University of Illinois at Urbana-Champaign
Dept. of Electrical and Computer Engineering

ECE 120: Introduction to Computing

Introduction and Overview

ECE120: Introduction to Computing

Lectures	M/W/F	
U. Bhowmik	9 a.m.	1015 ECEB
D. Choi	10 a.m.	1013 ECEB
K. Levchenko	1 p.m.	1013 ECEB
D. Jones	3 p.m.	2017 ECEB

Discussions Thursday

Heting Gao, Tianqi Gao, Hongliang Li, Shuijin Liu, Yanan Liu, Christopher Ryu, Jiarui (Jerry) Sun

What is ECE120?

- Teach a systems perspective that includes both hardware and software (and math!)
- ECE culture and goals
- Expectations of engineers
- Lifelong learning necessary
- Understand and identify tradeoffs
- International group—leverage it!
- Academic reality and grade philosophy

Why Start with Computers?

Why study computers first?

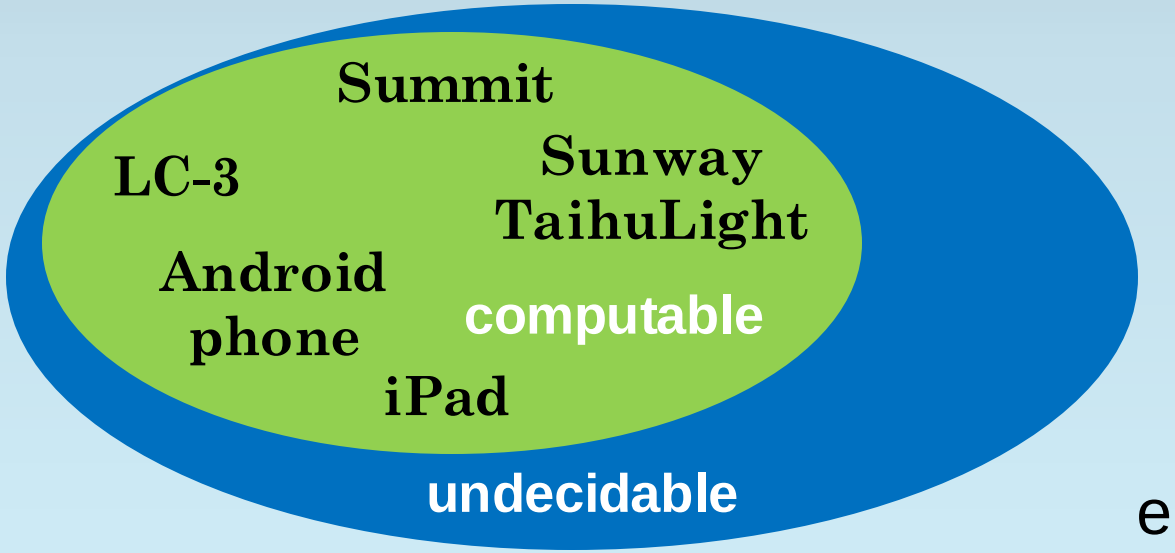
Do Aeronautical Engineers start with the high-bypass turbofan engine?

Or do they start with basic dynamics and lift?

Why not build up to computers slowly?

Computers are Universal Computation Devices

Described by Alan Turing in 1936



space of problems to solve (& examples of computers that solve each subspace)

Church-Turing Hypothesis: Computers and humans can compute the same things.

More Neurons than Transistors?

"The apparatus they [animals] use for timing their movements has more in common with an electronic computer, although it is strictly different in fundamental operation. The basic unit of biological computers, the nerve cell or neurone, is really nothing like a transistor in its internal workings. Certainly the code in which neurones communicate with each other seems to be a little bit like the pulse codes of digital computers, but the individual neurone is a much more sophisticated data-processing unit than the transistor. Instead of just three connections with other components, a single neurone may have tens of thousands. The neurone is slower than the transistor, but it has gone much further in the direction of miniaturization, a trend which has dominated the electronics industry over the past two decades. This is brought home by the fact that there are some ten thousand million neurons in a human brain: you could pack only a few hundred transistors into a skull."

--Richard Dawkins, "The Selfish Gene," Oxford University Press, New York and Oxford, 1976, p. 51

Today: Billions of Transistors

Dawkins was writing in 1976.

Moore's Law continued.

1997: Pentium released, **4.5 million** transistors

Today: **21.1 billion** transistors on **815mm^2**

(NVIDIA GV100 GPU – https://devblogs.nvidia.com/inside-volta/)

Smaller than neurons!

… still only 3 terminals

ECE Has Undergone a Digital Convergence

Many alumni, including EEs, in the industry are now computer people.

Most solutions are digital.

Digital system design provides a critical set of skills needed by nearly every ECE grad.

These skills will enable you to go further faster…

Bottom Up Approach Provides a Firm Understanding

Why do we build from the ground up?

- Helps you develop of solid understanding of the design an operation of each level.
- Easier to make effective use of abstractions and to improve those abstractions.
- Our students have been successful based on this approach (alumni feedback).

Where to Find Information

Start with the Wiki!

Remember this link:

https://wiki.illinois.edu/wiki/display/ece120

Read the Wiki Every Day

https://wiki.illinois.edu/wiki/display/ece120

What you will find includes:

- announcements from course staff
- course information and timing
- assignments, solutions, exams, and due dates
- a place for exchanging information
  - ask any non-personal questions here
  - do not post answers

What to Read (and What Not to Read)

Reading materials
- Patt & Patel, 2nd edition
- ~150 pages of notes (free online)

Read the notes (see the Wiki for which parts)
- before class
- AND after class

Look at learning objectives in notes summary sections.

Use the online tools to practice skills.

Be wary of the Web. No one has screened the content for accuracy.

Workload Includes Labs and Homework

Weekly lab assignments
- Software and hardware
- Usually due Fridays at 5 p.m.
- See assignment for specifics of how and when to turn in

FIRST LAB: due on Friday, September 6th

Weekly homework assignments
- Paper and computer-based
- Submit scanned online
  - First attempt due on Mondays at 8 p.m.
  - Corrections with explanations due the next Monday at 8 p.m.

FIRST HOMEWORK: due Monday, September 2nd (Labor Day)

Workload Also Includes Exams

Three midterms

- Midterm 1 will cover Lectures ~1-9
- Midterm 2 will cover Lectures ~10-19
- Midterm 2 will cover Lectures ~20-30

Final exam (TBD, 8am)

- Covers entire course

If you have a conflict, **let us know early**!
(specific deadlines are on the Wiki)

And Workload Includes Discussions

A question for you:

What skill least developed in many ECE grads?

The answer that many alumni and employers give: **soft skills**!

In discussion section every Thursday, you will…

- work in small groups
- solve fun problems related to lecture together
- practice working with others

How Will We Grade?

Labs	15% *
Homework	15% *
Discussions	5% *
Midterms	10%, 15%, 15%
Final	25%

* Lowest scores (of all weeks) dropped for labs, homework, and discussion sheets.

No late assignments accepted.

ECE120 Grading Scale is Absolute

90% of total points → A of some sort

80% of total points → B of some sort

70% of total points → C of some sort

(more detail on the Wiki)

Many of your classes here will be curved. But not this course!

Don't Cheat!

See **the Academic code**.

Discussion sections are done in groups.

In some labs you will have partners.

Otherwise, work must be your own.

It's ok to talk and help each other understand, but it's not ok to give/share/lend/copy/allow someone to copy answers.

Emergency response: Run > Hide > Fight

Emergencies can happen anywhere and at any time, so it's important that we take a minute to prepare for a situation in which our safety could depend on our ability to react quickly. Take a moment to learn the different ways to leave this building. If there's ever a fire alarm or something like that, you'll know how to get out and you'll be able to help others get out. Next, figure out the best place to go in case of severe weather – we'll need to go to a low-level in the middle of the building, away from windows. And finally, if there's ever someone trying to hurt us, our best option is to run out of the building. If we cannot do that safely, we'll want to hide somewhere we can't be seen, and we'll have to lock or barricade the door if possible and be as quiet as we can. We will not leave that safe area until we get an Illini-Alert confirming that it's safe to do so. If we can't run or hide, we'll fight back with whatever we can get our hands on. If you want to better prepare yourself for any of these situations, visit police.illinois.edu/safe. Remember you can sign up for emergency text messages at emergency.illinois.edu.

Run

Leaving the area quickly is the best option if it is safe to do so.

- Take time now to learn the different ways to leave your building.
- Leave personal items behind.
- Assist those who need help, but consider whether doing so puts yourself at risk.
- Alert authorities of the emergency when it is safe to do so.

Hide

When you can't or don't want to run, take shelter indoors.

- Take time now to learn different ways to seek shelter in your building.
- If severe weather is imminent, go to the nearest indoor storm refuge area.
- If someone is trying to hurt you and you can't evacuate, get to a place where you can't be seen, lock or barricade your area if possible, silence your phone, don't make any noise and don't come out until you receive an Illini-Alert indicating it is safe to do so.

Fight

As a last resort, you may need to fight to increase your chances of survival.

- Think about what kind of common items are in your area which you can use to defend yourself.
- Team up with others to fight if the situation allows.
- Mentally prepare yourself – you may be in a fight for your life.

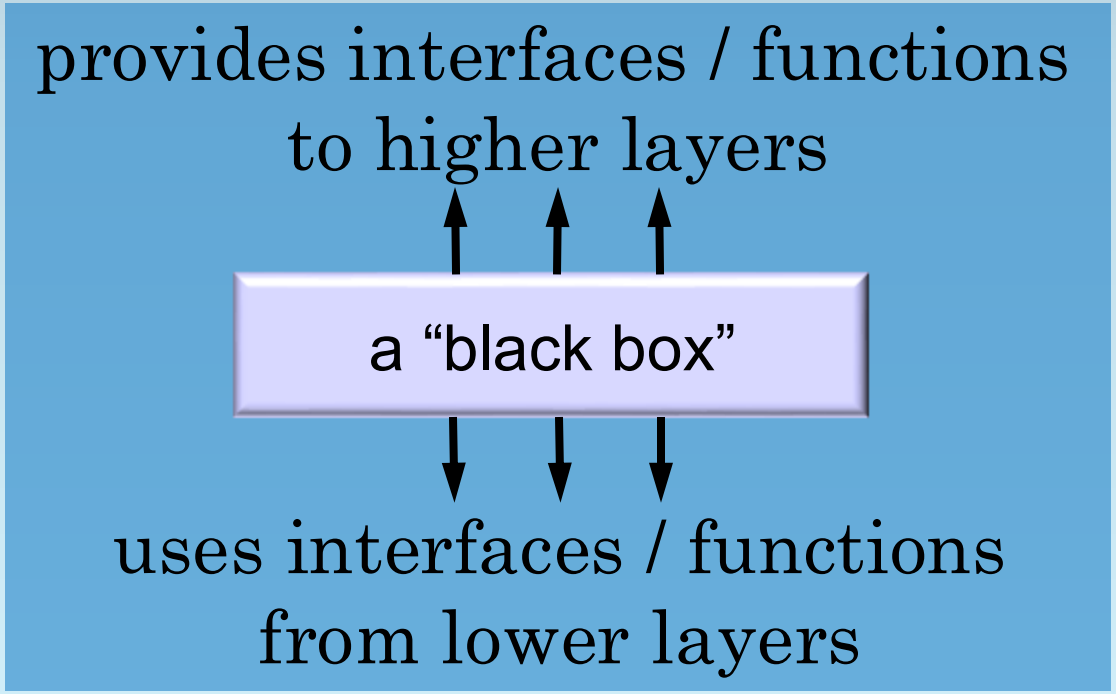
University of Illinois at Urbana-Champaign
Dept. of Electrical and Computer Engineering

ECE 120: Introduction to Computing

Abstraction Layers in Digital Systems

Abstraction Separates Function from Implementation

An abstraction layer…



Many implementations are possible!

Humans Learn to Use Many Abstractions

Example: taxi

- function: take customer to a human-specified location
- lower layers: car / van / truck / limousine / motorcycle, driver / autonomous control!

Example: water faucet

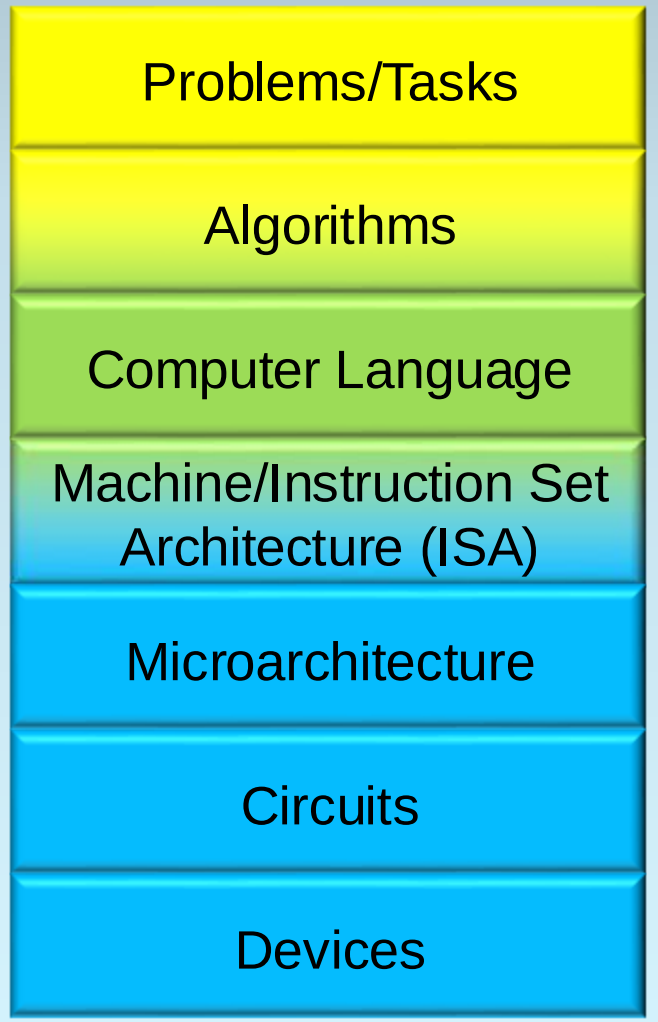
- function: get water at a specific (fuzzy) rate
- lower layers: plumbing, water tanks / cisterns / wells / aquaducts, valves, knobs

Digital Systems are Comprised of Seven Layers

The colors indicate the typical basis for each layer

- **human language / theory**
- **software**
- **digital hardware**

(figure based on Patt & Patel Ch. 1)

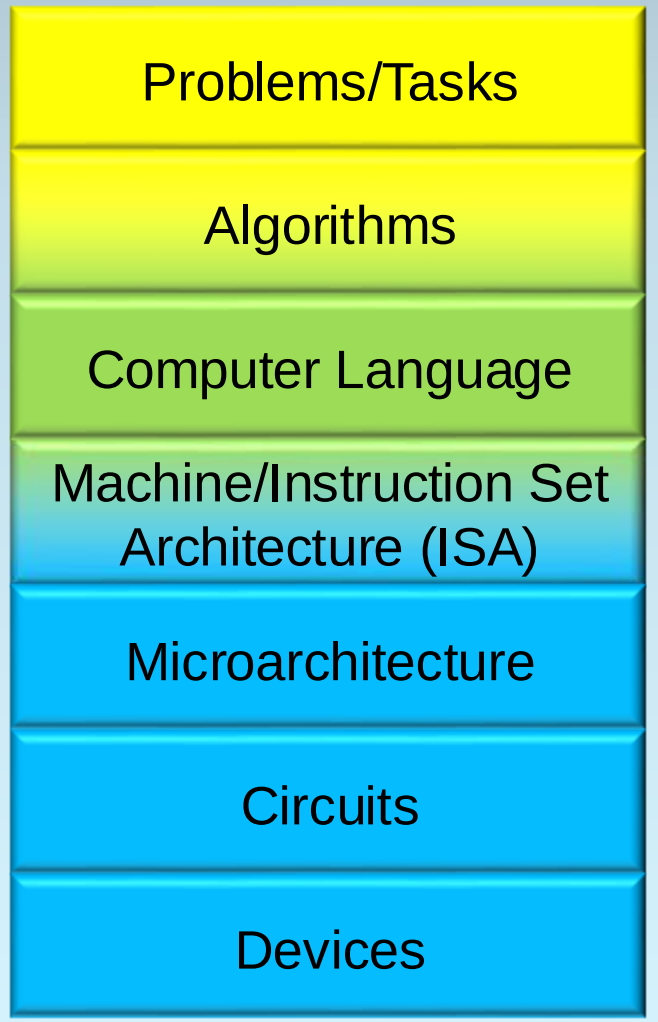


Don't Talk to Electrons. Please.

Below the device layer are the electrons.

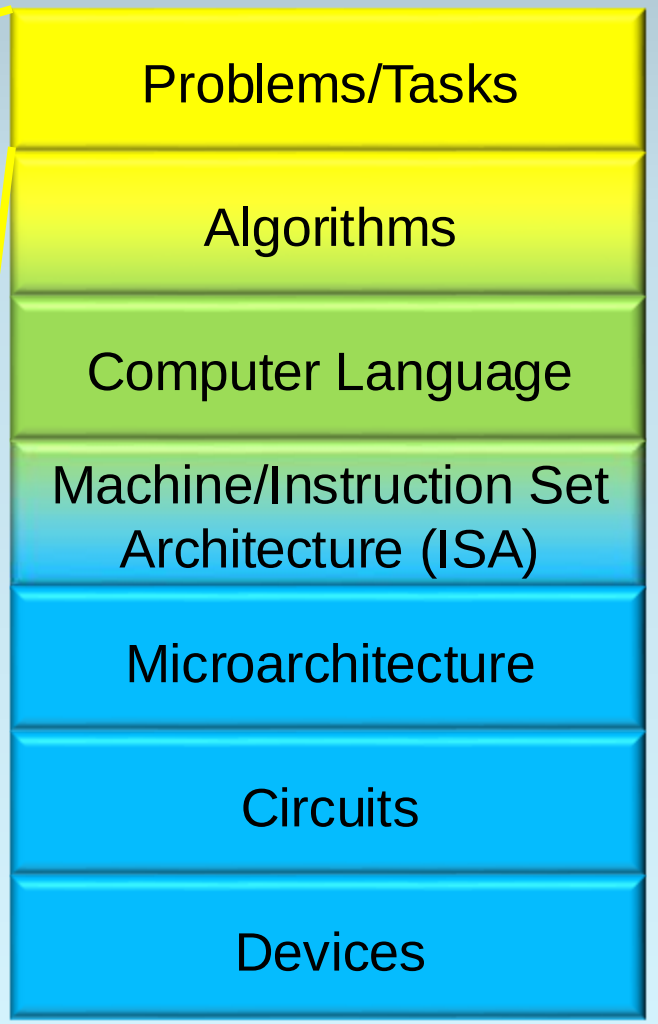
We'd like to just tell them what we want done.

But they don't seem to listen.



Human Problem Descriptions are the Top Layer

Problems/Tasks
- stated in natural (human) language
- For example: What's the sum of numbers between 1 and 3?



Sorry, But Your Answer is Wrong

Question:
What's the sum of numbers between 1 and 3?

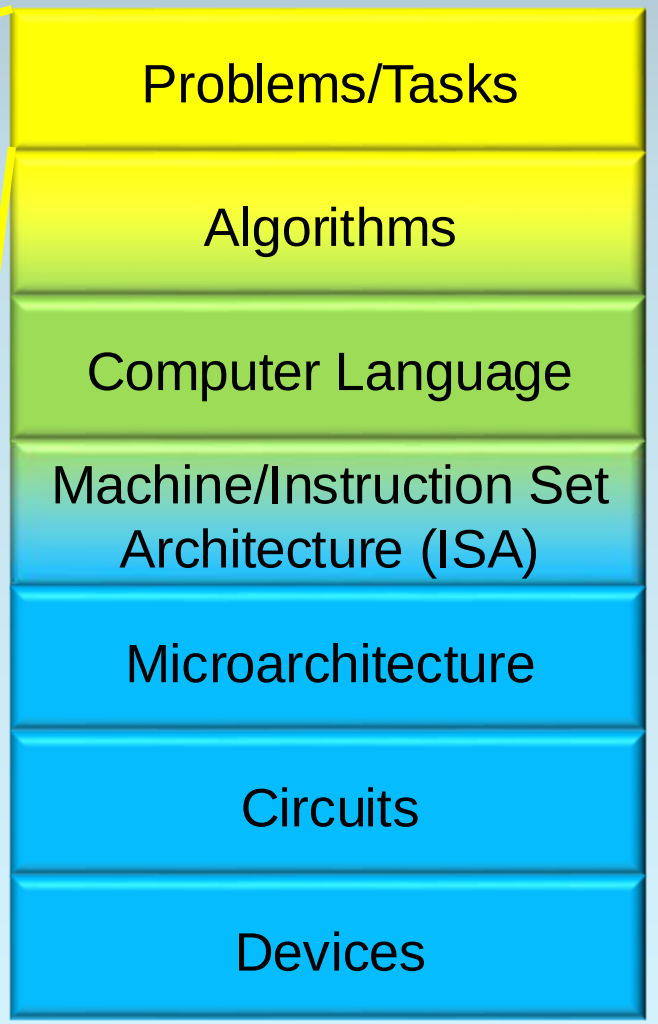
Did you answer 6? (Did you include 1 and 3?)
- What's between the bread in a peanut butter sandwich? **Is the bread between the bread?**

Did you answer 2? (Did you exclude 1 and 3?)
- What about **2.5**? What about **Π/2**? **e**?

Did you answer infinity?
- You're still wrong! (Too geeky! You'll probably end up as a professor one day.)

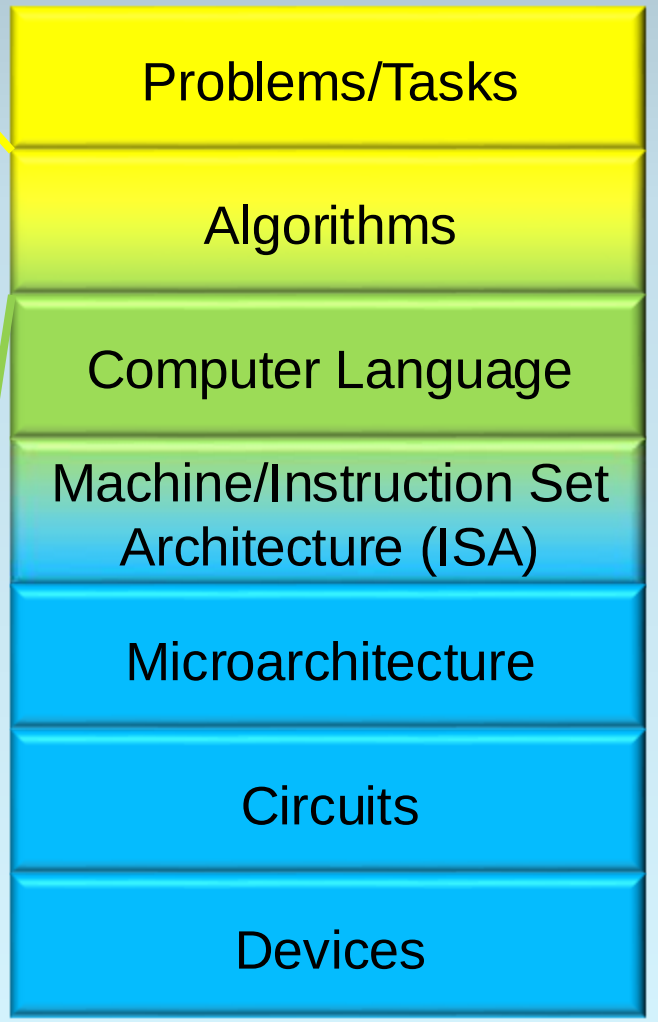
Human Languages Suffer from Ambiguity

Problems/Tasks
- stated in natural (human) language
- For example: What's the sum of numbers between 1 and 3?
- **Problem inherent to natural language: ambiguity.**
- Another example: Time flies like an arrow.



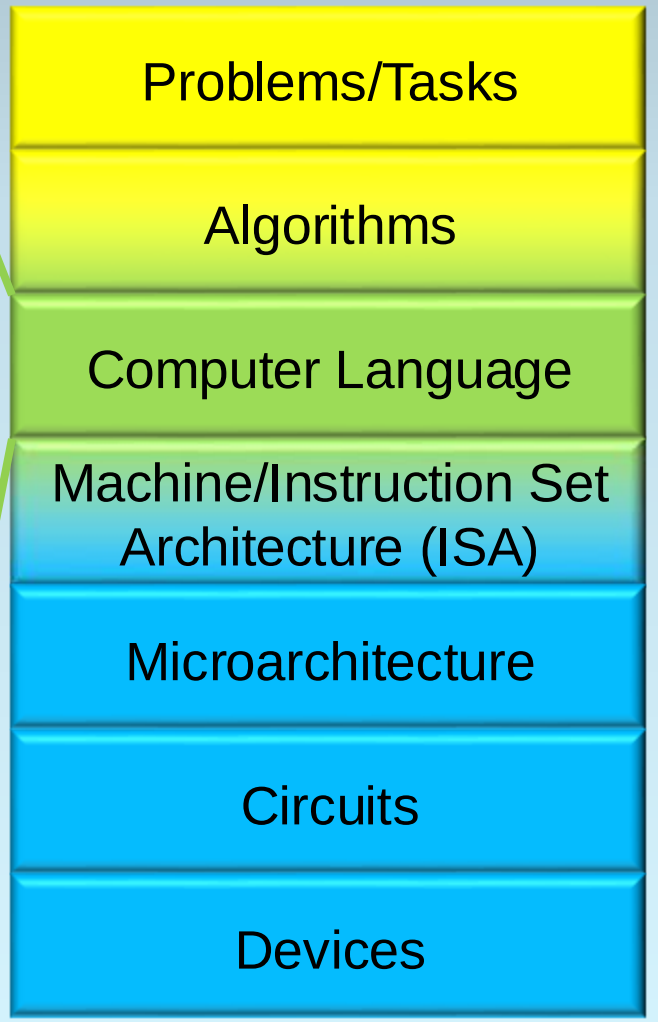
A Task Can be Solved by Many Algorithms

Algorithms
- a step-by-step process to solve a problem
- requires three things:
  - **definiteness** (no ambiguity)
  - **effective computability** (each step simple enough for a computer)
  - **finiteness** (finishes)



An Algorithm Can be Implemented in Many Languages

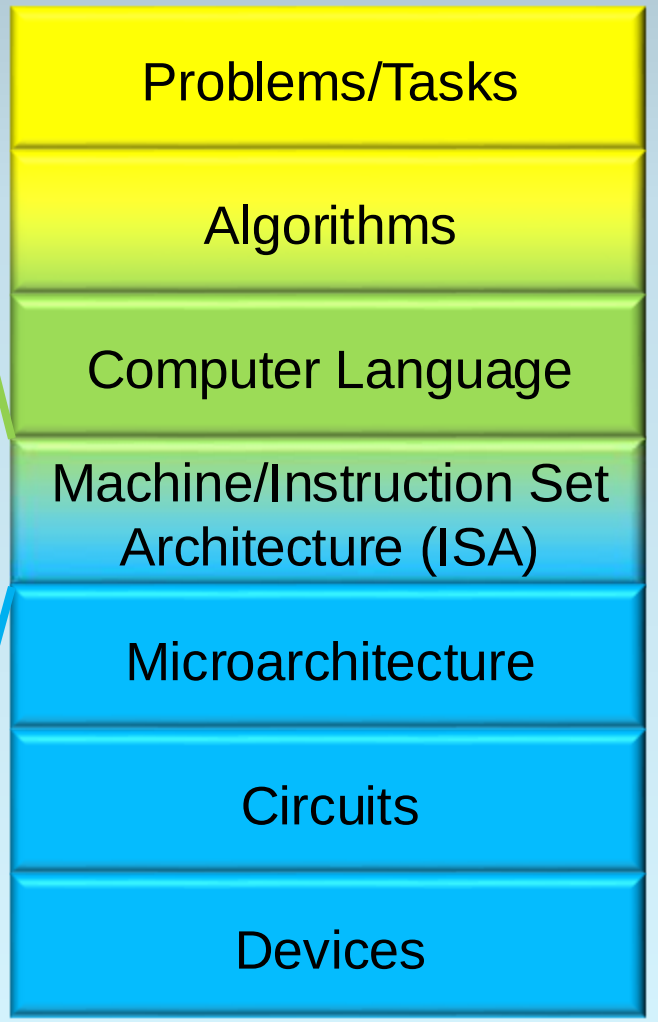
Computer Language
- 1000s of choices
- examples: C, C++, Java, Python
- we use C in 120 & 220
  - easy mapping to lower levels
  - a subset of other languages



A Language Can be Implemented with Many ISAs

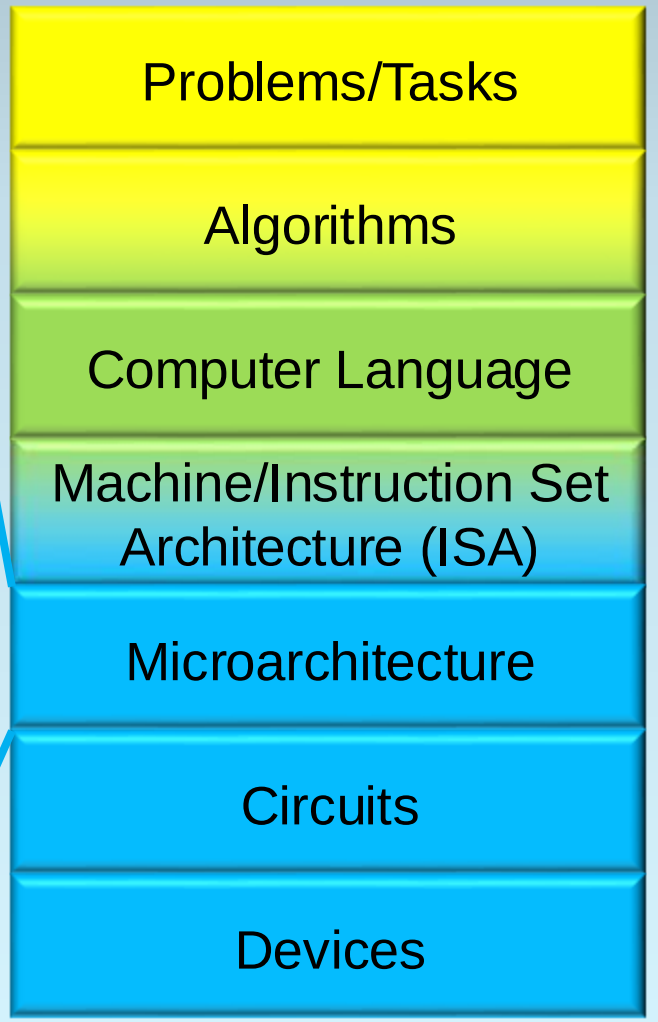
Machine/Instruction Set Architecture (ISA)

- interface between software and hardware
- examples: x86, ARM, PowerPC



An ISA Can be Executed by Many Microarchitectures

Microarchitecture
- digital hardware
- executes instructions from an ISA
- examples
  - X86 ISA: i5, i7, Opteron, Phenom
  - ARM: Cortex A15, Cortex A9, Kynetis K



Our Class Builds from the Ground Up

Your future work!

Future classes (CS374)

Week #3 (plus ECE220)

ECE120 covers these layers from the ground up (more on circuits in ECE110/210; more on devices in later years).