Unit 1 Review

15-110 – Friday 09/25

Agenda

- Tie it all together
- Review **program evaluation** rules
- Review **half adder** vs **full adder** vs **n-bit adder**
- Review **boolean expression** vs **circuit** vs **truth table**
- Review **strings**

Unit 1 Goals

Our first unit had two major themes: developing key **programming skills**, and understanding the basics of **computer organization**.

How do the topics we discussed fit into these themes?

Programming Skills

We started with **programming basics**. A program is an implementation of an **algorithm**.

Data and **variables** are the core part of any program. **Strings** are a particularly advanced data type that contains special **operations** and **methods**.

While programming, we'll sometimes run into **errors**. We learned the basic **error types**, discovered **what causes them**, and discussed **debugging**.

Programming Skills (continued)

We also use **control structures** to change how we move through the steps of a program. **Nesting** control structures let us create more complex algorithms.

Conditionals let us choose whether or not to run a series of steps.

Loops (either **while loops** or **for loops**) let us repeat actions, as long as we define the **loop control variable**.

Functions let us store an algorithm under a name, as long as we clearly define the **inputs, returned value,** and **side effects**, and pay attention to variable **scope**.

Computer Organization

We also discussed the basics of how **computers organize data**. This is done partially through the process of **abstraction** to change levels of detail in systems.

We discussed how the computer **tokenizes, parses,** and **translates** Python into **bytecode**.

We also discussed how **variable and constant tables** and **value and call stacks** help the computer keep track of information.

Computer Organization (continued)

We explored how computers are implemented using **circuits**.

We can also represent a circuit as a **truth table**, which shows the outputs associated with every possible combination of **binary** inputs.

We also discussed how these concepts can be abstracted, by implementing **addition via circuits** and implementing **text and colors via binary**.

Upcoming Topics

In the next unit, we'll dive deeper into programming by focusing on algorithm design. We'll discuss **data structures** (new ways to organize data) and **efficiency** (how to determine how 'fast' our algorithms are).

We'll get back to computer organization in Unit 3, where we'll discuss how computers **scale up** to work on much larger inputs.