

Basic Computing Concepts

Dr. Chia-Ling Tsai

Basic Concepts

- Hardware
- Hardware details
- Layout of a typical computer
- Software
- Binary code
- Programming languages

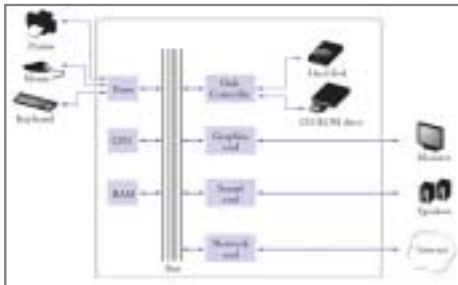
Hardware

- Physical components of a computer
- Includes
 - Internal components capable of
 - Storage of information
 - Both *data* and *programs*
 - Execution of logical commands (programs)
 - External components – devices – which connect the internal components to the user, allowing *input* and *output*

Hardware Details

- Central processing unit (CPU)
 - Chip
 - Transistors
- Storage
 - Primary storage: Random-access memory (RAM)
 - Secondary storage: e.g. hard disk
 - Removable storage devices: e.g. "thumb" drives, floppy disks, tapes, CDs, DVDs

Layout of a Typical Computer



Software

- Program – set of instructions written in a specific language that allows a computer to understand and perform an algorithm.
- Popular languages include C, C++, and Java.
- Gives a computer its flexibility
 - Different program allows different task to be performed.
- Converting an algorithm to a program is called *implementation*.

Implementation

- Converting an algorithm to a program should be straightforward if well-designed
 - Should become your second-nature soon!
- Pseudo-code:
Find-Maximum(A)
placeholders i, max
max ← A[1]
loop i from 2 to A.length, **begin**
 if max < A[i], **do**
 max ← A[i]
 end if
end loop i
return max

Implementation: Java vs Pascal

Java

```
int i, max;  
max = A[0];  
for (i=1; i<n; i++)  
{  
  if (A[i] > max)  
    max = A[i];  
}
```

Pascal

```
var  
  i, max: Integer;  
begin  
  max = A[1];  
  for i:=2 to n do begin  
    if A[i]>max then  
      max := A[i];  
  end;  
end.
```

Binary Code

- A computer is an *electronic* device – it doesn't "speak" *any* language
- Communication is by electronic *switches* (on-off)
- Switch positions can be represented by (encoded using) the digits 1 (on) and 0 (off)
 - Two positions, so called *binary code*
- Binary code is called *machine language*
 - This is the *only* language "understood" by a computer

Language Levels

- Working in machine language is extremely tedious for humans
 - Also, it's error-prone
- Easier-to-use languages were developed
 - *Low-level languages* are like machine language, but allow the use of *labels* instead of binary codes
 - *High-level languages* are closer to spoken languages, although without ambiguity and much more limited
- A program written in such a language must be *translated* into machine language

Low-level Language

- Assembly language
 - Labels can be used to help the programmer remember
 - Instruction names ("ADD" instead of 01000111)
 - Names for places where data are stored ("salary" instead of 11011010)
- Each instruction corresponds to an instruction in machine language, so translation is straightforward
- A program that does the translation is called an *assembler*

Low-level Language

- Example: MIPS architecture
 - Always 32 bits long
 - First 6-bits for the operation
 - The rest depends on the operation type

```
{ sp |          target address          |
  2   |          1024                      | decimal
00010 00010 00010 0000 0000 0000 0000 0000 | binary
```

Jump to address 1024

High-level Language

- More like ordinary spoken language
- But inflexible and unambiguous
- Easier for programmers to use than low-level language
- Harder to translate into machine language
 - One *statement* generally corresponds to many instructions
- A program that does the translation is called a *compiler*

Example

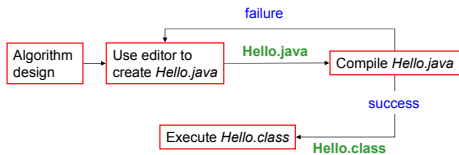
- English
If the interest rate is more than 100, print the message "Rate error"
- Machine language
00010101 00101000 00010000 01100100
10100011 11110000
- Assembly language
iload intRate
bipush 100
if_icmpgt 240
- High-level language (Java)
if (intRate > 100)
System.out.println("Rate error");

Java

- Java is a popular high-level language
- Java is reasonably easy to learn
 - It is used professionally, so it is not ideal for beginners
- Once you learn to program in one language, you can learn other languages much more easily
 - There are only a few constructs in programming
 - The syntax

Basics on Programming

- The cycle from an algorithm to an executable
 - Use an editor to implement the algorithm in a high-level language
 - Use a compiler to *translate* the program into a machine-language program
 - *Execute* ("run") the machine-language program



Program Errors

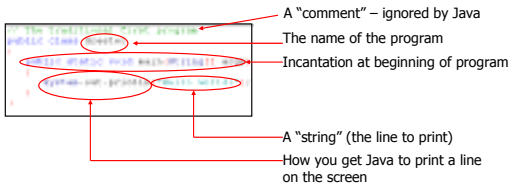
- Mistakes are easy to make
- Two main kinds of errors
 - Syntax errors caught by the compiler
 - Logic errors (you have to catch them yourself!)
- Word of advice:
 - It pays off to design and edit the program with extra care—*debugging can be very time-consuming!*

Getting started

- How to "log in" to the computer
- What tools are available at the College and for download at home, onto a laptop, etc.
- How to start up and use one or more of the tools to
 - *Edit* a Java program
 - *Compile* the program
 - *Execute* the compiled program
- Where and how to save your work

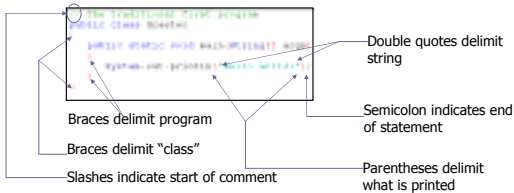
Hello World!

■ The traditional first program



Hello World!

■ Punctuation



Editing

- Type in the program using a simple editor
 - You can use any *plain text* editor, such as NotePad
 - You may *not* use word processor, such as Word, which saves text in a special format
- Always save your work regularly!

Editing

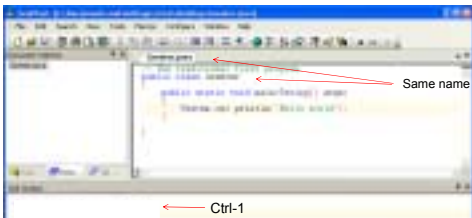
- Some editors can be configured to be “aware” of Java
 - TextPad is so configured at Iona, and you can download and configure your own copy (inexpensive, but not free) (<http://www.iona.edu/faculty/jmallozzi/TP6Setup.htm>, thanks to Dr. Mallozzi!)
 - Eclipse IDE for Java Developers (free): <http://www.eclipse.org/downloads/>
 - As you type, some help is given – indentation, highlighting of punctuation and key words of language
- Macros can be used to avoid some repetitive typing
- After you type the program, you can compile and execute it without leaving the editor

Command-line Mode: Compiling & Executing

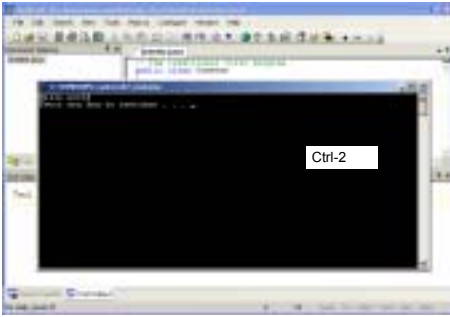
- Compiling: *javac HelloWorld.java*
 - Java Development Environment
 - <http://java.sun.com/javase/downloads/index.jsp>
 - Download the latest update, e.g. JDK 6 Update 6 (Windows Offline Installation)
- Executing: *java HelloWorld*
- Compile in Java-aware editors
 - TextPad (Install JDK first)
 - Eclipse (Install JRE first. Your machine might already have it.)

TextPad: Compiling

- Install JDK first
- To make the editor java-aware:
 - File->Open
 - Input the *File name* with a .java extension



TextPad:Executing



In-class Exercise

- Let's write a program together that prints the following pattern

```
*****  
*  
*****  
*  
*****
```

Eclipse: Compiling & Executing

- Install JRE first. (Your machine might already have it.)
- Saving the program triggers compilation



Summary

- Basic components of a computer
- Steps it takes from algorithm design to an executable
- Editing and compiling of Java programs using various methods.
