**Chapter 0 (intro)**

Algorithm: set of steps that defines how a task is performed (discovered & represented in compatible form with machine)

Program: algorithm representation

Programming: process of developing a program, encoding it in machine form, and inserting into a machine

Software: programs and algorithms

Hardware: equipment/machinery

History of Algorithms

* The study of algorithms was originally a subject in mathematics
* Early examples of aglorithms: long division algorithm & Euclidean algorithm
* Godel’s Incompleteness theorem: some problems cannot be solved by algorithms

History of Computing

* Early computing devices
* Abacus: positions of beads represent numbers
* Gear-based machines (1600s-1800s)

Positions of gears represent numbers

Blaise Pascal, Wilhelm Leibniz, Charles Babbage (0.2)

Early Data Storage

* Punched cards
* First used in Jacquard Loom (1801) to store patterns for weaving cloth
* Storage of programs in Babbage’s Analytical Engine
* Popular through the 1970s
* Gear positions

Early Computers

* Based on mechanical relays
* 1940: Stibitz at Bell Laboratories
* 1944: Mark I: Howard Aiken and IBM at Harvard
* Based on vacuum tubes
* 1937-1941: Atanasoff-Berry at Iowa State
* 1940s: Colossus: secret German code-breaker
* 1940s: ENIAC: Mauchly & Eckert at U. of Penn.

Personal computers

* First used by hobbyists
* IBM introduced the PC in 1981
* Accepted by business
* Became the standard hardware design for most desktop computers
* Most PCs use software from Microsoft

Into the Millennium

* Internet revolutionized communications
* WWW
* Search engines
* Miniaturization of computing machines
* Embedded (GPS, in automobile engines)
* Smartphones

The Overarching Themes of Computer Science

* Computing technology effects:
* Governments, economics, scientific research, role of data, communication, …
* 7 “Big Ideas” that unite computer science:
* Algorithms, Abstraction, Creativity, Data, Programming, Internet and Impact

Algorithms

* The science of algorithms
* Draws from other subjects, including
* Math
* Engineering
* Psychology
* Business Administration

The central role of algorithms in computer science

Algorithms: limitations of, execution of, communication of, representation of, discovery of, analysis of, application of

Abstraction

* Abstraction: the distinction between the external properties of an entity and the details of the entity’s internal composition
* Abstract tool: a “component” that can be used without concern for the component’s internal properties

Creativity

* Computer science is inherently creative
* Discovering and applying algorithms is a human activity
* Extends forms of expression in many ways
* Creating large software systems is like conceiving a grand new sculpture

Data

* Computers can represent any information
* That can be discretized and digitized
* Algorithms process and transform data
* Massive storage capacities
* High speed networks

Programming

* Programming is referred as translating human intentions into executable algorithms
* Computer hardware is capable of only simple algorithmic steps
* Abstractions in a programming language allow humans to reason and encode solutions to complex problems

Internet

* Profound impact in the way info is stored, retrieved, and shared
* Privacy and security

Impact

* Social, ethical, legal impacts including:
* Security concerns
* Issues of software ownership and liabilities
* Social impact of database tech
* Consequences of A.I.
* No “correct” answers, instead increase awareness of:
* Various stakeholders
* Alternatives
* Short term and long term consequences
* Character-based ethics
* “good behavior” is a consequence of “good character”

**Chapter 1 (Data Storage)**

Bits and Bit Patterns

* Bit: binary digit (0 or 1)
* Bit patterns are used to represent info
* Numbers, text characters, images, sound, and others

Boolean Operations

* An operation that manipulates one or more true/false values
* 0 = false, 1 = true
* George Boole’s logic
* AND (1 AND 1 = 1), OR (1 OR 0 = 1), XOR (1 XOR 1 = 0), NOT (1=0, 0=1)

Gates

* Gate: a device that computes a Boolean operation
* Often implemented as (small) electronic circuits
* Provide the building blocks from which computers are constructed
* VLSI (very large scale integration): allows millions of electrical components to be constructed on a wafer (5)

Flip-flops

* Flip-flop: a circuit built from gates that gates that can store one bit
* One input line is used to set its stored value to 1
* One input line is used to set its stored value to 0
* While both input lines are 0, the most recently stored value is preserved

Hexadecimal Notation

* A shorthand notation for long bit patterns
* Divides a pattern into groups of 4 bits each
* Represents each group by a single symbol
* Ex: 10100011 becomes A3
* Stream: long string of bits

Main Memory Cells

* Cell: unit of main memory (8 bits = 1 byte)
* Most significant bit: the bit at left (high-order) end
* Least significant bit: the bit at right (low-order) end

Main Memory Addresses

* Address: A “name” that uniquely identifies 1 cell in the computer’s main memory
* The names are numbers
* Numbers assigned consecutively starting at 0
* Numbering the cells in this manner associates an order with the memory cells
* RAM (Random Access memory): memory in which individual cells can be easily accessed in any order
* DRAM (Dynamic Memory): RAM composed of volatile memory

Measuring Memory Capacity

* Kilobyte = 2^10 bytes
* Megabyte = 2^20 bytes
* Gigabyte = 2^30 bytes

Mass Storage

* Additional memory devices: magnetic disks, CDs, DVDs, magnetic tape, flash drives, solid-state disks
* Advantages over main memory
* Less volatility, larger storage capacities, low cost, in many cases can be removed

Magnetic Systems (text)

* Magnetic disk: thin disk used to hold data
* Track: each head transverses a circle
* Cylinder: each reposition of read/write heads; a new set of tracks
* Sector: each track divided into small arcs where into is recorded
* Zoned-bit recording: outer edge tracks have more sectors than the center
* Disk system performance:

1. Seek time: time for read/write heads to move 1 track to another
2. Rotation delay (latency time): half time for disk to make a rotation
3. Access time: sum of seek time & rotation delay
4. Transfer rate: rate at which data can be transferred to or from disk

* Magnetic tape: info recorded on magnetic coating
* Floppy disk drives: single platters with a magnetic coating in a cartridge

Optical Systems

* Compact disk (CD): info recorded by creating variations in their reflective surfaces; info then retrieved by laser that detects irregularities on the surface
* CD-DA (digital audio): CD applied to audio recordings
* DVDs (digital versatile disks): multiple, semi-transparent layers that serve as distinct surfaces when viewed by a towed laser
* BDs (Blu-rays): provides 5x the DVD capacity

Flash Drives

* Flash memory: circuits that traps electrons in tiny silicon dioxide chambers
* Repeated erasing slowly damages the media
* Mass storage of choice for: digital cameras & smartphones
* SD (secure digital) Cards provide GBs of storage
* Flash drives: for general mass storage applications
* SSDs (solid-state disks): larger flash memory drives designed to take place of magnetic hard disks
* Wear-leveling: reduces impact of SSD sectors from limited lifetime
* Sd memory cards: provide up to 2GBs
* SDHC (High Capacity): up to 32GBs
* SDXC (Extended Capacity) mem cards: exceed a TB

Representing Text

* Each character is assigned a unique bit pattern
* ASCII: uses patterns of 7-bits to represent most symbols used in written English text
* ISO (Internat. Org for Standardization) developed a number of 8 bit extensions to ASCII, each designed to accommodate a major language group
* Unicode: uses patterns up to 21-bits to represent the symbols used in languages world wide, 16-bits for world’s commonly used languages

Representing Numeric Values

* Binary notation: uses bits to represent a number in base two
* Limitations of computer representations of numeric values
* Overflow: occurs when a value is too big to be represented
* Truncation: occurs when a value cannot be represented accurately
* Computer-aided design (CAD): 3D objects are displayed to manipulated on computer display screens

Representing Images

* Bit map techniques
* Pixel: short for “picture element”
* RGB
* Luminance and chrominance
* Vector techniques
* Scalable
* TrueType and PostScript

Representing Sound

* Sampling techniques
* Used for high quality recordings
* Records actual audio
* MIDI
* Used in music synthesizers
* Records “musical score”

The Binary System

* Traditional decimal system is based on powers of ten
* Binary system is based on powers of two
* (0+0=0), (1+0=1), (0+1=1), (1+1=10)

Storing Integers

* Two’s complement notation: the most popular means of representing integer values
* Excess notation: another means of representing integer values
* Both can suffer from overflow errors

Storing Fractions

* Floating-point Notation: consists of a sign bit, a mantissa field, and an exponent field
* Related topics include normalized form & truncation errors

Data & Programming

* A programming language is a computer system created to allow humans to precisely express algorithms using a higher level of abstraction

Getting Started with Python

* Python: a popular programming language for applications, scientific computation, and as an introductory language for students

Variables

* Variables: name values for later use
* Analogous to mathematic variables in algebra

Debugging – (syntax errors, semantic errors, runtime errors)

Data Compression

* Lossy vs lossless
* Run-length encoding
* Frequency-dependent encoding (Huffman codes)
* Relative encoding
* Dictionary encoding (includes adaptative dictionary encoding such as LZW encoding.)

Compressing Images

* GIF: good for cartoons
* JPEG: good for photographs
* TIFF: good for image archiving

Compressing Audio and Video

* MPEG: high def tv broadcast & video conferencing
* MP3: temporal masking & frequency masking

Communication Errors

* Parity bits (even vs odd)
* Checkbytes
* Error correcting codes

**Chapter 2**

Computer Architecture

* CPU (central processing unit): circuitry in a computer that controls the manipulation of data
* Arithmetic/logic unit vs control unit
* Registers: general purpose and special purpose
* Motherboard: machine’s main circuit board
* Bus: collection of wires that the CPU extracts data from supplying the address of the pertinent memory cell along with an electronic signal
* Mobile internet devices (MID)
* Microprocessors: very small CPUs

CPU Basics

1. Arithmetric/logic unit: circuitry that performs operations on data
2. Control unit: circuitry for coordinating the machine’s activities
3. Register unit: data storage cells (registers)

Stored Program Concept

* Cache memory: portion of high-speed memory located within the CPU itself
* Stored-program concept: idea of storing a computer’s program in its main memory
* A program can be encoded as bit patterns and stored in main memory. From there, the CPU can then extract the instructions and execute them. In turn, the program to be executed can be altered easily.
* Machine instruction: an instruction (or command) encoded as a bit pattern recognizable by the CPU
* Machine language: the set of all instructions recognized by a machine

Machine Language Philosophies

* RISC (reduced instruction set computing): few, simple, efficient, and fast instructions
* Ex: PowerPC from Apple/IBM/Motorola and ARM
* CISC (complex instruction set computing): many, convenient, and powerful instructions
* Ex: intel

Machine Instruction Types

* Data transfer: copy data from one location to another
* Arithmetic/logic: use existing bit patterns to compute a new bit patterns
* Control: direct the execution of the program
* I/O instructions: input/output activities of the machine

Parts of a machine Instruction

* Op-code: specifies which operation to execute
* Operand: gives more detailed info about the operation
* Interpretation of operand varies depending on op-code

Program Execution

* Controlled by two special-purpose registers
* Program counter: address of next instruction
* Instruction register: current instruction
* Machine cycle
* Fetch, decode, execute

Arithmetic/Logic Operations

* Logic: AND, OR, XOR (masking)
* Rotate and shift: circular shift, logical shift, arithmetic shift
* Arithmetic: add, subtract, multiply, divide
* Precise action depends on how the values are encoded (two’s complement vs floating-point)
* Bitwise operations: combine 2 strings of bits to produce a single output string by applying the basic operation to individual columns
* Masking: mask determines which part of the other operand will affect the result

Rotation & Shift operations

* Circular shift, logical shift, and arithmetic shifts (14)

Communicating with Other Devices

* Controller: an intermediary apparatus that handles communication between the computer and a device
* Specialized controllers for each type of device
* General purpose controllers (USB and FireWire)
* Port: the point at which a device connects to a computer
* Memory-mapped I/O: CPU communicates with peripheral devices as though they were memory cells
* Direct memory access (DMA): main memory access by a controller over the bus
* Von Neumann Bottleneck: Insufficent bus speed impedes performance
* Handshaking: the process coordinating the transfer of data between components
* Parallel communication: several communication paths transfer bits simultaneously
* Serial communication: bits are transferred one after the other over a single communication path

Programming Data Manipulation

* Programming languages shields users from details of the machine:
* A single Python statement might map to one, tens, or hundreds of machine instructions
* Programmer does not need to know if the processor is RISC or CISC
* Assigning variables surely involves LOAD, STORE, and MOVE op-codes

Functions

* Function: a name for a series of operations that should be performed on the given parameter or parameters
* Function call: appearance of a function in an expression or statement
* Argument value: a value plugged into a parameter
* Fruitful functions return a value
* Void functions, or procedures, don’t return a value]

Other Architectures

* Techs to increase throughput:
* Pipelining: overlap steps of the machine cycle
* Parallel processing: use multiple processors simultaneously
* SISD: no parallel processing
* MIMD: different programs, different data
* SIMD: same program, different data

Chapter 3 (Operating Systems)

Functions of Operating systems

* Oversee operation of computer
* Store and retrieve files
* Schedule programs for execution
* Coordinate the execution of programs

Evolution of shared computing

* Batch processing: execution of jobs by collecting them in a single batch, then executing them
* Interactive processing: requires real-time processing (execute tasks under a deadline) (15)
* Time-sharing/multitasking: implemented by multiprogramming, provided service to multiple users at the same time (multitasking: one user executing numerous tasks simultaneously)
* Multiprogramming: time intervals that execute each job restricted to only one interval at a time
* Multiprocessor machines

Types of software

* Application software: performs specific tasks for users
* System software: provides infrastructure for application software & consists of operating system and utility software

Operating System Components

* User interface: communicates with users (text based (Shell) & graphical user interface (GUI))
* Kernel: performs basic required functions (file manager, device drivers, memory manager, scheduler and dispatcher)

File Manager

* Directory (folder): user-created bundle of files and other directories (subdirectories)
* Directory path: sequence of directories within directories

Memory manager

* Allocates space in main memory
* May create the illusion that the machine has more memory than it actually does (virtual memory) by playing a “shell game” in which blocks of data (pages) are shifted back and forth between main memory and mass storage

Bootstrapping

* Boot loader: program in ROM (ex in firmware)
* Run by the cpu when power is turned on
* Transfers operating system from mass storage to main memory
* Executes jump to operating system

Processes  
- process: the activity of executing a program

* Process state: current status of the activity (program counter, general purpose registers, related portion of main memory)

Process Administration

* Scheduler: adds new processes to the process table and removes completed processes from the process table
* Dispatcher: controls the allocation of time slices to the processes in the process table
* The end of a time slice is signaled by an interrupt

Handling Competition for Resources

* Semaphore: a “control flag”
* Critical region: a group of instructions that should be executed by only one process at a time
* Mutual exclusion: requirement for proper implementation of a critical region

Deadlock

* Processes block each other from continuing
* Conditions required for deadlock

1. Competition for non-sharable resources
2. Resources requested on a partial basis
3. An allocated resource can not be forcibly retrieved

Security

* Attacks from outside (problems: insecure passwords & sniffing software, counter measures: auditing software)
* Attacks from within (problem: unruly processes, counter measurers: control process activities via privileged modes and privileged instructions)