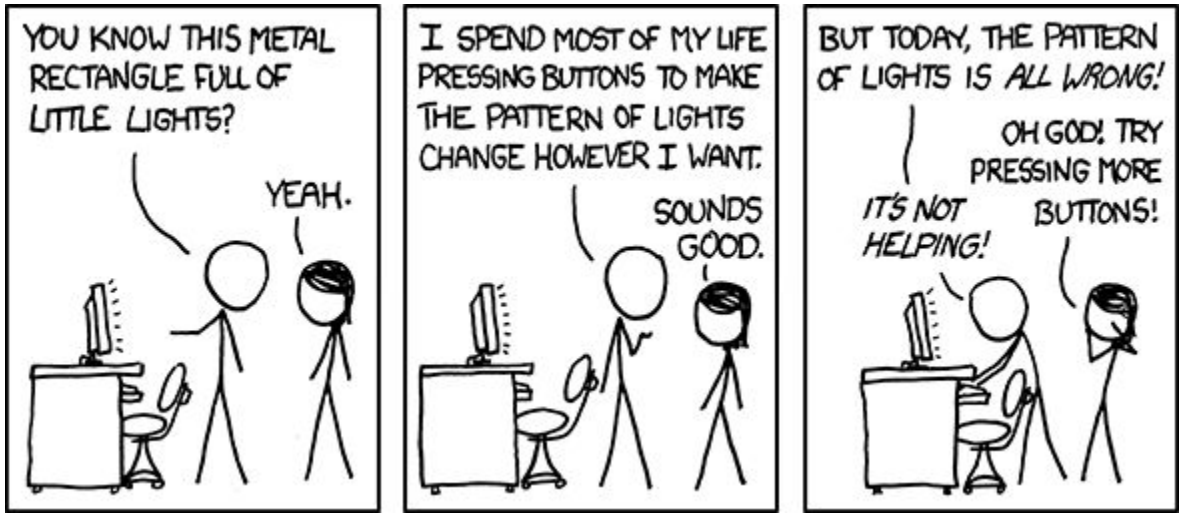
Welcome!


YOU KNOW THIS METAL RECTANGLE FULL OF LITTLE LIGHTS?
YEAH.
I SPEND MOST OF MY LIFE PRESSING BUTTONS TO MAKE THE PATTERN OF LIGHTS CHANGE HOWEVER I WANT.
SOUNDS GOOD.
BUT TODAY, THE PATTERN OF LIGHTS IS ALL WRONG!
IT'S NOT HELPING!
OH GOD! TRY PRESSING MORE BUTTONS!

Today’s agenda

1) Get introduced to the course
2) Get introduced to each other
3) Bird’s eye view of what we’re going to learn in this workshop, specifically:
    a) **Hardware:** today we’ll look at a computer’s physical components, and
    b) **Software:** the different levels of software that interact with the components.

About the course:

Let’s refer to the course website!

www.computing-workshop.com

Administrative Biz: Register

If you want a certificate and guaranteed spot!

Getting to know each other

Fill out the icebreaker activity sheet to get to know each other a bit!

Disassembling a computer

Using the disassembly worksheet:

1) Identify and **remove** each component in your desktop
2) *Using the internet to help you*, write a brief description of the functionality of each component.

when your group is done with a particular component, please drop it off in the appropriate basket at the front!

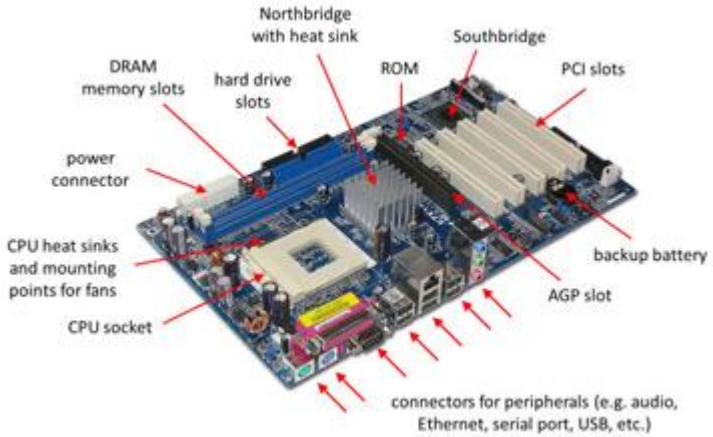
Reviewing each component, volunteers?

Component
Motherboard
Memory (RAM)
Microprocessor
Hard Drive
Power supply
CD ROM
Case Fan
CPU Fan

Motherboard (mobo)

https://www.youtube.com/watch?v=kwdQhv6WOfM

https://www.youtube.com/watch?v=yJVNfDa0s_A

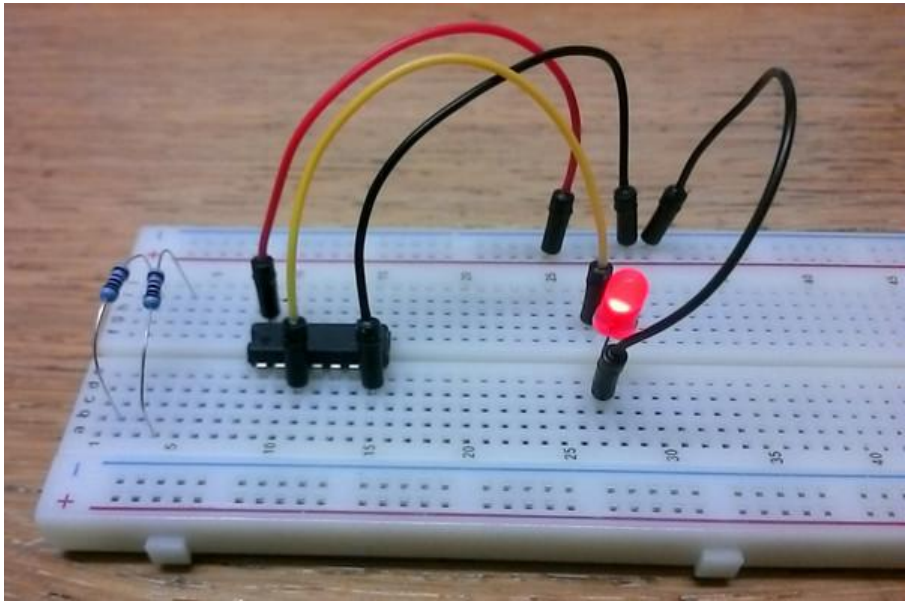


Microchips (memory and processor)

Integrated circuits with various basic electronic components.

These chips can store data (memory) or do things (processor)

Over the next few lessons we will be making our own processors and memory using breadboards!! This is exciting!!

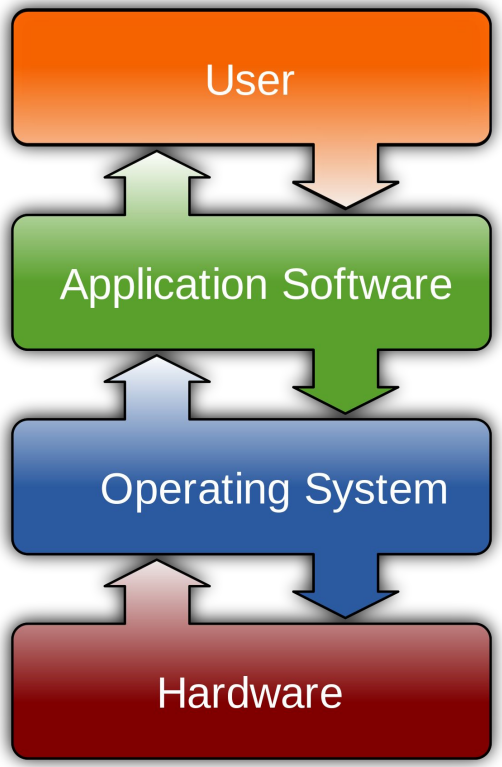


Using these components for the greater good!

But how do we even use these components?

With software!

Software is the instructions that manipulate the hardware.

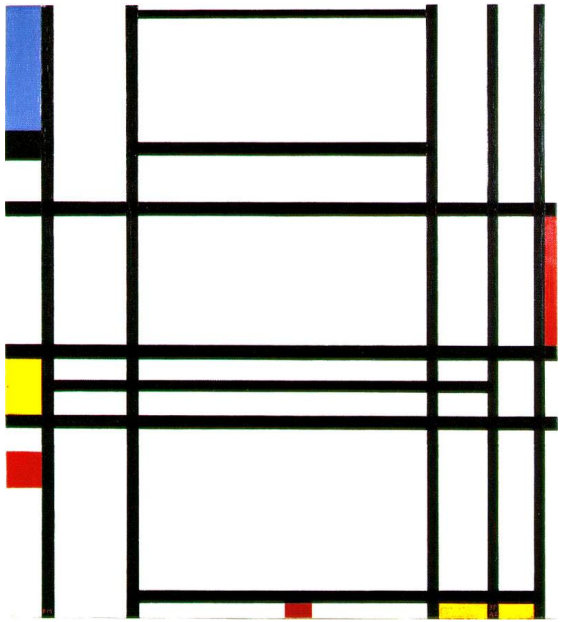


Abstraction

These instructions that manipulate the hardware come in many varieties and can be built upon each other using abstraction.

Abstraction is giving a name to a function/thing. It packages up complexity.

Let’s run an example. Volunteers?



Power of abstraction

There are 2 ways abstraction is very powerful.

1) It allows things to be done much easier.

Imagine if to draw a square, I had to first define what a rectangle, lines, intersections, and angles are each time. Very tedious!

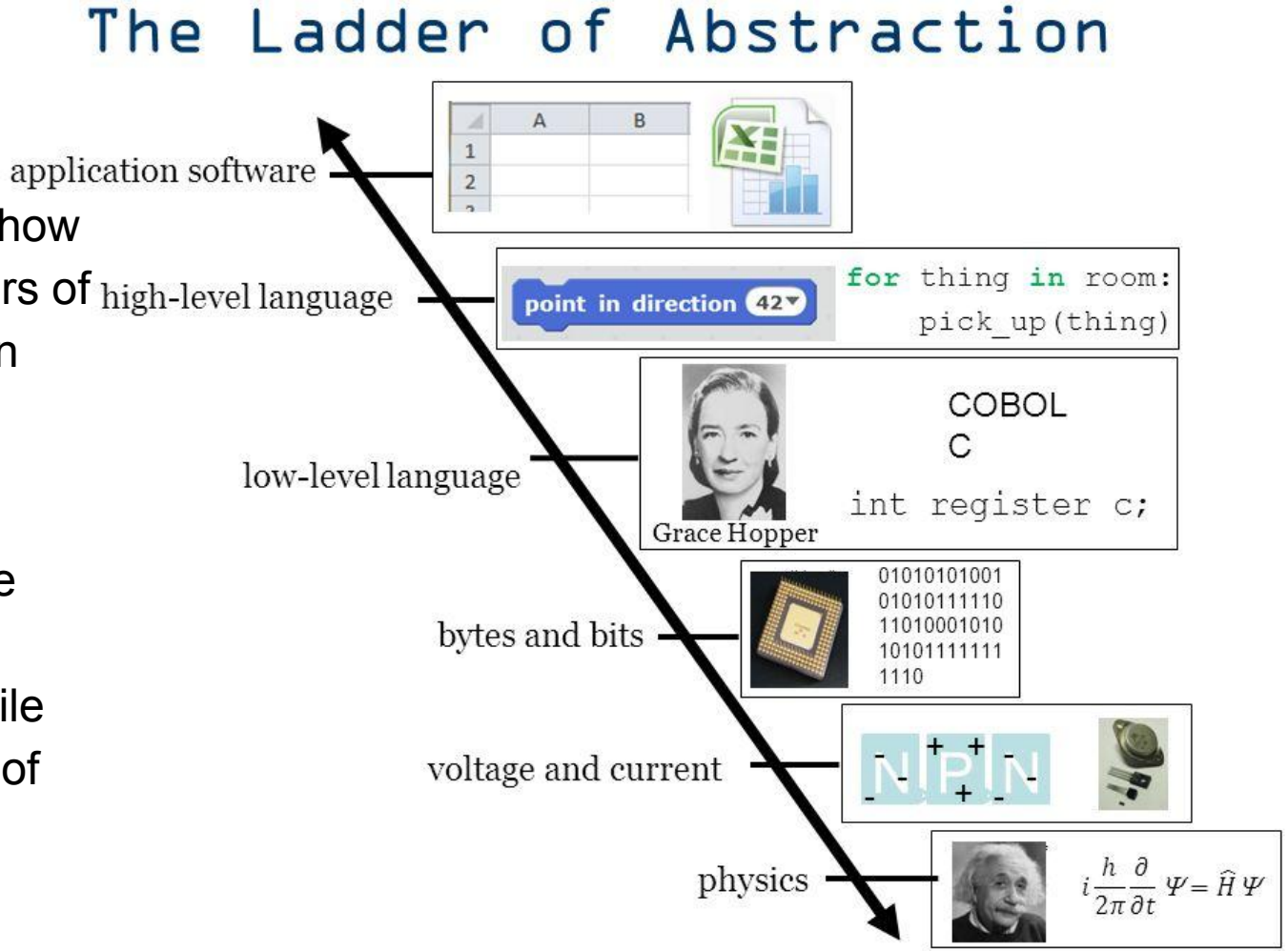
2) They are composable.

You can compose 2 or more abstractions to form larger abstractions.

Bringing it back

The analogy explains how there are multiple layers of instructions, built up on each other using abstraction.

In this workshop, we're going to focus on high level programming while building up the ladder of abstraction.



Let’s look more closely at each layer

Each group will be assigned one of the following levels of abstraction:

1) Application
2) High level programing language
3) Low level programing language (Ex. C)
4) Assembly code
5) Binary (machine language)
6) Current and voltage circuit

Your group will be expected to explain your assigned level of abstraction to the rest of the participants.

Binary & encodings/representations

https://www.youtube.com/watch?v=1GSjbWt0c9M?t=58s

Recap

Today we learned about the following things:

- Each other
- Physical components of a computer
- How software and hardware are linked through abstraction

Next week we will:

- Starting from the bottom of the ladder of abstraction: binary and circuits
- Starting at the top of the ladder of abstraction: take our first foray into coding in Haskell!

**remember to register if you want a certificate and a spot!*