Computing and Programming

Notes for CSC 100 - The Beauty and Joy of Computing
The University of North Carolina at Greensboro

Reminders: What you should be doing!

Blown to Bits, Chapter 1:

- Should have completed "reading reflection"
- Over the next 9 days: participate in online discussion

Before class on Wednesday:

- Read "Introduction to Abstraction" (link on schedule)

Labs:

- Review solution for Lab 2 if you didn't understand everything
  - Video solution walk-through in Canvas
  - Lab 3 builds on Lab 2, so make sure you understand Lab 2!

Question to Start the Day...

Context: Computation takes place as a sequence of steps (can consider a 2 GHz computer to be taking 2 billion steps per second).

Question: What can a computer do in a single step?

Or… Which of the following can be done in a step?

1. Store the number 123 in a variable
2. Compute the sum of a variable plus 5
3. Compute the square root of a number
4. Compute the GCD of two numbers
5. Add together two 1000-digit numbers
6. Find the highest test score in a class of 10 students
7. Draw Alonzo on the screen
8. Find a collection of bank transactions that add up to a target value

A Basic Computational Step

Super-simplified CPU (Central Processing Unit):

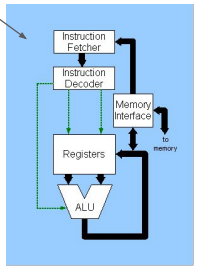


Diagram from Wikimedia Commons

Things to know:

- Data transferred around on wires (dark black arrows in diagram) - fixed number of wires limits how big (# digits) a number can be
- "ALU" is "Arithmetic Logic Unit" - this actually does the computations: two inputs, one output

So in general:

A ***step*** is an operation on two values from a limited range of numbers.

Can't operate on big numbers or more than two things in a single step.

How Fast Can People Compute One Step?

We are going to have a calculation race - how fast are you?

Make sure you have a sheet of paper and pencil/pen

On the following screen are three arithmetic problems

When I change slides, start working on these and solve them as fast as you can - I'll time you!

Raise your hand when you have the answers.

The Problems

```
 132831        942
+476884       x837


 412856
-304158
```

The Answers

```
  132831          942
+476884         x837
  609715        788454

  412856
-304158
  108698
```

How Fast?

If t is the fastest time, then t/3 seconds per calculation (or 3/t calculations per second)

Obviously, computers can do this faster, but...

> In June 2017 the most powerful computer on earth could do 93,015,000,000,000,000 calculations per second (93.015 petaflops).
>
> See http://www.top500.org/

Thinking about computations on this scale is incredibly different from thinking about computations at a few calculations per minute.

This is why Computer Science has become so important!

Some Other Questions...

How accurate were you?

Were all the calculations the same difficulty?

- *What makes some calculations harder than others? A fundamental computer science question!*

What about cost?

- *How much would it cost to do 1 calculation per second non-stop for a year, paying $10/hour?*

Exercise: You identify the steps

Problem: Find the largest value in this list of numbers

338	773	655	72	882	80	533	278	626	689
[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]	[10]

Numbers in [brackets] are positions - use to refer to values (example: value [4])

What you're allowed to do:

- Set aside storage to remember values or positions (these are variables)
- Access a value based on its position
- Do any operation with one or two values (arithmetic, store in a variable, …)
- Make decisions based on comparing two values
- Control your "program" (repeating operations, etc.)

To do:

1. Decide on a general strategy (algorithm)
2. Go around the room executing: each person to identify the next "step" to take

A few more words about "steps"

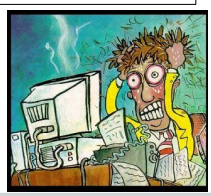
Steps are very simplistic and not very powerful

But… do a few billion every second, and you can make some very complex, "smart-looking" behavior!

Note: While it works for computers, doing lots of stupid things at high speed is not the "smart-looking" behavior that you should aspire to as a college student.

If we had to specify every single step, programmers would not be very productive.

Goal: Make programming easier!



How Programming Languages Help

How you want to describe an animation

Move Alonzo 10 spaces to the right

How the computer must do it

Find what images are behind Alonzo
For each row of the background
  For each column of the background
    Redraw the background pixel

Add 10 to Alonzo's x coordinate

For each row of Alonzo
  For each column of Alonzo
    Draw the pixel of Alonzo

Snap! lets you do this

instead of this

["High Level" vs "Low Level"]

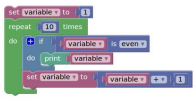
The goals of a good programming language:
  a) is natural and understandable to people
  b) can be efficiently (and unambiguously) turned into actual computer code

Different Languages, Different Styles

Basic structure of many languages is similar

Google Blockly lets you look at the ***same program*** in different languages:

As Blocks



In Python

```
variable = 1
for count in range(10):
  if variable % 2 == 0:
    print(variable)
  variable = variable + 1
```

In JavaScript

```
var variable;

variable = 1;
for (var count = 0; count < 10; count++) {
  if (variable % 2 == 0) {
    window.alert(variable);
  }
  variable = variable + 1;
}
```

Note: Don't get the wrong idea - some languages are *very* different!

See https://blockly-demo.appspot.com/static/demos/code/index.html#p7cnbv

Some Quotes

These are by Alan Perlis (1922-1990): An early computer scientist and first winner of the Turing Award in 1966.

> "A programming language is low level when its programs require attention to the irrelevant."
>
> "A language that doesn't affect the way you think about programming, is not worth knowing."
>
> "There will always be things we wish to say in our programs that in all known languages can only be said poorly."

And since programmers are never satisfied with the programming language they are using, one more quote - from Bjarne Stroustrup (inventor of C++):

> "There are only two kinds of programming languages: those people always bitch about and those nobody uses."

Some Quotes

These are by Alan Perlis (1922-1990): An early computer scientist and first winner of the Turing Award in 1966.

> "A programm
> to the irreleva
> "A language
> not worth kn
> "There will al
> known langu

And since progr
language they a
(inventor of C++

> "There are o
> bitch about and those nobody uses."

The "Turing Award" is the top award given in the field of Computer Science - sometimes referred to as the "Nobel Prize of Computer Science."

Named after Alan Turing (1912-1954), one of the most influential founders of the field of Computer Science.

- Designed computers in WWII to break German codes
- General model of computers named after him ("Turing machines")
- Also stated a test for Artificial Intelligence (the "Turing Test")
- Watch *The Imitation Game*!!