APS105 Winter 2012

Jonathan Deber
jdeber -at- cs -dot- toronto -dot- edu

Lecture 35
April 13, 2012

Today

- Exam Info/Tips
- Review
- Big Picture

Exam Info

- Check Registrar's website for timing/location
- 2.5 hours
- No external aids
- Cumulative (emphasis on later topics)

Office Hours

- I will have some during the exam period

Exam Tips

- Read the whole thing first
- Read every question carefully

- Make things easy for the marker

What We're Asking

- More than just "what does the function do?"

- Do we want a complete program?
- Do we want a code snippet/fragment?
- Do we want an example of using a function?

Question 0

Write your student number in the space provided at the bottom of each odd-numbered page.

Failure to do so will result in a 2 mark deduction.

Buggy Code Question

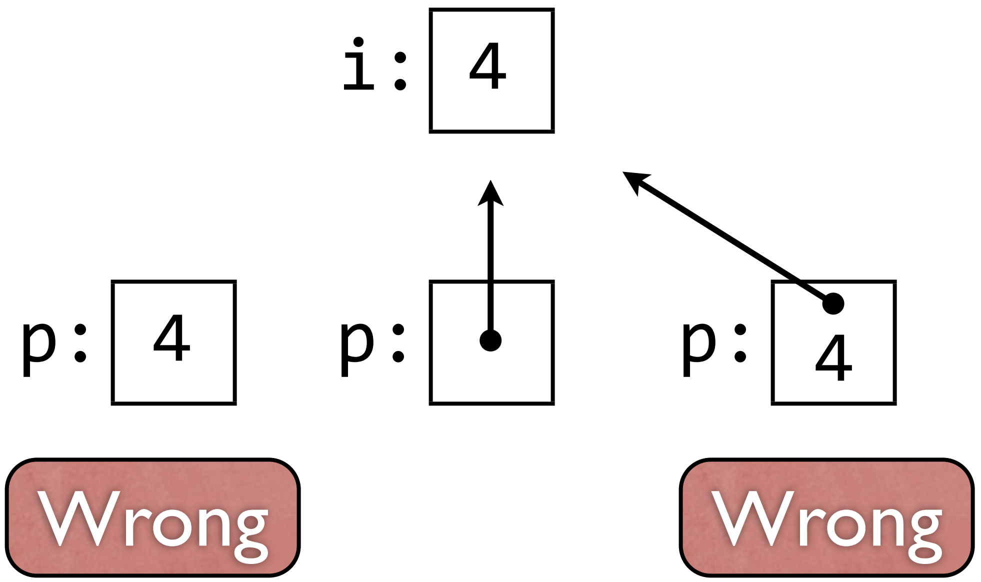
```
1
 voi printHello(void)
 {
       printf('Hello');
 }
                  2
```

1) "void" is misspelled "voi"

2) Format string in single quotes, should be in double quotes

Pointers Point!

```
int i = 4;
int *p = &i;
```



Approaching Coding Questions

- Read it carefully
    - *What am I being asked to do?*
- Figure out the pseudocode
    - *How do I go about doing it?*
- Write the code
    - *How do I translate the pseudocode into C code?*
- Re-read the question
    - *Does my code do what it was supposed to?*

How to Handwrite Code

- We realize that there is no compiler on paper

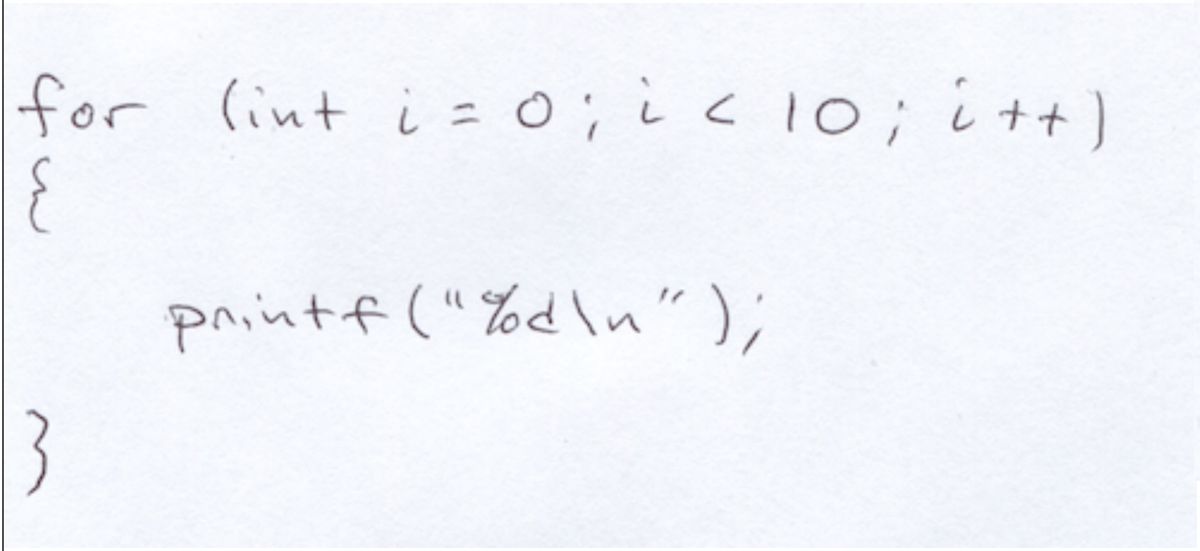
```
printf("%d" value);
printf("%d". value);
```

- However, consistent or egregious syntax errors are a problem

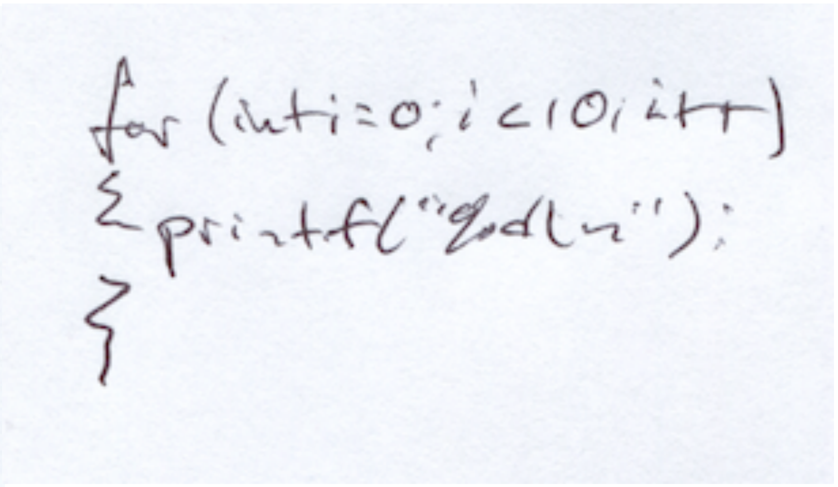
```
while { if [! 45.3] > 3} {true};
```

How to Handwrite Code

- Some symbols are hard to draw ({ } and &)
- Indenting
- Tell us what you’re doing
- Leave lots of space



```
for (int i = 0; i < 10; i++)
{
    printf("%d\n");
}
```



```
for (inti=0;i<10;i++)
{printf("%d\n");
}
```

```
for  (int  i = 0 ; i < 10 ; i++)
{

    printf("%d\n");

}
```

```
for (int i=0; i<10; i++)
{ printf("%d\n");
}
```

```
for  (int  i = ~~0~~1 ; i <=10 ; i++)
{
  if (i != 5)
  {
    printf("%d\n");
  }
}
```

```
for (int i=~~0~~1; i <=10; i++)
{if(i!=5)
{printf("%d\n");
}}
```

Function Reference Sheet

- We will provide a reference page with documentation for some built-in functions
- You are free to use other (non-prohibited) functions as well
- You do not have to use every function on the sheet

Exam Period Tips

- Figure out a study schedule
  - What material for what course on what day
- Don’t completely ignore exams that are later
- Schedule breaks

Review

What We've Covered

- What's a computer?
- What does it mean to program it?
- How to we take an English problem and translate it to a program?

Source: http://cm.bell-labs.com/cm/cs/who/dmr/picture.html

What We've Covered

- What's a computer?
- What does it mean to program it?
- How to we take an English problem and translate it to a program?
- How do we write a program?
- What are the parts of a program?
  - Variables, expressions, statements

What We've Covered

- Types
  - `int`, `double`, `char`
  - Arithmetic operators
- Style
- Reading/writing data
  - `printf()`, `scanf()`, conversion specifiers
- Booleans (`bool` and Boolean operators)
- Relational and comparison operators

What We've Covered

- `if`, `else if`, and `else`
- `while` and `for` loops
- Arrays
- Functions (parameters, return values, scope)
  - `math.h`, Random Numbers
  - Helper Functions
- Pointers
  - Pointer Arithmetic

What We've Covered

- Strings
  - Literals, arrays, pointers
  - `string.h` (including functions you should never use)
  - Reading/writing (including functions you should never use)
- Arrays of pointers
- Dynamic memory
  - `malloc()`, `free()`, `realloc()`
  - Dynamically allocated arrays, `sizeof()`
  - Dynamic memory bugs

What We've Covered

- `NULL`
- Multi-file programs
- Structures (and `typedef`)
  - Pointers to `structs`
  - Dot and arrow operators
- Commenting, testing, and debugging

Grace Hopper’s Lab Notebook

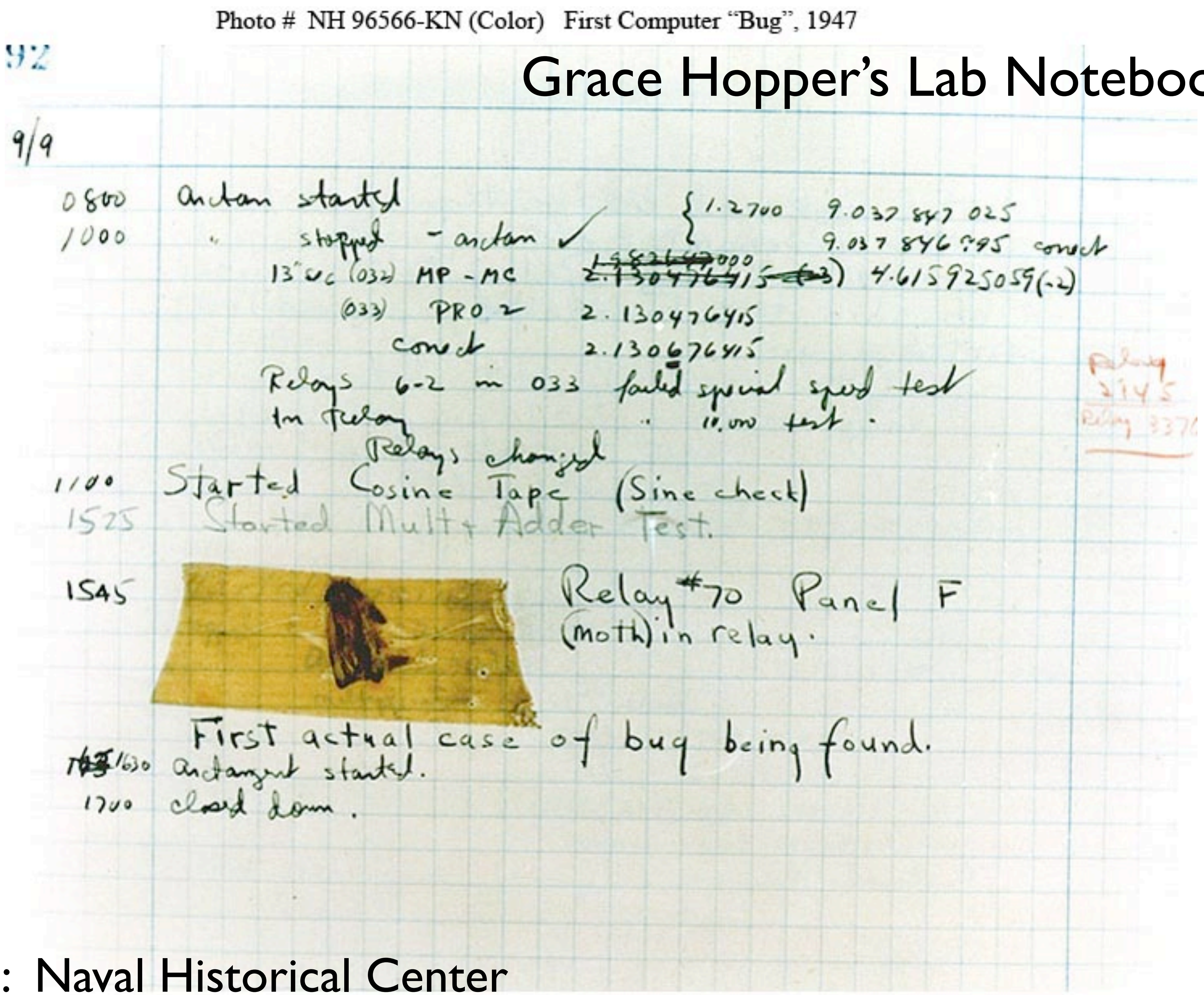


Photo # NH 96566-KN (Color) First Computer “Bug”, 1947

92

9/9

0800 Arctan started {1.2700 9.037 847 025

1000 " stopped - arctan ✓ 9.037 846 995 correct

13" UC (032) MP - MC ~~1.982147000~~ ~~2.130476415~~ (23) 4.615925059(-2)

(033) PRO 2 2.130476415

correct 2.130676415

Relays 6-2 in 033 failed special speed test

In relay " 10,000 test .

Relays changed

1100 Started Cosine Tape (Sine check)

1525 Started Mult + Adder Test.

1545 Relay #70 Panel F (moth) in relay.

First actual case of bug being found.

~~1630~~ 1630 Antangent started.

1700 closed down.

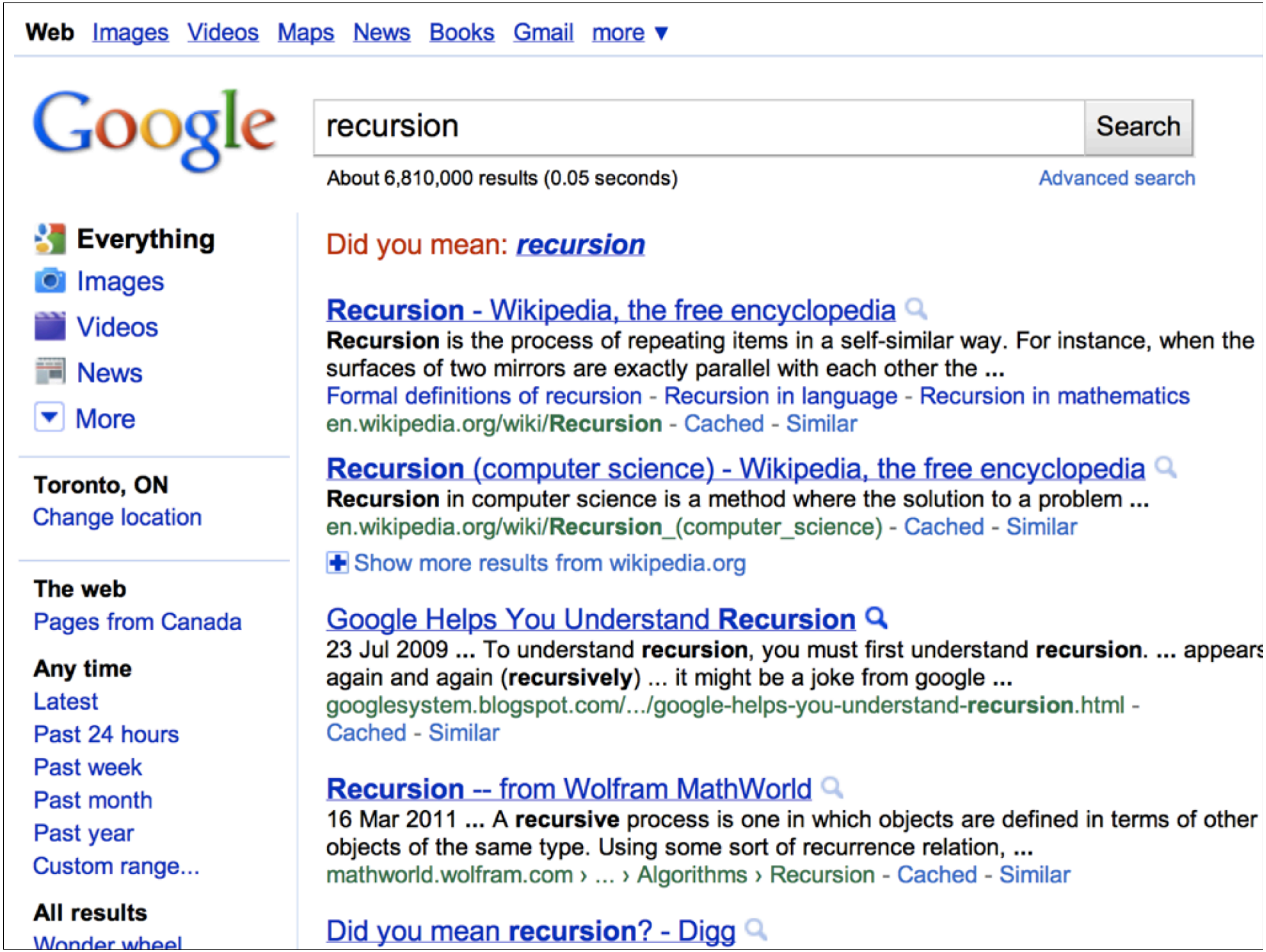
Relay 2145

Relay 3370

Image: Naval Historical Center

What We've Covered

- Dynamic memory bugs
- `NULL`
- Structures (and `typedef`)
  - Pointers to `structs`
  - Dot and arrow operators
- Commenting, testing, and debugging
- Recursion



About 6,810,000 results (0.05 seconds)

Toronto, ON

The web

Any time

All results

Did you mean: ***recursion***

Recursion - Wikipedia, the free encyclopedia
Recursion is the process of repeating items in a self-similar way. For instance, when the surfaces of two mirrors are exactly parallel with each other the ...
Formal definitions of recursion - Recursion in language - Recursion in mathematics
en.wikipedia.org/wiki/**Recursion** - Cached - Similar

Recursion (computer science) - Wikipedia, the free encyclopedia
Recursion in computer science is a method where the solution to a problem ...
en.wikipedia.org/wiki/**Recursion**_(computer_science) - Cached - Similar

Show more results from wikipedia.org

Google Helps You Understand **Recursion**
23 Jul 2009 ... To understand **recursion**, you must first understand **recursion**. ... appears again and again (**recursively**) ... it might be a joke from google ...
googlesystem.blogspot.com/.../google-helps-you-understand-**recursion**.html - Cached - Similar

Recursion -- from Wolfram MathWorld
16 Mar 2011 ... A **recursive** process is one in which objects are defined in terms of other objects of the same type. Using some sort of recurrence relation, ...
mathworld.wolfram.com › ... › Algorithms › Recursion - Cached - Similar

Did you mean **recursion**? - Digg

What We've Covered

- Dynamic memory bugs
- `NULL`
- Structures (and `typedef`)
  - Pointers to `structs`
  - Dot and arrow operators
- Commenting, testing, and debugging
- Recursion
- Linked lists

What We've Covered

- `ctype.h` character functions
- Searching
  - Sequential
  - Binary
- Sorting
  - Maintain sorted order while adding
  - Simple sorts (bubble, selection, insertion)
  - Complex sort (merge sort)

Building on the Foundation

Real Programs

- “Real” programs are built from the same tools
- Much more complicated
  - More languages (maybe)
  - More libraries
  - More data structures
  - More algorithms
- Could be millions of lines of code

Web Browser

- Read a (long) string containing the page's HTML
- Process the string, and build up data structures in memory
- Traverse those data structures, and figure out what to draw
- Draw the page on screen with calls to OS drawing functions

Image Editing

- An image is just an array of pixels
- Editing it is just changing those pixels

Operating System

- Data structures hold info about memory, files, *etc.*
- Implement a bunch of functions that provide an interface
  - Other software can call them
  - Hardware can trigger them
- Wait for some bit of hardware or software to request something, and process that request

Closing Thoughts

- Computers do exactly what you tell them
- To work with them, we need to think algorithmically
- Computing is built out of abstractions

Thank You!