

Computer Science I with Java *Introduction*

- Computer science is a *discipline*
- What is computing?
- Mathematics of computing
- Networks, Internet
- Java
- Course topics

Central ideas

- Computer science is a *discipline*
- It aims at *solving problems*
- The tools include
 - *analysis* (understanding and breaking down problems)
 - *algorithms* (step-by-step plans)
 - *interaction*
- CS has a *rational* and an *empirical* side

CS I introduces a curriculum that includes:

- Hardware (architecture, organization)
- Operating systems and networks
- Data management
- Theory (logic, statistics, discrete math, algorithms, automata, AI)
- **Programming and software engineering**

Principles of computing: Computing mechanics

- *Concern*: structure and operation of computations; computation, communication, coordination, automation, recollection
- *Core technologies*: algorithms, models, compilers, languages, circuits, operating systems, databases, networks, software engineering, human-computer interaction, parallel computation, computer architecture, robotics, graphics, etc.

[Denning]

Design concerns

- *Simplicity* (via abstraction, structure)
- *Performance* (throughput, response time)
- *Reliability* (redundancy, recovery, integrity)
- *Evolvability* (adaptation to changes in function and scale)
- *Security* (access control privacy, authentication)
- *Design principles*: abstraction, information hiding, modularity, packages, version control, divide and conquer, layering, hierarchy, reuse, interfaces, encapsulation, virtual machines

[Denning]

Computing practices

Categories:

- *Programming* (using programming languages – programs \neq computer science!)
- *Engineering systems* (combining hardware and software)
- *Modeling and validation* (to predict and verify behavior)
- *Innovating* (generating durable changes in how systems and communities operate)
- *Applying* (building systems to support practical work)

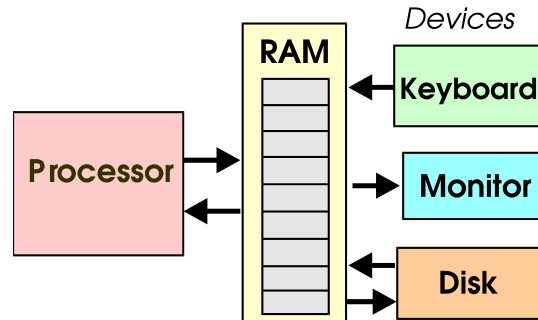
Computing power

- Brain:
 - ~ 10^{11} neurons
 - ~ 10^{15} synapses (connections)
 - ~ 10^{16} firings/sec (in parallel, 200 mph)
- PC:
 - 1 processor
 - ~ 10^1 registers (very fast storage)
 - ~ 10^9 bits RAM (fast memory)
 - ~ 10^{12} bits hard drive storage (slow)
 - ~ 10^9 operations per second
(serial; 72 million mph)

Models of computation

- A model is an *abstract* description that lets us focus on essentials
- The languages we discuss assume the *Random Access Machine* model, with *stored programs*
- Other models: neural network, finite automaton, Turing machine, parallel random access machine, cellular automaton, recursively definable function

Architecture of a computer



- Data flows as shown by arrows
- Data is communicated and stored as *bits* (on/off pulses or switches)

The chronic software crisis

- The global market demands new ways be found to work and do business
- New software is required rapidly
- Software development has tended to be behind schedule
- Software is often unreliable
- *Solutions*: design, documentation, readable code, structured techniques, object-oriented technology

Guiding ideas

- Algorithm design
- Documentation
- Modular decomposition
- Objects and classes of objects

Steps in problem solving

Repeat until problem is solved:

1. Specify the desired results
2. Design a solution
3. Code solution in programming language
4. Compile, test, debug

Kinds of *abstractions* used in CS I

- Variables and other expressions with values (variables occupy storage)
- *Data types* (string, integer, real)
- Control structures (e.g., loop, branch)
- Programs, subprograms (methods)
- *Classes*: specify categories of objects, e.g., *student*, *course*
- *Objects*: instances of classes

Computation and symbol manipulation

- **Symbol:** an abstraction by which we may represent parts of the real world
- *Example:* {0,1} is a set of symbols
- This set may be used to build symbols of any complexity (numerals, words, pix...)
- We can *operate* on symbols (add, concatenate, reorder, etc.)
- A computation can combine sequences, branches, loops

Computations may or may not have *input*

- *Problem:* Display interest charged when buying a \$20,000 car over 3 years at 10%
- *Solution:*
$$Y1 = 20,000 \times 0.1$$
$$Y2 = Y1 + Y1 \times 0.1$$
$$Y3 = Y2 + Y2 \times 0.1$$
$$Total = Y1 + Y2 + Y3$$
Display *Total*
- **Challenge:** Describe a version of the above with inputs

Problems

- Write an expression whose value is the area of the surface of a 2' x 3' x 5' box
- ... a box w feet wide, h feet high, and d feet deep
- Write an expression whose value is the 5% sales tax on a purchase of a \$199 hard drive and a \$79.95 software package
- ... A hard drive costing h dollars and software costing s dollars, taxed at t %

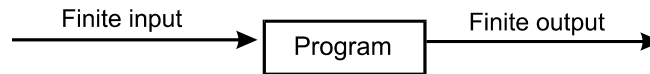
Computations may be *interactive*

- Typical menu- or command-driven user environment:

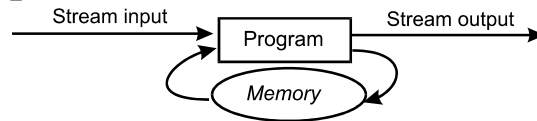
```
Repeat
  input a command
  execute command
until command = "quit"
```
- With interaction, input may depend on previous output
- Almost all computing today is interactive
- Internet computing presents new challenges related to interaction

3 computing paradigms

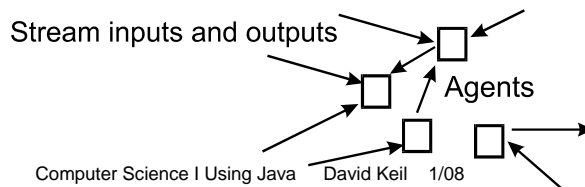
- Algorithmic



- Sequential interactive



- Multi-stream interactive



Number (numeral) systems

- Any natural number may be represented in many ways; e.g., twelve:

Unary 111111111111

Decimal 12

Octal (base-8) 14

Hexadecimal 0c

Binary 1100

- In all but the unary system, the value represented by a digit depends on its place in the numeral

Expression evaluation is algebraic

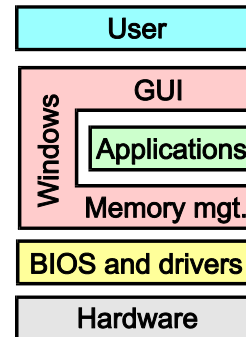
- *Problem:*
Let $a = 4$
Let $b = 2$
Find $a^2 - b + 5$
- *Solution:* $a^2 - b + 5$
 $= 16 - 2 + 5$
 $= 14 + 5$
 $= 19$
- *Algebra:* A set of values and a set of operators

Some sets we will use

- Truth values: $\{0,1\} = \{true, false\}$
- Digits: $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$
- Natural numbers (G. Peano):
 1. 0 is a natural number
 2. Every natural number has a successor natural number
 3. There are no other natural numbers
- Other sets: real numbers, integers, characters, strings, keywords, rules, operations

Operating-system and application software

- The operating system runs at all times
- It is started by a ROM “boot” process
- The OS may support multiple application programs concurrently
- The OS links applications to the hardware
- Both application programs and the OS have user interfaces
- OS examples: Win XP, Mac OS, UNIX



The stored-program model

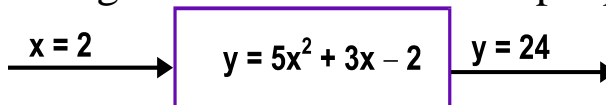
- The computer user chooses software and input data:



- A program in effect defines a machine:



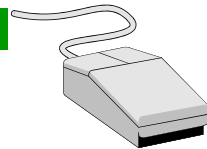
- An algebraic formula is a simple program:



Features of *graphical user environments* (e.g., Windows)

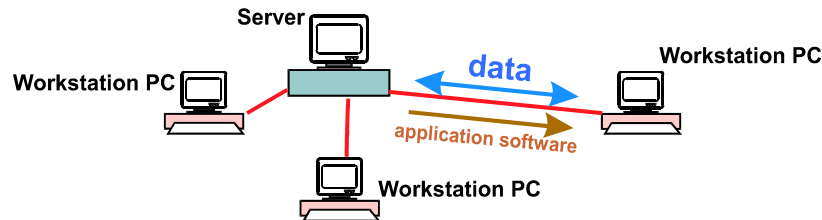
- Program loading (Start menu; icons, Run)
- Task switching (task bar)
- Device control (Start / Settings / Control panels)
- Disk directory tree structure
 - program files
 - data files
 - folders
- Menus, icons, dialog boxes
- Drag and drop
- Double-click to execute commands

Objects in a graphical user interface



- A window is an *object* with position, size, contents
- A window opens, closes, moves, resizes
- Other objects: icons, menus, buttons
- In Windows, clicking on an object with right mouse button gives user access to object's properties and methods (operations)

Networks connect PCs



- Cables and network cards connect local area networks (LANs)
- Each workstation PC runs its own software
- In *client-server* computing, the software (e.g., in response to database query) runs on the *server*

Internet computing

- The Internet is a network of computer networks
- It is connected by service providers and the telephone and TV-cable systems
- Email and Web data travels in *packets* (each with a small amount of data)
- We access Web pages by supplying a *uniform resource locator* (URL)
- The web-page data, in *Hypertext Markup Language* (HTML) form, is communicated to the user's computer

Extensions of HTML

- *Script languages* (e.g., Perl, CGI) improve interactivity
- *JavaScript* enables web site to execute algorithms designed by programmer
- JavaScript is *interpreted* by browser, whereas Java code is *compiled* into processor-executed *machine code*

Social and professional issues

- History of computing
- Social context of computing
- Methods and tools of analysis
- Professional and ethical responsibilities
- Risks and liabilities of computer-based systems
- Intellectual property
- Privacy and civil liberties

[Source: *Computing Curricula 2001*]

Java

- We will use the Java programming language to present many computer-science concepts
- You will get hands-on experience developing and debugging problem solutions using Java
- Java is an object-oriented language

CS I topics

1. Web-page design and JavaScript
2. Computer systems and hardware
3. Problem solving and program design
4. Java programming
5. Implementing classes
6. Branch and loop statements
7. Arrays and collections
8. Designing classes

References

Cay Horstmann. *Big Java*, 3rd Ed. Wiley, 2008.

ACM/IEEE. Computing Curricula 2001.

Peter J. Denning. Great principles of computing. *CACM* 46(11), November 2003, pp. 15-20.