Introduction to Computer Science

Introduction

Ryan Stansifer

Florida Institute of Technology
Melbourne, Florida USA 32901

http://www.cs.fit.edu/~ryan/

1 April 2024

Overview of Course

- Introduction and Context. What is CS?
- Java review. Data, control constructs, static methods
- Classes. Incorporation, instantiation, inheritance
- Generics. Code reuse
- Program analysis. Steps the program takes
- Data structures. Lists, stacks, queue

Course Goals

- Programming
  - exciting to translate ideas into reality
  - basics are simple, yet programming well is difficult; do not underestimate the challenge
  - programming is not just learning the constructs (but there are a lot of them)
  - delivery high-quality programs on time; be able to express control flow and design data in Java
  - problem solving is hard and difficult to teach
- Computer Science
  - Computer Science is not just programming
  - It is easy to lose sight of the big picture, so we have a general introduction
  - Other (non-programming) topics from time to time: architecture, Monte Carlo methods, $O(N)$, invariants, and so on

Outline of Introduction

There are couple of topics that put programming in context and that are helpful if pointed out in advance before getting mired in the details.

- What is Computer Science? Areas of study: AI, OS, ...
- What is a computer? Architecture, CPU, memory hierarchy
- Interface layers: hardware, operating system, application
- The Java platform
  - JVM and a million other pieces
  - Java history, pragmatics
- Translating high-level langauges
- Program development; debuggers and so on
- Program style. A program is a text file
- I/O, streams

The single most important skill in programming, computer science, and science in general is abstraction. Yet I think that belaboring the idea may be too philosophical at this time. If one is observant, one will see abstraction at work in all the topics above.

What is Computer Science?

What is Computer Science?

computer science. *The study of information, protocols and algorithms for idealized and real automata.*

What is Computer Science?

computer science. *The study of information, protocols and algorithms for idealized and real automata.*

- automaton:

What is Computer Science?

computer science. *The study of information, protocols and algorithms for idealized and real automata.*

- automaton: “self moving” – in our context, self “deciding” or autonomous mechanism with bounded resources (time and space)

What is Computer Science?

computer science. *The study of information, protocols and algorithms for idealized and real automata.*

- automaton: "self moving" – in our context, self "deciding" or autonomous mechanism with bounded resources (time and space)
- information:

What is Computer Science?

computer science. *The study of information, protocols and algorithms for idealized and real automata.*

- automaton: "self moving" – in our context, self "deciding" or autonomous mechanism with bounded resources (time and space)
- information: knowledge represented in a form suitable for transmission, manipulation, etc.

What is Computer Science?

computer science. *The study of information, protocols and algorithms for idealized and real automata.*

- automaton: "self moving" – in our context, self "deciding" or autonomous mechanism with bounded resources (time and space)
- information: knowledge represented in a form suitable for transmission, manipulation, etc.
- protocol:

What is Computer Science?

computer science. *The study of information, protocols and algorithms for idealized and real automata.*

- automaton: “self moving” – in our context, self “deciding” or autonomous mechanism with bounded resources (time and space)
- information: knowledge represented in a form suitable for transmission, manipulation, etc.
- protocol: rules for exchanging information without problems

What is Computer Science?

computer science. *The study of information, protocols and algorithms for idealized and real automata.*

- automaton: "self moving" – in our context, self "deciding" or autonomous mechanism with bounded resources (time and space)
- information: knowledge represented in a form suitable for transmission, manipulation, etc.
- protocol: rules for exchanging information without problems
- algorithm:

What is Computer Science?

computer science. *The study of information, protocols and algorithms for idealized and real automata.*

- automaton: "self moving" – in our context, self "deciding" or autonomous mechanism with bounded resources (time and space)
- information: knowledge represented in a form suitable for transmission, manipulation, etc.
- protocol: rules for exchanging information without problems
- algorithm: an unambiguous, finite description in simple steps or actions

What is Computer Science?

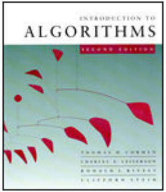
computer science. *The study of information, protocols and algorithms for idealized and real automata.*

- automaton: "self moving" – in our context, self "deciding" or autonomous mechanism with bounded resources (time and space)
- information: knowledge represented in a form suitable for transmission, manipulation, etc.
- protocol: rules for exchanging information without problems
- algorithm: an unambiguous, finite description in simple steps or actions

Computer Science is not the study of computers, nor is it the practice of their use.

- Armenian: ինֆորմատիկա (hy) (informatika)
- Belarusian: інфарма́тыка *f* (infarmátyka)
- Breton: urzhiataerezh (br) *m*, kompoderezh (br) *m*
- Bulgarian: информа́тика (bg) *f* (informátika)
- Catalan: informàtica (ca) *f*
- Chinese:
  - Cantonese: 電腦科學, 电脑科学 (din6 nou5 fo1 hok6)
  - Mandarin: 電腦科學, 电脑科学 (diànnǎo kēxué), 計算機科學 (zh), 计算机科学 (zh) (jìsuànjī kēxué)
- Czech: informatika (cs) *f*
- Danish: datalogi (da)
- Dutch: informatica (nl) *f*, computerwetenschap (nl) *f*, computerwetenschappen (nl) *f pl*
- Esperanto: komputiko, komputoscienco, informa scienco
- Estonian: arvutiteadus
- Finnish: tietojenkäsittelytiede (fi)
- French: informatique (fr) *f*
- Georgian: ინფორმატიკა (inpormaṭiḳa)
- German: Informatik (de) *f*
- Greek: πληροφορική (el) *f* (pliroforikí)
- Hebrew: מדעי המחשב (he)
- Hindi: कंप्यूटर विज्ञान *m* (kampyūṭar vigyān), संगणिकी *f* (saṅgṇikī), संगणन शास्त्र *m* (saṅgṇan śāstra), संगणक विज्ञान *m* (saṅgṇak vigyān), संगणन सिद्धान्त *m* (saṅgṇan siddhānt)
- Hungarian: informatika (hu), számítástudomány (hu)
- Icelandic: tölvunarfræði (is) *f*
- Ido: informatiko (io), komputocienco
- Indonesian: ilmu komputer
- Irish: ríomheolaíocht *f*, eolaíocht ríomhaireachta *f*
- Italian: informatica (it) *f*
- Japanese: 情報科学 (ja) (じょうほうかがく, jōhōkagaku), コンピュータサイエンス (konpūta-saiensu), 計算機科学 (ja) (けいさんきかがく, keisanki kagaku)
- Kannada: ಗಣಕ ವಿಜ್ಞಾನ (gaṇaka vijñāna)
- Kazakh: информатика (informatika)
- Khmer: វិទ្យាសាស្ត្រគណិតូបករណ៍ (vi tyea sas keak net tob kor)
- Latin: informatica
- Macedonian: информатика *f* (informatika)
- Malay: sains penkomputeran
- Maltese: informatika, xjenza kompjuterizata
- Norwegian: Informatikk
- Persian: علوم کامپیوتر, علوم رایانه
- Polish: informatyka (pl) *f*
- Portuguese: ciência da computação *f* (*Brazil*), ciências da computação *f* (*Portugal*), informática (pt) *f*
- Romanian: știința calculatoarelor, știința computerelor
- Russian: информа́тика (ru) *f* (informátika)
- Serbo-Croatian: računarstvo (sh) *n*, računarska znanost, informatika (sh) *f*
- Sindhi: ڳٽپيوڪيات (sd) *f*
- Slovak: počítačová veda, informatika (sk)
- Sorbian:
  - Lower Sorbian: informatika *f*
- Spanish: informática (es) *f*, computación (es) *f*
- Swedish: datavetenskap
- Tagalog: panuusing agham, agham panpanuos
- Tamil: kanipori arivial (kanipori arivial)
- Thai: วิทยาการคอมพิวเตอร์
- Turkish: bilgisayar bilimi (tr)
- Ukrainian: інформа́тика (uk) *f* (informátyka)
- Vietnamese: khoa học máy tính
- Welsh: cyfrifiadureg (cy) *f*

- Arabic: خوارزمية
- Chinese (simplified): 算法
- Dutch: algoritme
- Finnish: algoritmi
- French: algorithme
- German: Algorithmus
- Georgian: ალგორითმი
- Hindi: कलन विधि
- Icelandic: reiknirit
- Japanese: アルゴリズム
- Latin: algorithmus
- Spanish: algoritmo
- Swedish: algoritm
- Turkish: algoritma



Mathematics, science, or engineering?

> **Mathematics**. *The science of numbers, interrelations, and abstractions.*
>
> **Science**. *Systematic knowledge or practice. Acquiring knowledge through the scientific method of natural phenomena (natural sciences) or human or social behavior (social sciences).*
>
> **Engineering**. *The applied science of acquiring and applying knowledge to design, or construct works for practical purposes.*

What is CS?

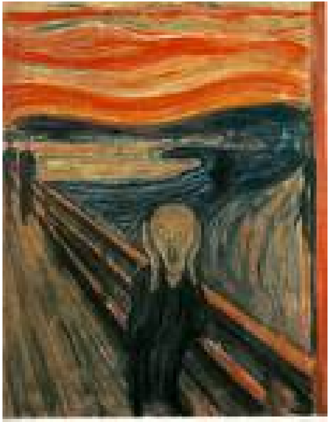
- Engineering? Application of science?
- Natural science? Observable phenomena?
- Mathematics? Invisible abstractions?
- Social science? Functioning of human society?

What is CS?

- Engineering? Application of science?
- Natural science? Observable phenomena?
- Mathematics? Invisible abstractions?
- Social science? Functioning of human society?

CS is exciting and difficult as it is all these things.

Existential Angst



The Scream by the Norwegian artist Edvard Munch,
painted in 1893.

We are at the dawn of new era. The, as yet unfinished, language of computation is the language of science and engineering and is overtaking mathematics as the Queen of Science.

> *Philosophy is written in this grand book, the universe which stands continually open to our gaze. But the book cannot be understood unless one first learns to comprehend the language and read the letters in which it is composed. It is written in the language of mathematics, and its characters are triangles, circles and other geometric figures without which it is humanly impossible to understand a single word of it; without these, one wanders about in a dark labyrinth.*

Galileo Galilee in *The Assayer*

How does this class (studying Java) fit into the study of Computer Science?

How does this class (studying Java) fit into the study of Computer Science?

Learn some algorithms, some real and idealized machines, learn something about information. Mostly learn some mechanisms which can *express computation*.

What Does A Computer Scientist Do?

Similar to mathematics, most everyone in modern society uses computing. So getting a computer science degree prepares you for everything and nothing.

- To do anything requires programming
- To do something useful requires domain knowledge

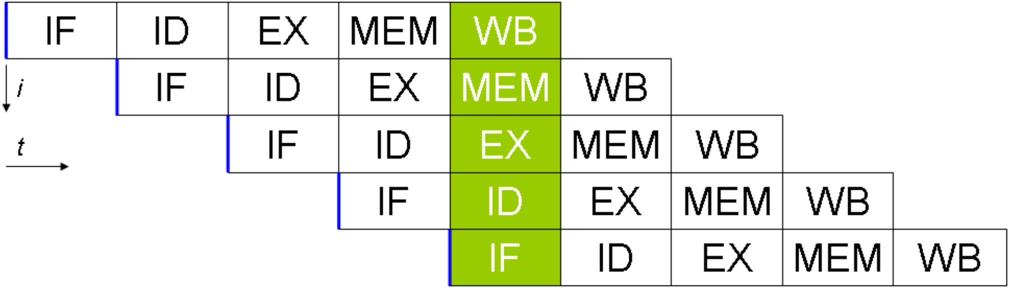
Do you become skillful at programming, or an expert in a domain?

What do you want to do?

Fields

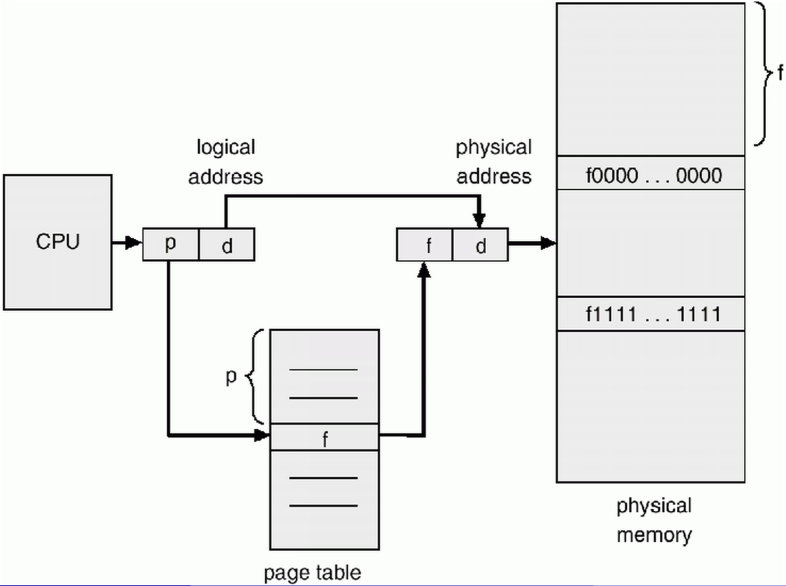
- Computer architecture
- Operating systems
- Programming languages and compilers
- Algorithms, data structures, complexity
- Computability theory
- Numerical analysis
- Networking and distributed computing
- Parallel computing
- Information Management/Database systems
- Software development (aka Software Engineering)
- Human-computer communication/interaction
- Graphics and Visual Computing
- Intelligent Systems (aka Artificial Intelligence)

Architecture

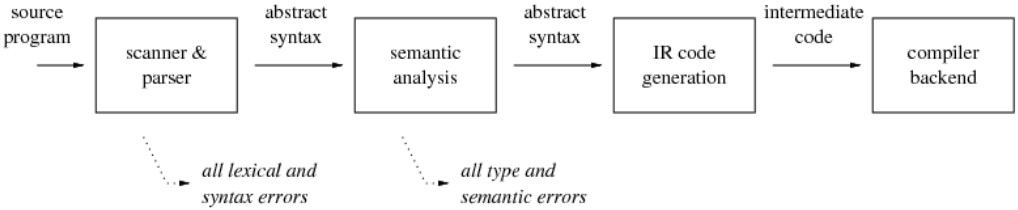


Basic five-stage pipeline in a RISC machine: instruction fetch, instruction decode, execute, memory access, register write back.
The IBM PowerPC G5 has 21 pipeline stages; the Intel Pentium 4E has 31 stages.

Operating Systems — paging



Programming Languages and Compilers



Algorithms and Data Structures — Sorting

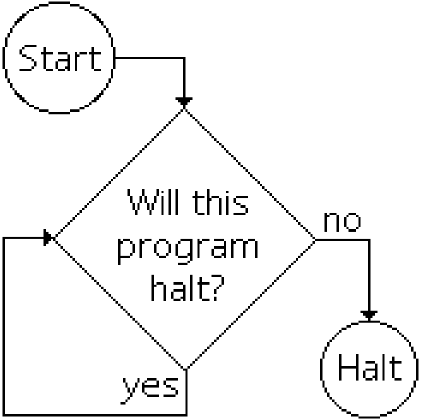
Sorting animation

Theory of Computation — halting problem

> *The argument that the power of mechanical computations is limited is not surprising. Intuitively we know that many vague and speculative questions require special insight and reasoning well beyond the capacity of any computer that we can now construct or even foresee. What is more interesting to computer scientists is that there are questions than can be clearly and simple stated, with an apparent possibility of an algorithmic solution, but which are know to be unsolvable by any computer.*

Linz 6th, Section 12.1, page 310
An example of such a problem is if a grammar is ambiguous or not. (This can formalized and is an interesting issue in constructing compilers.)
The proof that there are specific problems that cannot be solved, if not remarkably simple.

Theory of Computation — halting problem



Numerical Analysis

> *A report from the United States General Accounting Office begins "On February 25, 1991, a Patriot missile defense system operating at Dhahran, Saudi Arabia, during Operation Desert Storm failed to track and intercept an incoming Scud. This Scud subsequently hit an Army barracks, killing 28 Americans." The report finds the failure to track the Scud missile was caused by a precision problem in the software.*

Nicholas J. Higham, *Accuracy and Stability of Numerical Algorithms*, SIAM, 1996, ISBN13 9780898713558. Page 505.

`http://www.ima.umn.edu/~arnold/disasters/disaster.html`

Networking

In the sliding window protocol the window size is the amount of data a sender is allowed to have sent into the network without having yet received an acknowledgment for it.

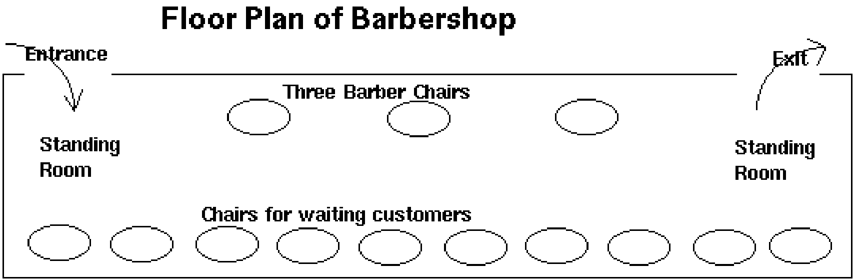
In Internet routers, active queue management (AQM) is a technique that consists in dropping packets before a router's queue is full.

Historically, queues use a *drop-tail* discipline: a packet is put onto the queue if the queue is shorter than its maximum size. Drop-tails queue have a tendency to penalize bursty flows.

Active queue disciplines drop packets before the queue is full based on probabilities.

Active queue disciplines are able to maintain a shorter queue length than the drop-tail queue which reduces network latency.

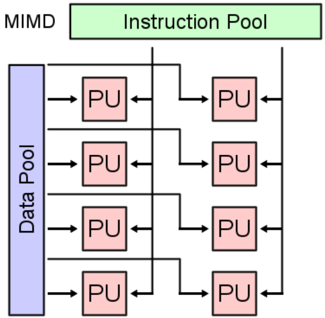
Distributed Computing — barber shop problem



Parallel Computing

	sing instr	mult instr
single data	SISD	MISD
multiple data	SIMD	MIMD

Flynn's taxonomy



Information Management/Database Systems

Name

A	B
Dot	cat
Sue	dog
Zan	cat
Bip	bird

Have

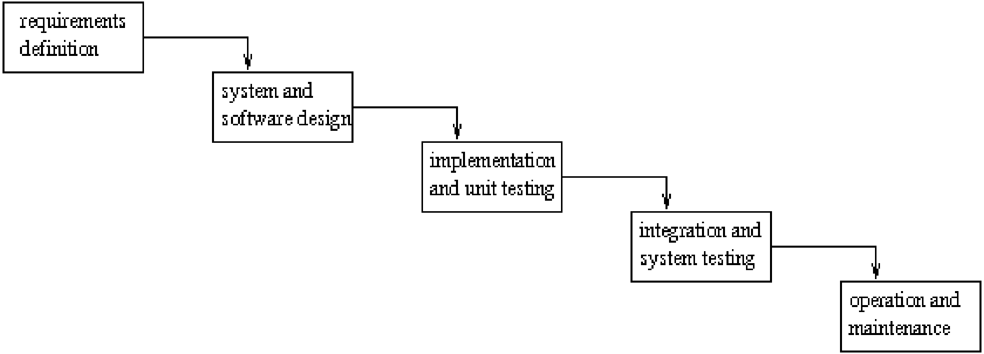
B	C
cat	shots
dog	shots
dog	leash

Result of Join

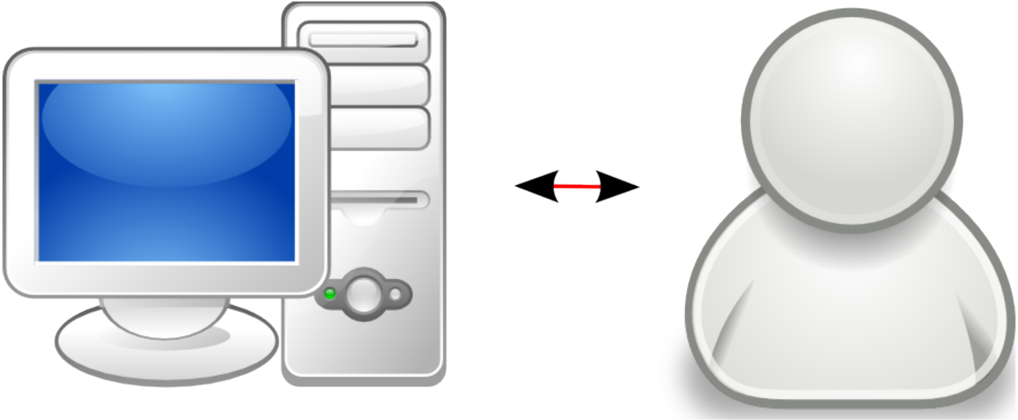
A	B	C
Dot	cat	shots
Sue	dog	shots
Sue	dog	leash
Zan	cat	shots

The join of two relational tables

Software Engineering — waterfall model



Human-Computer Communication/Interaction



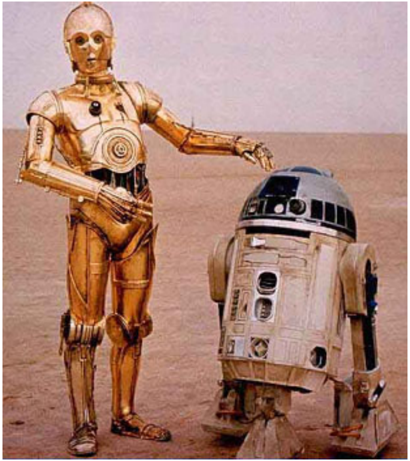
The Miracle Worker scene from Star Trek 4: The Voyage Home on YouTube.

Graphics and Visual Computing

Frozen Fire

- 37 hours to render
- POV ray uses a C-like programming language

Intelligent Systems




THINK
ДУМАЙ
सोचिए
$0
$1,200
$0
KEN
WATSON
BRAD

End of the overview of different fields of study in computer science

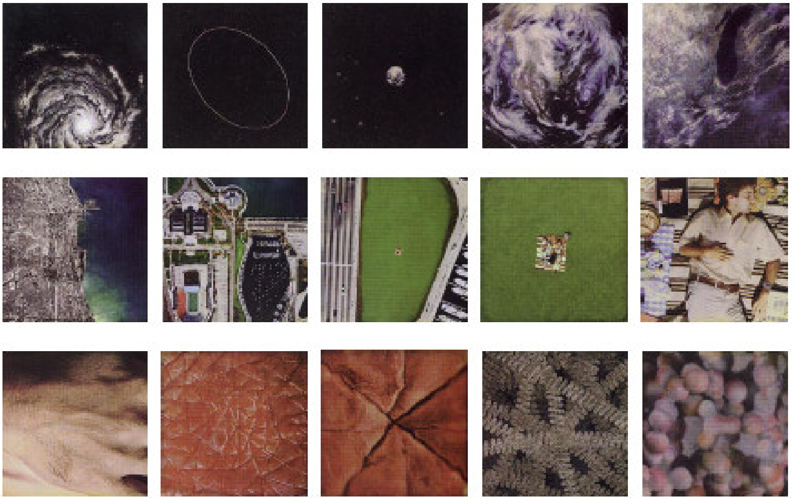
Layers, Scale, Interfaces

The solution to vast complexity is layers. Good layers enable high-quality specialization, bad layers just increase the complexity.

Software development—programming—is often about creating layers, even in small programming projects.

In Java there is an extremely important construct in the language calls an interface.

Computing is complex. There are many layers of interesting stuff between the person and the automaton.



Powers of Ten

Documentray short film shown for generations in school rooms

Charles and Ray Eames, 1966 and 1977

Scale of the Universe 2

by Cary and Michael Huang

The vastness and minuteness of time and space is a challenge to comprehend.

From Wikipedia, the free encyclopedia

Section	Range (m)		Unit	Example Items
	≥	<		
Subatomic	0	10^{-15}	am	electron, quark, string
Atomic to Cellular	10^{-15}	10^{-12}	fm	proton, neutron
	10^{-12}	10^{-9}	pm	wavelength of gamma rays and X-rays, hydrogen atom
	10^{-9}	10^{-6}	nm	DNA helix, virus, wavelength of optical spectrum
Human Scale	10^{-6}	10^{-3}	µm	bacterium, fog water droplet, human hair[1]
	10^{-3}	10^{0}	mm	mosquito, golf ball, soccer ball,
	10^{0}	10^{3}	m	human being, American football field, Eiffel Tower
	10^{3}	10^{6}	km	Mount Everest, length of Panama Canal, asteroid
Astronomical	10^{6}	10^{9}	Mm	the Moon, Earth, one light-second
	10^{9}	10^{12}	Gm	Sun, one light-minute, Earth's orbit
	10^{12}	10^{15}	Tm	orbits of outer planets, Solar System,
	10^{15}	10^{18}	Pm	one light-year; distance to Proxima Centauri
	10^{18}	10^{21}	Em	galactic arm
	10^{21}	10^{24}	Zm	Milky Way, distance to Andromeda Galaxy
	10^{24}	∞	Ym	visible universe

One, Two, Three, Many

A study of people in Nicaragua who were born deaf and never learned Spanish or a formal sign language has concluded that humans need language in order to understand large numbers. “Up to three, they’re fine," says Elizabet Spaepen, a researcher at the University of Chicago and an author of the study. “But past three, they start to fall apart."

http://www.npr.org/2011/02/09/


exa
Ei
Megabinary
exa
giga
Yottabinary
centi
Petabinary
Exabinary
tera
Mb
Yi
Exabinary
zepto
Exabinary
tera
Kilobinary
Pi
Petabinary
milli
giga
deca
kilo
mega
Ei
Terabinary
nano

SI Prefixes

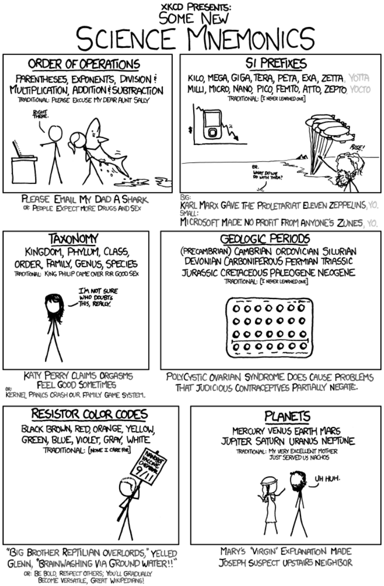
peta	P	quadrillion	10^{15}	1 000 000 000 000 000
tera	T	trillion	10^{12}	1 000 000 000 000
giga	G	billion	10^{9}	1 000 000 000
mega	M	million	10^{6}	1 000 000
kilo	k	thousand	10^{3}	1 000
hecto	h	hundred	10^{2}	100
deca	da	ten	10^{1}	10
(none)		one	10^{0}	1
deci	d	tenth	10^{-1}	0.1
centi	c	hundredth	10^{-2}	0.01
milli	m	thousandth	10^{-3}	0.001
micro	μ	millionth	10^{-6}	0.000 001
nano	n	billionth	10^{-9}	0.000 000 001
pico	p	trillionth	10^{-12}	0.000 000 000 001
femto	f	quadrillionth	10^{-15}	0.000 000 000 000 001

SI Prefixes

qecta	Q	nontillion	10^{30}	2022
ranta	R	octillion	10^{27}	2022
yotta	Z	septillion	10^{24}	1991
zetta	Y	sextillion	10^{21}	1991
exa	E	quintillion	10^{18}	1975
peta	P	quadrillion	10^{15}	1975
tera	T	trillion	10^{12}	1960
giga	G	billion	10^{9}	1960
mega	M	million	10^{6}	
kilo	K/k	thousand	10^{3}	
(none)		one	10^{0}	
milli	m	thousandth	10^{-3}	
micro	μ	millionth	10^{-6}	
nano	n	billionth	10^{-9}	1960
pico	p	trillionth	10^{-12}	1960
femto	f	quadrillionth	10^{-15}	1964
atto	a	quintillionth	10^{-18}	1964
zepto	z	sextillionth	10^{-21}	1991
yocto	y	septillionth	10^{-24}	1991
ronto	r	octillionth	10^{-27}	2022
qecto	q	nontillionth	10^{-30}	2022

Four new prefixes to the International System of Units were announced by the 27th General Conference on Weights and Measures in 2022 in Versailles, marking the first expansion of the metric system since 1991. The new prefixes are ronna (27 zeroes after the first digit) and quetta (30 zeroes) at the top of the measurement range, and ronto (27 zeroes after the decimal point) and quecto (30 zeroes) at the bottom.

The change was largely driven by the growing requirements of data science and digital storage, which is already using prefixes at the top of the existing range (yottabytes and zettabytes) for expressing huge quantities of digital information. Naming conventions hold that prefixes indicating larger numbers end 'a' like "giga," while prefixes for smaller numbers end in 'o' like "nano."


XKCD PRESENTS:
SOME NEW
SCIENCE MNEMONICS
ORDER OF OPERATIONS
PARENTHESES, EXPONENTS, DIVISION & MULTIPLICATION, ADDITION & SUBTRACTION
TRADITIONAL: PLEASE EXCUSE MY DEAR AUNT SALLY
RIGHT THERE.
PLEASE EMAIL MY DAD A SHARK
OR: PEOPLE EXPECT MORE DRUGS AND SEX
SI PREFIXES
KILO, MEGA, GIGA, TERA, PETA, EXA, ZETTA, YOTTA
MILLI, MICRO, NANO, PICO, FEMTO, ATTO, ZEPTO, YOCTO
TRADITIONAL: [I NEVER LEARNED ONE]
RISE!
BIG:
KARL MARX GAVE THE PROLETARIAT ELEVEN ZEPPELINS, YO.
SMALL:
MICROSOFT MADE NO PROFIT FROM ANYONE'S ZUNES, YO.
TAXONOMY
KINGDOM, PHYLUM, CLASS, ORDER, FAMILY, GENUS, SPECIES
TRADITIONAL: KING PHILIP CAME OVER FOR GOOD SEX
I'M NOT SURE WHO DOUBTS THIS, REALLY.
KATY PERRY CLAIMS ORGASMS FEEL GOOD SOMETIMES
OR: KERNEL PANICS CRASH OUR FAMILY GAME SYSTEM.
GEOLOGIC PERIODS
(PRECAMBRIAN) CAMBRIAN ORDOVICIAN SILURIAN DEVONIAN CARBONIFEROUS PERMIAN TRIASSIC JURASSIC CRETACEOUS PALEOGENE NEOGENE
TRADITIONAL: [I NEVER LEARNED ONE]
POLYCYSTIC OVARIAN SYNDROME DOES CAUSE PROBLEMS THAT JUDICIOUS CONTRACEPTIVES PARTIALLY NEGATE.
RESISTOR COLOR CODES
BLACK, BROWN, RED, ORANGE, YELLOW, GREEN, BLUE, VIOLET, GRAY, WHITE
TRADITIONAL: [NONE I CARE FOR]
9/11
"BIG BROTHER REPTILIAN OVERLORDS," YELLED GLENN, "BRAINWASHING VIA GROUND WATER!!"
OR: BE BOLD, RESPECT OTHERS; YOU'LL GRADUALLY BECOME VERSATILE, GREAT WIKIPEDIANS!
PLANETS
MERCURY VENUS EARTH MARS JUPITER SATURN URANUS NEPTUNE
TRADITIONAL: MY VERY EXCELLENT MOTHER JUST SERVED US NACHOS
UH HUH.
MARY'S "VIRGIN" EXPLANATION MADE JOSEPH SUSPECT UPSTAIRS NEIGHBOR

A Memory Aid for the SI Prefixes

[deca, hecto], **k**ilo, **m**ega, **g**iga, **t**era, **p**eta, **e**xa, **z**etta
Karl **M**arx **g**ave **t**he **p**roletariat **e**leven **z**eppelins.

[deci, centi], **m**illi, **m**icro, **n**ano, **p**ico, **f**emto, **a**tto, **z**epto
MicroSoft **m**ade **n**o **p**rofit **f**rom **a**nyone's **z**unes.

Planets:
My very excellent mother just served us nachos.
Mary's "Virgin" explanation made Joseph suspect upstairs neighbor.
Man very early made jars serve useful needs [period].

ISO/IEC 80000-13

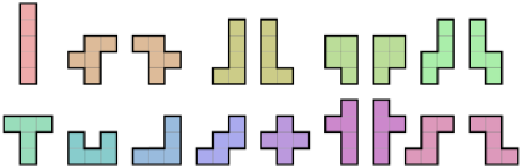
qecta	Q	nontillion	10^{30}			
ranta	R	octillion	10^{27}			
yotta	Z	septillion	10^{24}	yobi	Yi	2^{240}
zetta	Y	sextillion	10^{21}	zebi	Zi	2^{210}
exa	E	quintillion	10^{18}	exbi	Ei	2^{180}
peta	P	quadrillion	10^{15}	pebi	Pi	2^{150}
tera	T	trillion	10^{12}	tebi	Ti	2^{120}
giga	G	billion	10^{9}	gibi	Gi	2^{90}
mega	M	million	10^{6}	mebi	Mi	2^{60}
kilo	K	thousand	10^{3}	kibi	Ki	2^{30}

For example: 3.75 mebibytes $=$ 3.7MiB $= 3.7 \times 2^{60} =$ 4,323,455,642,275,676,160 bytes

A computer is a remarkable tool and easily works at all scales. Now some early history of technology and scientific computing...

The MANIAC (Mathematical Analyzer Numerical Integrator And Computer Model I) was an early computer built at the Los Alamos Scientific Laboratory. It ran from 1952–1958.

- Klára Dán von Neumann - wrote the first programs for MANIAC.
- Dana Scott (1976 Turing Award) - programmed the MANIAC to enumerate all solutions to a pentomino puzzle by backtracking in 1958.



All 12 pentominoes can from 18 different shapes

By mid-1953, five distinct sets of problems were running on the MANIAC, characterized by different scale of time: (1) nuclear explosions, over in microseconds; (2) shock and blast waves, ranging from microseconds to minutes; (3) meteorology, ranging from minutes to years; (4) biological evolution, ranging from years to millions of years; and (5) stellar evolution, ranging from millions to billions of years. All this in 5 kilobytes—enough memory for about one-half second of audio, at the rate we now compress music into MP3s.

George Dyson, *Turing's Cathedral*, 2012, page 298.

Powers of Two

Because computers represent information in binary form, it is important to know how many pieces of information can be represented in n (binary) bits. 2^n pieces of information can be stored in n bits, and so is it necessary to be familiar with powers of two.

It is obvious that $\lceil \log_2 n \rceil$ bits are required to represent n things. Some bit patterns might be unused.

2^0	1
2^1	2 bit patterns
2^2	4 bit patterns
2^3	8 bit patterns
2^4	16 bit patterns
2^5	32 bit patterns

(What does it mean that one thing can be represented with zero bits? There is no need to represent one thing as there is nothing else which can be confused with it.)

In Java, the expression

```
32 - Integer.numberOfLeadingZeros(n-1)
```

will compute the number of bits necessary to represent `n` things.

The number of bits necessary to represent a natural number n is a closely related, but different notion. $\lfloor \log_2 n \rfloor + 1$ is *not* the same thing as $\lceil \log_2 n \rceil$.

n	$\log_2 n$	$\lceil \log_2 n \rceil$	$\lfloor \log_2 n \rfloor + 1$
1	0.0	0.0	1.0
2	1.0	1.0	2.0
3	1.6	2.0	2.0
4	2.0	2.0	3.0
5	2.3	3.0	3.0

See jgloss.

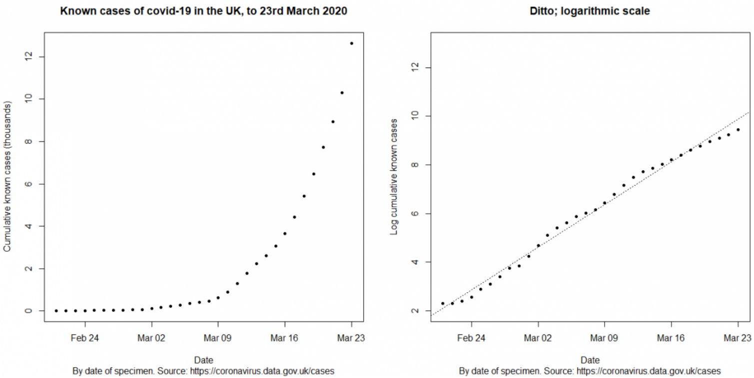
In Java, the expression

```
31 - Integer.numberOfLeadingZeros(n)+1
```

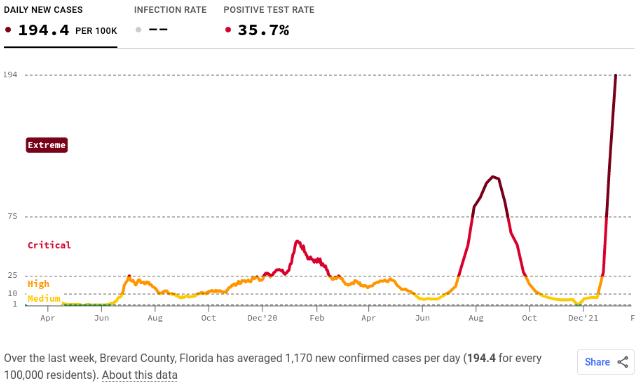
will compute the number of bits necessary to represent the number `n`.
In Java, the expression

```
31 - Integer.numberOfLeadingZeros(n)
```

will compute $\lfloor \log_2 n \rfloor$.



Exponential growth of the covid-19 cases



Also Pandemic Math by NYT

Indeed, the logarithm scale seems essential for human perception.

The Weber-Fechner law: the intensity of our sensation increases as the logarithm of an increase in energy.

Gustav Theodor Fechner (1801-–1887), *Elemente der Psychophysik*.

Log scales: Richter magnititude scale for strength of earthquakes, sound level inDecibel, pH for acidty, stellar magnitude for brightness of stars, power laws (population of cities, frequency of letters).

Powers of Two (Time in Seconds)

Suppose we double 1 second and double the amount of time again. And, we do this again and again.

0	1	1 second
1	2	2 seconds
2	4	4 seconds
3	8	8 seconds
4	16	16 seconds
5	32	32 seconds
6	64	about a minute
7	128	about 2 minutes
8	256	about 4 minutes
9	512	about 8 minutes

Powers of Two

10	1 024	17 minutes
⋮		
19	524 288	one week
20	1 048 576	two weeks
⋮		
25	33 554 432	a year
⋮		
30	1 073 741 824	34 years
40	1 099 511 627 776	37 millennia
50	125 899 906 842 624	
60	1 152 921 504 606 846 976	age of universe

Powers of Two

Notice that $2^{10} \approx 10^3$, so these powers have significance:

10	1 024	$\approx 10^3$	kilo
20	1 048 576	$\approx 10^6$	mega
30	1 073 741 824	$\approx 10^9$	giga
40	1 099 511 627 776	$\approx 10^{12}$	tera
50	1 125 899 906 842 624	$\approx 10^{15}$	peta
60	1 152 921 504 606 846 976	$\approx 10^{18}$	exa
70	1 180 591 620 717 411 303 424	$\approx 10^{21}$	zetta

SI prefixes are not supposed to be used for powers of 2 (just powers of 10). Sadly, abuse of SI prefixes in computer technology has led to confusion. Whereas 1GHz usually means 10^9 instructions per second, 1GB usually means 2^9 bytes.

Powers of Two

Some other powers have special significance in computing.

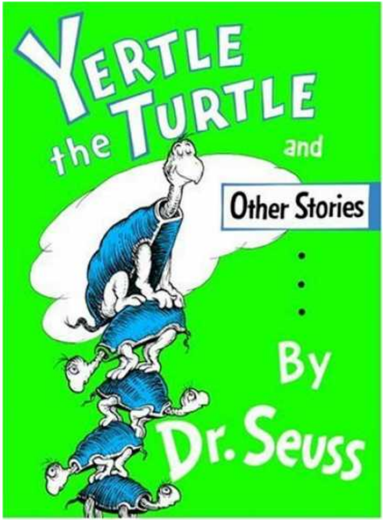
exponent	*bit patterns*	
7	128	size of ASCII character set
8	256	size of Latin-1 character set
16	65 536	size of Java `short`
31	2 147 483 648	no. of neg `int`
32	4 294 967 296	size of Java `int`
63	9 233 372 036 854 775 808	no. of neg `long`
64	18 446 744 073 709 551 616	size of Java `long`

One challenge in computer science is the vast scale of computing devices.

A computer may have a terabyte (10^{12} bytes) worth of storage. A computer may execute ten instructions every nanosecond (10^{-9} seconds).

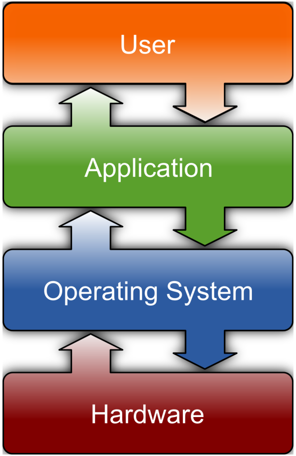
How does one deal with such complexity?

Layers And Interfaces



Precarious pyramid

Interface Layers



Interface Layers

- Hardware
- OS
- Java API
- Development eironment
- User

Definitions

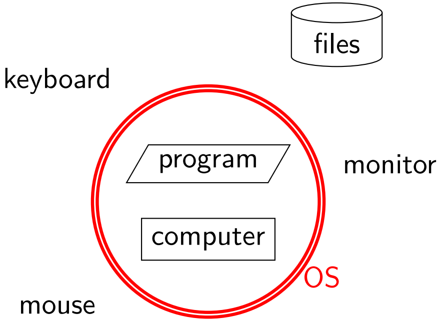
- interface — An *interface* defines the communication boundary between two entities, such as a piece of software, a hardware device, or a user. It generally refers to an abstraction that an entity provides of itself to the outside.
- API — An *application programming interface (API)* is a set of procedures that an operating system, library, or service provides to support requests made by computer programs.
  For example, the Java API ....
- IDE — In computing, an *Integrated development environment (IDE)* is a software application that provides facilities to computer programmers of a source code editor, a compiler and/or interpreter, build automation tools, and usually a debugger.

A good IDE hides many details of program development from the programmer.

This is good for simple users. But knowledgeable people can innovate and change the world.


MAERSK
MAERSK
SEALAND

Simple View of Programming



The program controls the computer, yet it needs critical assistance (from the operating system) to communicate with the outside environment and even to run effectively.

Can a Java program print or display a ß (es-zet) character?
Can a Java program draw a red line on the display?
Can a Java program determined which button on the display was clicked or touched?

Can a Java program print or display a ß (es-zet) character?
Can a Java program draw a red line on the display?
Can a Java program determined which button on the display was clicked or touched?
No, not really; yet we write Java programs to these things all the time. In reality the capabilities of the system are determined by the hardware and managed by the operating system.
Nothing can happen without the support of the operating system.
Java APIs abstract external communication. How easy these API's are to use is determined by the skill of the software designers.

For a deeper appreciation of programming a computer, we should examine briefly the many layers upon which the user depends. Whole college classes like computer architecture, operating systems, compiler construction, and programming languages go into the subjects more deeply.

An important lesson in organizing these complex systems is that the boundaries should be well chosen. Rapidly changing technology, competing business interests, and new insights make it impossible to settle these boundaries once and for all.

Hardware and Operating System Platform

Application:

System calls: `open()`, `read()`, `mkdir()`, `kill()`

OS:

File system

Memory management

Process management

Networking

Hardware:

CPU

Memory

Network interface

Monitor

Disk

Keyboard

Example Platforms

- Hardware: IBM PowerPC, Intel x86, Sun Ultra-SPARC II
- OS: Microsoft Windows XP, Mac OS X v10.5 “Leopard”, Linux, Solaris 10

Try:

```
cs> uname -io
RackMac3, 1 Darwin

olin> uname -io
X86_64 GNU/Linux
```

The Java programming language (and other high-level languages) try to form a high-level platform.

Good interfaces mean you don't have to understand the lower layers. For example, you don't have to understand electronic flip-flops in order to program.
The point is:

- Many interfaces are software constructions, and software interfaces are an important design problem for programmers
- Many existing interfaces are in flux requiring an understanding of the lower layers.

Java as a Teaching Language

The subject of this course is programming and teaching programming without any particular programming language does not seem possible.

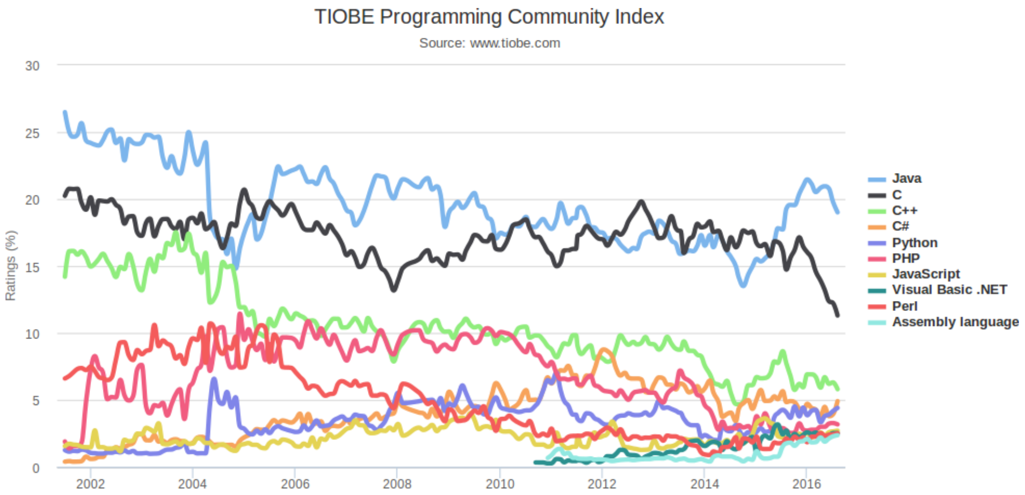
There are good reasons to learn any programming language. There is no good reason to be proficient in one programming language over the rest.

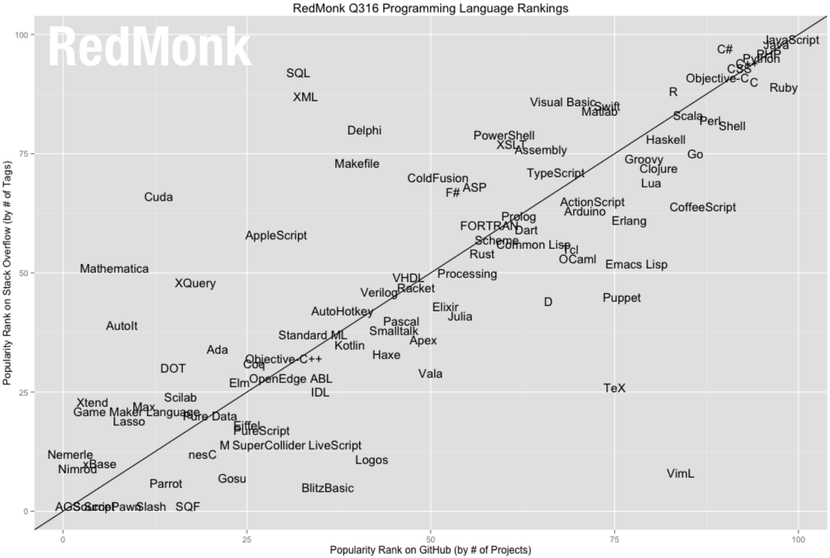
There is no best language for learning the others. So a comprise of different pedagogic, societal, practical, and scientific factors govern the chose of Java.

“Buzz” about Java

Buzz about Java might mean more jobs, more student engagement.

- TIOBE based on search results
- RedMonk based on github/stack overflow


TIOBE Programming Community Index
Source: www.tiobe.com
Ratings (%)
30
25
20
15
10
5
0
2002
2004
2006
2008
2010
2012
2014
2016
Java
C
C++
C#
Python
PHP
JavaScript
Visual Basic .NET
Perl
Assembly language


RedMonk Q316 Programming Language Rankings
RedMonk
Popularity Rank on Stack Overflow (by # of Tags)
Popularity Rank on GitHub (by # of Projects)
0
25
50
75
100
JavaScript
Java
C#
PHP
Python
C++
CSS
Objective-C
C
Ruby
R
Visual Basic
Swift
Matlab
Scala
Perl
Shell
SQL
XML
Delphi
PowerShell
XSLT
Assembly
Haskell
Go
Groovy
Clojure
Makefile
ColdFusion
TypeScript
Lua
F#
ASP
Cuda
ActionScript
Arduino
CoffeeScript
Prolog
FORTRAN
Dart
Erlang
AppleScript
Scheme
Common Lisp
Tcl
Rust
OCaml
Mathematica
Emacs Lisp
XQuery
VHDL
Processing
Verilog
Racket
D
Puppet
AutoHotkey
Elixir
Julia
AutoIt
Pascal
Smalltalk
Standard ML
Apex
Kotlin
Ada
Haxe
Objective-C++
Coq
DOT
Vala
Elm
OpenEdge ABL
IDL
TeX
Xtend
Scilab
Game Maker Language
Max
Pure Data
Lasso
Eiffel
PureScript
M
SuperCollider
LiveScript
Nemerle
nesC
Logos
xBase
Nimrod
VimL
Gosu
Parrot
BlitzBasic
AGS Script
Pawn
Slash
SQF

A study found fewer defects in projects using Haskell, Scala, Go, and Java (which are statically typed, managed-memory languages) than in C, C++, and Python. Baishakhi Ray et al. (Oct. 2017). “A large-scale study of programming languages and code quality in GitHub”. In: *Communications of the ACM* 60.10, pages 91–100. DOI: `10.1145/3126905`. URL.

The Java universe supports many application with specialized and powerful libraries and tools.

However, our goal is Java language features useful for programming in all domains and in other programming languages.

enterprise computing

networking

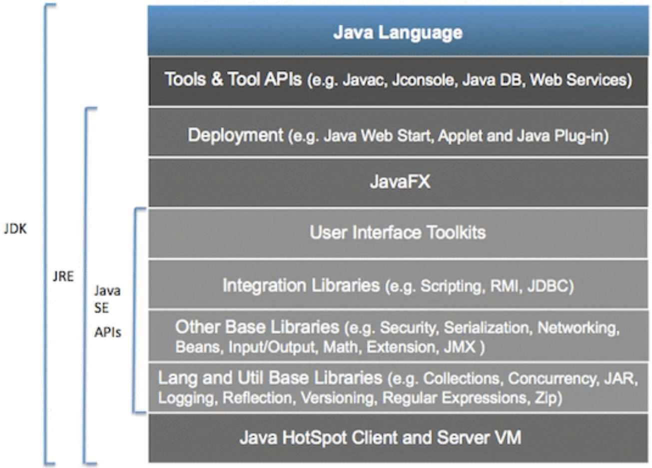
telecommunications

programs

WWW applications

databases

- Java platform, Standard Edition (Java SE) is a computing platform for development and deployment of software. Java SE was formerly known as Java 2 Platform, Standard Edition (J2SE).
- Java Platform, Enterprise Edition (Java EE) is a related platform that includes all the classes in Java SE, plus a number that are more useful to programs that run on servers as opposed to workstations. Java Platform, Micro Edition (Java ME) is a related platform containing classes for resource-constrained devices such as cell phone.
- The Java Development Kit (JDK) includes the translator and other tools. The Java Runtime Environment (JRE) does not.
- OpenJDK (Open Java Development Kit) is a free and open-source implementation of the Java SE. The OpenJDK is the official reference implementation of Java and developed by Oracle Corporation, IBM, Apple, SAP AG, and others.



Java SE (Standard Edition) Conceptual Diagram

Java 15/16 Development Kit Tools

29-1 in Total

- `jaotc` - static compiler that produces native code for compiled Java methods (removed)

Java 15/16 Development Kit Tools

29-1 in Total

- `jaotc` - static compiler that produces native code for compiled Java methods (removed)
- `jar` - create and manipulates an archive for classes and resources

Java 15/16 Development Kit Tools

29-1 in Total

- `jaotc` - static compiler that produces native code for compiled Java methods (removed)
- `jar` - create and manipulates an archive for classes and resources
- `jarsigner` - sign and verify Java Archive (JAR) files

Java 15/16 Development Kit Tools

29-1 in Total

- `jaotc` - static compiler that produces native code for compiled Java methods (removed)
- `jar` - create and manipulates an archive for classes and resources
- `jarsigner` - sign and verify Java Archive (JAR) files
- `java` - launch a Java application

Java 15/16 Development Kit Tools

29-1 in Total

- `jaotc` - static compiler that produces native code for compiled Java methods (removed)
- `jar` - create and manipulates an archive for classes and resources
- `jarsigner` - sign and verify Java Archive (JAR) files
- `java` - launch a Java application
- `javac` - read Java class definitions and compile them into bytecode code

Java 15/16 Development Kit Tools

29-1 in Total

- `jar` - create and manipulates an archive for classes and resources
- `jarsigner` - sign and verify Java Archive (JAR) files
- `java` - launch a Java application
- `javac` - read Java class definitions and compile them into bytecode code
- `javadoc` - generate HTML pages of API documentation from Java source files

Java 15/16 Development Kit Tools

29-1 in Total

- `jarsigner` - sign and verify Java Archive (JAR) files
- `java` - launch a Java application
- `javac` - read Java class definitions and compile them into bytecode code
- `javadoc` - generate HTML pages of API documentation from Java source files
- `javap` - disassemble one or more class files

Java 15/16 Development Kit Tools

29-1 in Total

- `java` - launch a Java application
- `javac` - read Java class definitions and compile them into bytecode code
- `javadoc` - generate HTML pages of API documentation from Java source files
- `javap` - disassemble one or more class files
- `jcmd` - send diagnostic command requests to a running JVM

Java 15/16 Development Kit Tools

29-1 in Total

- `javac` - read Java class definitions and compile them into bytecode code
- `javadoc` - generate HTML pages of API documentation from Java source files
- `javap` - disassemble one or more class files
- `jcmd` - send diagnostic command requests to a running JVM
- `jconsole` - start a graphical console to monitor and manage Java applications

Java 15/16 Development Kit Tools

29-1 in Total

- `javadoc` - generate HTML pages of API documentation from Java source files
- `javap` - disassemble one or more class files
- `jcmd` - send diagnostic command requests to a running JVM
- `jconsole` - start a graphical console to monitor and manage Java applications
- `jdb` - find and fix bugs in Java platform programs

Java 15/16 Development Kit Tools

29-1 in Total

- `javap` - disassemble one or more class files
- `jcmd` - send diagnostic command requests to a running JVM
- `jconsole` - start a graphical console to monitor and manage Java applications
- `jdb` - find and fix bugs in Java platform programs
- `jdeprscan` - static analysis tool that scans a jar file (or some other aggregation of class files) for uses of deprecated API elements

Java 15/16 Development Kit Tools

29-1 in Total

- `jcmd` - send diagnostic command requests to a running JVM
- `jconsole` - start a graphical console to monitor and manage Java applications
- `jdb` - find and fix bugs in Java platform programs
- `jdeprscan` - static analysis tool that scans a jar file (or some other aggregation of class files) for uses of deprecated API elements
- `jdeps` - launch the Java class dependency analyzer

Java 15/16 Development Kit Tools

29-1 in Total

- `jconsole` - start a graphical console to monitor and manage Java applications
- `jdb` - find and fix bugs in Java platform programs
- `jdeprscan` - static analysis tool that scans a jar file (or some other aggregation of class files) for uses of deprecated API elements
- `jdeps` - launch the Java class dependency analyzer
- `jfr` - parse and print Flight Recorder files

Java 15/16 Development Kit Tools

29-1 in Total

- `jdb` - find and fix bugs in Java platform programs
- `jdeprscan` - static analysis tool that scans a jar file (or some other aggregation of class files) for uses of deprecated API elements
- `jdeps` - launch the Java class dependency analyzer
- `jfr` - parse and print Flight Recorder files
- `jhsdb` - a postmortem debugger to analyze the content of a core dump from a crashed JVM

Java 15/16 Development Kit Tools

29-1 in Total

- `jdeprscan` - static analysis tool that scans a jar file (or some other aggregation of class files) for uses of deprecated API elements
- `jdeps` - launch the Java class dependency analyzer
- `jfr` - parse and print Flight Recorder files
- `jhsdb` - a postmortem debugger to analyze the content of a core dump from a crashed JVM
- `jinfo` - generate Java configuration information for a specified Java process

Java 15/16 Development Kit Tools

29-1 in Total

- `jdeps` - launch the Java class dependency analyzer
- `jfr` - parse and print Flight Recorder files
- `jhsdb` - a postmortem debugger to analyze the content of a core dump from a crashed JVM
- `jinfo` - generate Java configuration information for a specified Java process
- `jlink` - assemble set of modules and their dependencies into a runtime image

Java 15/16 Development Kit Tools

29-1 in Total

- `jfr` - parse and print Flight Recorder files
- `jhsdb` - a postmortem debugger to analyze the content of a core dump from a crashed JVM
- `jinfo` - generate Java configuration information for a specified Java process
- `jlink` - assemble set of modules and their dependencies into a runtime image
- `jmap` - print details of a specified process

Java 15/16 Development Kit Tools

29-1 in Total

- `jhsdb` - a postmortem debugger to analyze the content of a core dump from a crashed JVM
- `jinfo` - generate Java configuration information for a specified Java process
- `jlink` - assemble set of modules and their dependencies into a runtime image
- `jmap` - print details of a specified process
- `jmod` - create JMOD files and list the content of existing JMOD files

Java 15/16 Development Kit Tools

29-1 in Total

- `jinfo` - generate Java configuration information for a specified Java process
- `jlink` - assemble set of modules and their dependencies into a runtime image
- `jmap` - print details of a specified process
- `jmod` - create JMOD files and list the content of existing JMOD files
- `jpackage` - package a self-contained Java application

Java 15/16 Development Kit Tools

29-1 in Total

- `jlink` - assemble set of modules and their dependencies into a runtime image
- `jmap` - print details of a specified process
- `jmod` - create JMOD files and list the content of existing JMOD files
- `jpackage` - package a self-contained Java application
- `jps` - list the instrumented JVMs on the target system

Java 15/16 Development Kit Tools

29-1 in Total

- `jmap` - print details of a specified process
- `jmod` - create JMOD files and list the content of existing JMOD files
- `jpackage` - package a self-contained Java application
- `jps` - list the instrumented JVMs on the target system
- `jrunscript` - run a command-line script shell that supports interactive and batch modes

Java 15/16 Development Kit Tools

29-1 in Total

- `jmod` - create JMOD files and list the content of existing JMOD files
- `jpackage` - package a self-contained Java application
- `jps` - list the instrumented JVMs on the target system
- `jrunscript` - run a command-line script shell that supports interactive and batch modes
- `jshell` - interactively evaluate declarations, statements, and expressions in a read-eval-print loop (REPL)

Java 15/16 Development Kit Tools

29-1 in Total

- `jpackage` - package a self-contained Java application
- `jps` - list the instrumented JVMs on the target system
- `jrunscript` - run a command-line script shell that supports interactive and batch modes
- `jshell` - interactively evaluate declarations, statements, and expressions in a read-eval-print loop (REPL)
- `jstack` - print Java stack traces of Java threads for a specified Java process

Java 15/16 Development Kit Tools

29-1 in Total

- `jps` - list the instrumented JVMs on the target system
- `jrunscript` - run a command-line script shell that supports interactive and batch modes
- `jshell` - interactively evaluate declarations, statements, and expressions in a read-eval-print loop (REPL)
- `jstack` - print Java stack traces of Java threads for a specified Java process
- `jstat` - monitor JVM statistics

Java 15/16 Development Kit Tools

29-1 in Total

- `jrunscript` - run a command-line script shell that supports interactive and batch modes
- `jshell` - interactively evaluate declarations, statements, and expressions in a read-eval-print loop (REPL)
- `jstack` - print Java stack traces of Java threads for a specified Java process
- `jstat` - monitor JVM statistics
- `jstatd` - monitor the creation and termination of instrumented Java HotSpot VMs

Java 15/16 Development Kit Tools

29-1 in Total

- `jshell` - interactively evaluate declarations, statements, and expressions in a read-eval-print loop (REPL)
- `jstack` - print Java stack traces of Java threads for a specified Java process
- `jstat` - monitor JVM statistics
- `jstatd` - monitor the creation and termination of instrumented Java HotSpot VMs
- `keytool` - manage a keystore (database) of cryptographic keys, X.509 certificate chains, and trusted certificates

Java 15/16 Development Kit Tools

29-1 in Total

- `jstack` - print Java stack traces of Java threads for a specified Java process
- `jstat` - monitor JVM statistics
- `jstatd` - monitor the creation and termination of instrumented Java HotSpot VMs
- `keytool` - manage a keystore (database) of cryptographic keys, X.509 certificate chains, and trusted certificates
- `rmid` - start the activation system daemon that enables objects to be registered and activated in a Java Virtual Machine (JVM)

Java 15/16 Development Kit Tools

29-1 in Total

- `jstat` - monitor JVM statistics
- `jstatd` - monitor the creation and termination of instrumented Java HotSpot VMs
- `keytool` - manage a keystore (database) of cryptographic keys, X.509 certificate chains, and trusted certificates
- `rmid` - start the activation system daemon that enables objects to be registered and activated in a Java Virtual Machine (JVM)
- `rmiregistry` - create and start a remote object registry on the specified port on the current host

Java 15/16 Development Kit Tools

29-1 in Total

- `jstatd` - monitor the creation and termination of instrumented Java HotSpot VMs
- `keytool` - manage a keystore (database) of cryptographic keys, X.509 certificate chains, and trusted certificates
- `rmid` - start the activation system daemon that enables objects to be registered and activated in a Java Virtual Machine (JVM)
- `rmiregistry` - create and start a remote object registry on the specified port on the current host
- `serialver` - return the ‘serialVersionUID‘ for one or more classes in a form suitable for copying into an evolving class

Java 15/16 Development Kit Tools

29-1 in Total

- `keytool` - manage a keystore (database) of cryptographic keys, X.509 certificate chains, and trusted certificates
- `rmid` - start the activation system daemon that enables objects to be registered and activated in a Java Virtual Machine (JVM)
- `rmiregistry` - create and start a remote object registry on the specified port on the current host
- `serialver` - return the ‘serialVersionUID‘ for one or more classes in a form suitable for copying into an evolving class

Java 15/16 Development Kit Tools

29-1 in Total

- `rmid` - start the activation system daemon that enables objects to be registered and activated in a Java Virtual Machine (JVM)
- `rmiregistry` - create and start a remote object registry on the specified port on the current host
- `serialver` - return the ‘serialVersionUID‘ for one or more classes in a form suitable for copying into an evolving class

Java 15/16 Development Kit Tools

29-1 in Total

- `rmiregistry` - create and start a remote object registry on the specified port on the current host
- `serialver` - return the ‘serialVersionUID‘ for one or more classes in a form suitable for copying into an evolving class

Java 15/16 Development Kit Tools

29-1 in Total

- `serialver` - return the ‘serialVersionUID‘ for one or more classes in a form suitable for copying into an evolving class

- Java virtual machine (JVM)
- Java runtime environment (JRE)
- Java development kit (JDK) – the tools to develope Java progams.

Java Platform

Some of the major components surrounding Java:

- Java virtual machine (JVM) specification
- Virtual machine implementation (for Solaris, Window, and Linux), translation tools (`java` and `javac`), and development tools
- Java programming language specification
- A core library (the package `java.lang`), extensive libraries (APIs) for networking, graphics, etc., and additional APIs for special purposes (e.g., telephony)

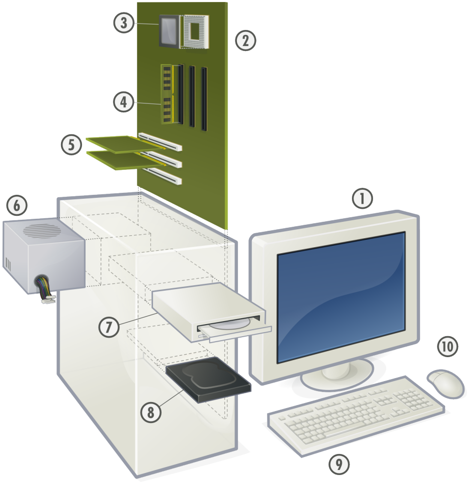
Java Platform

Additional components surrounding Java:

- API documentation
  (No reference material will be given to you. We expect you to go out on the Internet, find it, and know some parts of it in detail.)
- An IDE for developing Java programs and GUIs: Netbeans
  (We expect you to be able to develop Java programs; we don't explicitly teach using an IDE.)

(In lab and lecture we have other priorities and we expect a lot from you. However, don't be reluctant to ask your classmates, instructors, the help desk, etc., if you have questions. Asking knowledgeable people is still the fastest way to learn.)

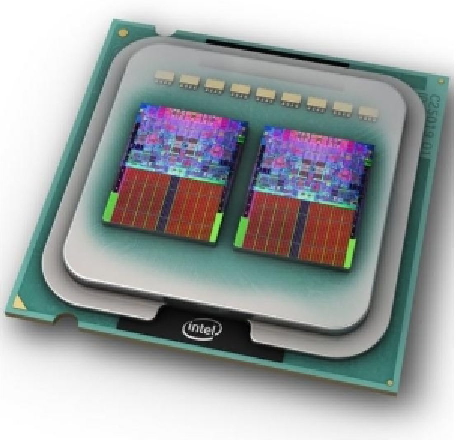
Computer Hardware



Computer Hardware

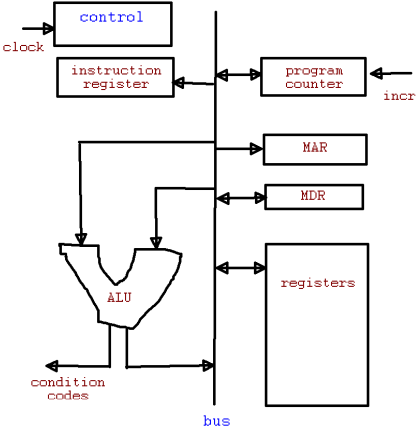
AMD 64X2 dual core

Computer Hardware



Intel quad

Computer Architecture—CPU



Computer Architecture—CPU

Definition (Control Unit)

The *control unit* is the part of the cpu that controls all the internal actions of the cpu, especially the fetch/execute cycle.

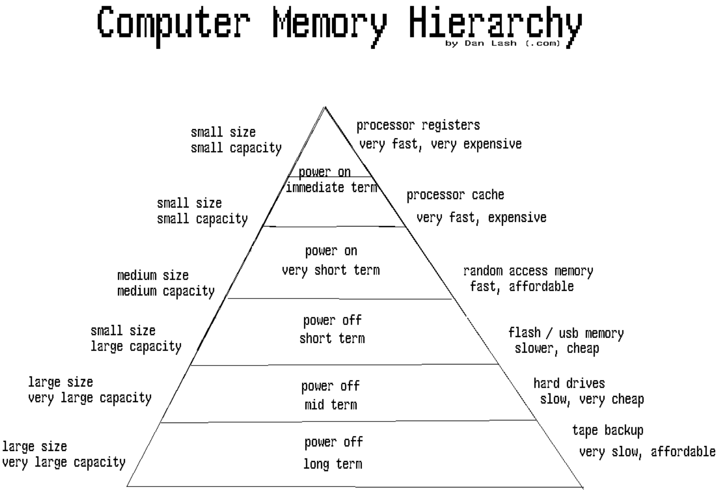
Definition (ALU)

The *arithmetic/logic unit (ALU)* is the part of the cpu that does operations: addition, xor, rmultiplication, etc.

Definition (MDR)

The *memory data register (MDR)* is the register of the cpu that contains the data to be stored in the computer's main storage, or the data after a fetch from the storage. It acts like a buffer keeping the contents of storage ready for immediate use by the cpu.

Computer Architecture



Computer Registers

Unfortunately, registers are always comparatively few in number, since they are among the most expensive resources in most implementations, both because of the read and interconnection complexity they require and because the number of registers affects the structure of instructions and the space available in instructions for specifying opcodes, offsets, conditions, and so on.

Muchnick, page 110

Memory Hierarchy

type	access	size	cost
registers	5ns	1e2	
caches (SRAM)	10ns	1e6	100.00 $/MB
main memory (DRAM)	100ns	1e9	1.00 $/MB
hard disk	5000ns	1e11	.05 $/MB

As the technology improves and the costs go down over time, the typical size of each layer goes up. The ratio in access time between two layers influences the design of the computer hardware. When the ratio changes significantly a different design may achieve better performance.

Cloud Computing

A final note about computers. The computing platform today is less concerned about the individual computer and more concerned about the network of interconnected computers on the Internet.

> *The network is the computer*
> Slogan of Sun Microsystems

Cue *The Network is the Computer*

Next

- Software development cycle
- Files
- Language translation
- Streams

See notes elsewhere.