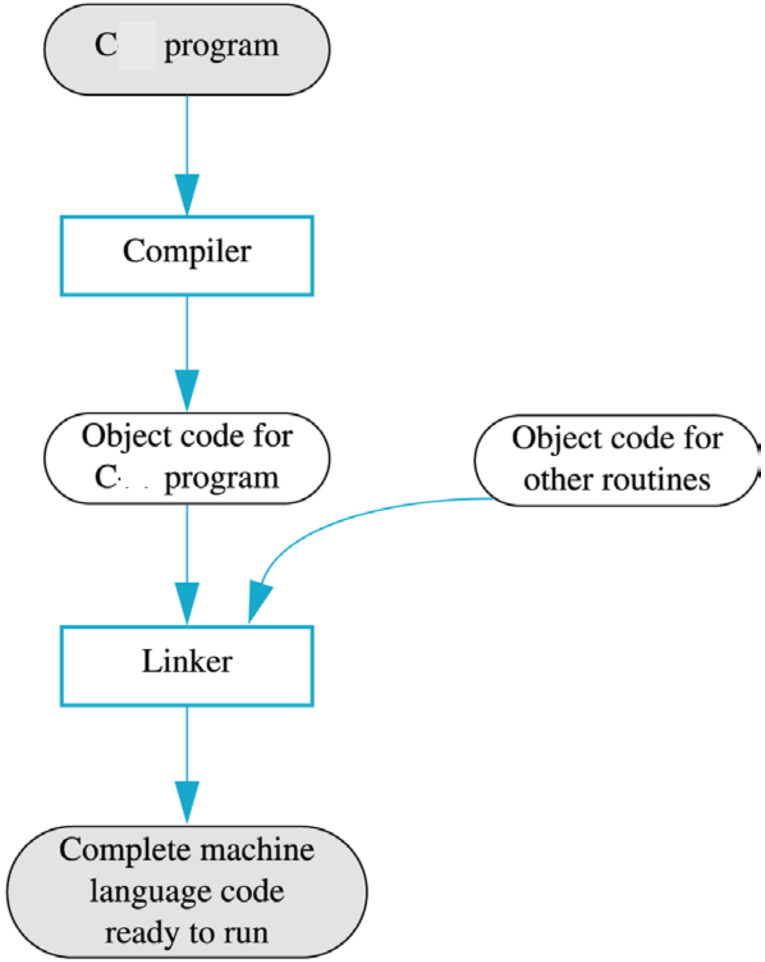
1s and 0s

```
01010101010001
10101010101111
10101001010101
10010101001000
00000001101111
00000000000000
11111111100001
```

Linkers

- Some programs we use are already compiled
  - Their object code is available for us to use
  - For example: Input and output routines
- A Linker combines
  - The object code for the programs we write
    and
  - The object code for the pre-compiled routines
    into
  - The machine language program the CPU can run

Preparing a C Program for Running



C program

Compiler

Object code for C program

Object code for other routines

Linker

Complete machine language code ready to run

So, what is “Memory”?

Memory is like a big table of numbered slots where bytes can be stored.

The number of a slot is its Address.
One byte Value can be stored in each slot.

Some “logical” data values span more than one slot, like the characters “Hello\n”

Addr	Value
0	
1	
2	
3	
4	‘H’ (72)
5	‘e’ (101)
6	‘l’ (108)
7	‘l’ (108)
8	‘o’ (111)
9	‘\n’ (10)
10	‘\0’ (0)
11	
12	

72?