CS 111: Program Design I

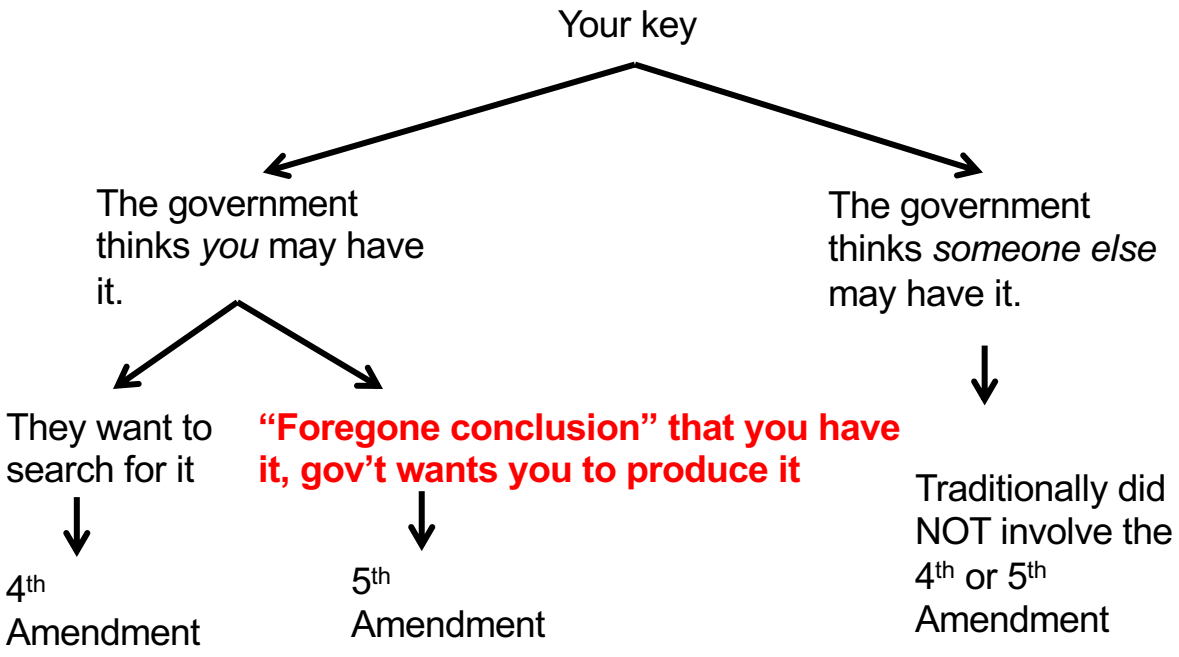
Lecture 4: Computers & Data but Mostly Variables and Strings

Robert H. Sloan & Richard Warner
University of Illinois at Chicago
Sept. 5, 2019

UIC

ENCRYPTION, PRIVACY, & GOVERNMENT POWER (CONT.)

Government Access Generally



A Witness Against Oneself

- On the court's view, it is *not* a foregone conclusion that there is child pornography on the drives.
- Producing the key would tell the government something it does not know
  - And that makes producing the key testimony.
- Why not a forgone conclusion? Because the encrypted drives could be empty.
  - Is that a good reason?
  - *Any* encrypted drive could be empty.
  - So a high hurdle.

Investigative Journalists

- You are an investigative journalists in the United States. You have *conclusive*—but classified—evidence of serious government wrongdoing on your laptop.
- The government—legally—takes your laptop.
- Your hard drive is encrypted.
- Should you have to give the government the key?
- A = Yes, B = No

Something Different: The Backdoor Debate

- A **backdoor** is a secret way into either the computer itself or into a particular piece of software that was left behind by the software developers.
- The government—in particular, the FBI—has been arguing for a backdoor into encryption programs that will allow them to decrypt and read the data the user has encrypted.
  - Two versions:
    - The users are not informed.
    - The users consent to the backdoor.

A Very Simple Backdoor

- Suppose you buy an encryption program using a Caesar cipher. The installation program secretly creates an empty text file, key.txt, on your hard drive.
- The program asks you to type in your secret key:

```
my_key = input('Input your key:')
```

- The program also has this code hidden in it:

```
f = open('key.txt','w') # Open file key.txt
f.write(my_key) # Write key to file key.txt
f.close()       # Close file
```

Questions

- Would you object to having this code hidden in your software?
- The government has for years attempted to get software developers to install government backdoors in encryption software. Do you think this is a good idea?
- It seems fairly clear that the NSA installed secret backdoors in various types of software. Do you think that is a good idea?

First Crypto War: Background

- Into the 1990's, the US restricted the export of encryption technologies to weak encryption.
  - To avoid developing two products, US companies offered weak encryption domestically too.
  - The restrictions were lifted in 2000.
    - *Bernstein v. United States*, http://cr.yp.to/export/problem.html
    - Daniel Bernstein, https://cs.uic.edu/profiles/daniel-j-bernstein/
- The FBI was not happy. The FBI appealed to CALEA (Communications Assistance for Law Enforcement Act 1994) put a backdoor in all landline communications.
- It wanted—still wants—the same for encrypted digital communications.

The Crypto Wars

- First Crypto War: Cold War to 2000
  - Export restrictions on strong encryption
- Second Crypto War: Post 9/11 to ?
  - FBI demands for a backdoor to counter domestic terrorism
- Third Crypto War: 2019 to ?
  - Attorney General Barr's renewed demands for backdoor
- The Long Running And Ongoing Crypto War
  - NSA counter encryption activities
  - "By 2010, the NSA had developed 'groundbreaking capabilities' against encrypted Internet traffic."

Backdoors Are Bad

- Assume—for now—that the government will never misuse the backdoor.
- Backdoors—like hacking technology in general—always spreads. The bad people will get it.
- The assumption of no misuse is wrong, as history overwhelmingly shows.
  - You design governments *assuming* misuse of power.
- You don't need backdoors for investigations.

COMPUTERS, DATA, SIZES

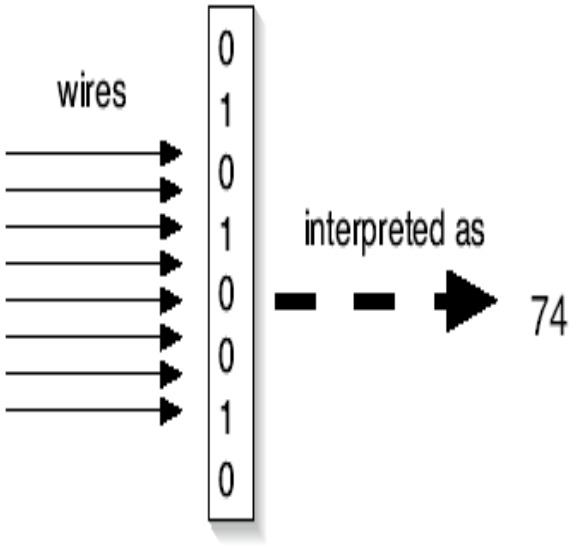
Did **you** register your clicker?

What computers understand

- 0's and 1's.
  - *Everything* is 0's and 1's
- Computers are *exceedingly* stupid
  - Only *data* they understand is 0's and 1's
  - Can do only the simplist things with those 0's and 1's
    - Move this value here
    - Add, multiply, subtract, divide these values
    - Compare these values, and if one is less than the other, go follow this step rather than that one.

Key Concept: Encodings

- But we can *interpret* these numbers any way we want.
  - We can *encode* information in those numbers
- Even notion that computer understands numbers is an interpretation
  - We encode voltages on wires as 0's and 1's
  - Which we can, in turn, interpret as a decimal number



Useful terminology

- 8 bit byte is fundamental unit of memory
- 1 byte is really tiny unit. More often see:

1 kilobyte	1 kB	10^3 bytes	1000 bytes
1 megabyte	1 MB	10^6 bytes	1,000,000 bytes
1 gigabyte	1 GB	10^9 bytes	1 billion bytes
1 terabyte	1 TB	10^{12} bytes	1 trillion bytes

- N.B. 2^{10}, 2^{20}, 2^{30}, 2^{40}, instead of powers of 10 (e.g., 1024 byte kB) also used! See, e.g., https://support.apple.com/en-us/HT201402

Units to data amounts

- 1GB can hold about
  - 250 photos (Based on 12 Megapixel camera, e.g., iPhone 7, 8, X, XR and JPEG 100% quality)
  - 7–30 minutes of video (depending on quality, frames per second, and 720–1080p)
  - 40,000 pages of simple Word docs
- So 1 TB hard drive can hold 200 hours of video and 10,000 photos with room left over (assuming video high-quality 720p @ 30fps)

Units to $ @ amazon, Sept. 2019

8TB = $139.99, or $17.50 per TB

What is a computer?

- A device that executes a stored program (sequence of instructions).

- A program is a particular writing of a recipe in some particular language. (Recipe is likely to be in English or French or Arabic or Hindi; program in a *programming language* such as C, Java, Visual Basic, or Python)

All computers consist of 3 components

- **Memory–**stores program and data (information)
  - Primary: RAM (Random Access Memory) “memory”
  - Secondary: hard drive “storage”
- **Central Processing Unit (CPU)**
  - **Control–**fetch next instruction, **decode** it, execute it
  - **Arithmetic Logic Unit–**perform **simple** operations on data (add, compare two for equality, etc.)
- **Input/Output**

Detour: Specs for a computer

- Ads for computers typically give:
  - Speed of the CPU (in GHz, say 1.0–3.25)
  - Amount of RAM in GB (say, 8–16)
  - Size of hard drive in GB or TB
  - Which “nice” I/O devices (e.g., retina display)
- Interestingly, *perceived* speed today often depends heavily on amount of RAM

Moore's Law

- Gordon Moore, one of founders of Intel, made claim that (essentially) computer power doubles for the same dollar every 18 months.
- This has held true for over 40 years.
  - (Note: some think the end is finally near.)
- Go ahead! Make your computer do the same thing to encrypt all 500,000 characters of the book you wrote! It doesn't care! And it won't take much time either!

Please complete the pre-class survey (now)

https://oberlin.qualtrics.com/jfe/form/SV_2ucqvlNbleRgoHX

Link also available from Welcome page of course Blackboard site

Alternate Problem if not taking survey:

Evaluate in your head; check with computer when done:

1. 5 ** 2
2. 9 * 5
3. 15 / 12
4. 12 / 15
5. 15 / 5 – 2
6. 12 // 15
7. 5 % 2
8. 9 % 5
9. 15 % 12
10. 12 % 15
11. 6 % 6
12. 0 % 7

VARIABLES CONTINUED & TYPES

Objects have a type

- And support operations appropriate for their type

- 2 + 2 → 4
- 3 * 2 → 6
- "hot" + "dog" → "hotdog"
  - But no "hot" * "dog"

```
In [1]: justice1 = 'Marshall'
In [2]: justice1
Out[2]: 'Marshall'
In [3]: justice2 = "O'Connor"
In [4]: justice2
Out[4]: "O'Connor"
```

```
In [5]: justice1 = justice1 + justice2
In [6]: justice1
```

Name	Value
justice1	Marshall
Justice2 is Gone	

A

Name	Value
justice1	MarshallO'Connor
justice2	O'Connor

B

Name	Value
justice1	MarshallO'Connor
justice2	MarshallO'Connor

C

Name	Value
justice1	Marshall
justice2	O'Connor

D

E. I don't know

Types

- Objects come in few different types.
  - E.g., strings vs. numbers
- In Python computer (i.e., interpreter) generally figures it out for us, but we still need to know little bit about this since, e.g.,

```
In [7]: justice1+justice2
Out[7]: "Sandra Day O'ConnorJohn Marshall"
```

Types

- Objects come in few different types.
  - E.g., strings vs. numbers
- In Python computer (i.e., interpreter) generally figures it out for us, but we still need to know little bit about this since, e.g.,

```
In [7]: justice2+justice1
Out[7]: "Sandra Day O'ConnorJohn Marshall"
In [8]: 3 + justice1
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: unsupported operand type(s) for +: 'int' and 'str'
```

Some Python Types

	Python Type	Example(s)
String	String	"argle-bargle"
Integer (whole number)	Integer	3, 0, 17, 42, -21, 100001
Decimal number	Float	3.14159
Boolean (true/false)	Boolean	True, False

Types: Pick row that is 100% correct

iClicker Choice	Integer	Float	String
A	1	2.25	""
B	"1"	4.4	'h'
C	1.0	2.0	"hello"
D	1	2.0	goodbye

We love you Python, oh yes we do!

- We have now covered well over half of everything you will need to know about types for this semester
- Types *much* bigger hassle in Java, C, C++

Assignment to variables: Semantics

<variable> = <expression>

1. Evaluate <expression>
2. Put that value into computer's memory and attach name <variable> as "sticky note" giving name for that memory location

Expressions

- Can be simple value, e.g.,
  - "Sandra Day O'Conner" or 17
- Also can be almost any mathematical statement