What is Programming?

CTEC1332 Software Engineering Practices

Version **1.0**; Last Modified: 2023-09-12

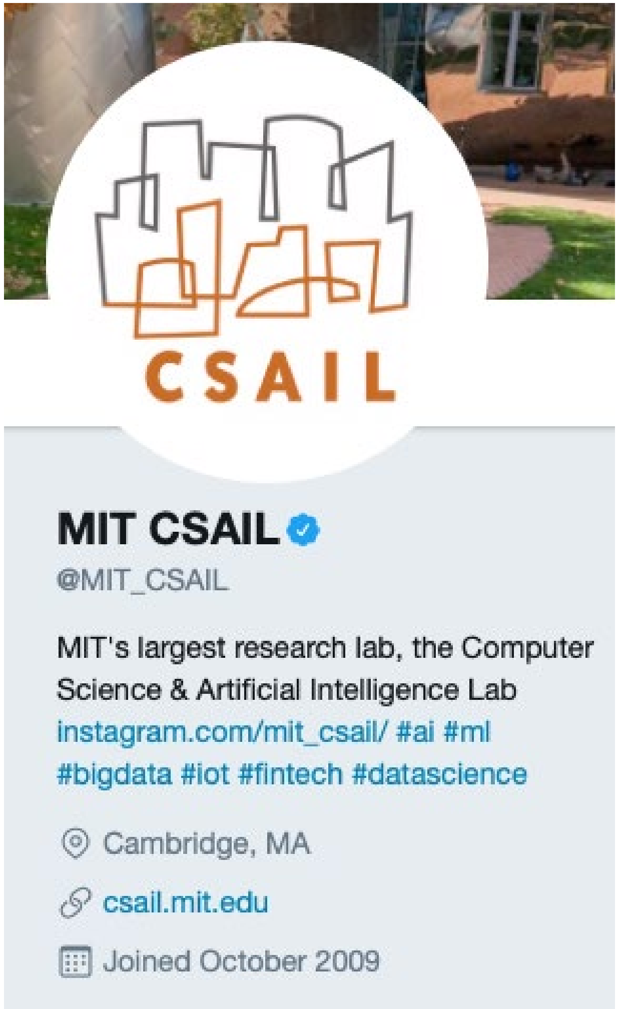
Niagara College Canada
APPLIED DREAMS.

NiagaraCollege.ca

Programming in Six Words...

- From MIT's Computer Science & AI Lab...

https://twitter.com/MIT_CSAIL/status/1143896699947114496



MIT CSAIL

@MIT_CSAIL

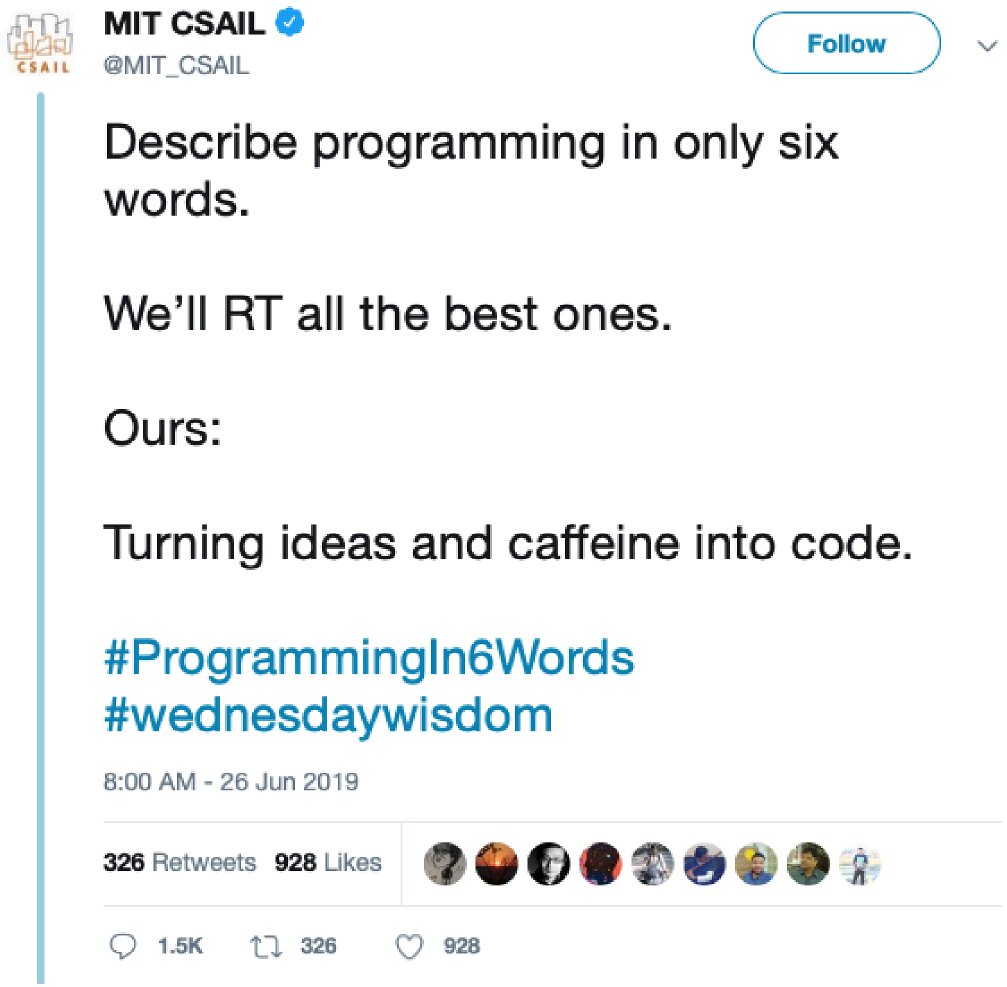
MIT's largest research lab, the Computer Science & Artificial Intelligence Lab instagram.com/mit_csail/ #ai #ml #bigdata #iot #fintech #datascience

Cambridge, MA

csail.mit.edu

Joined October 2009

Programming in Six Words...



MIT CSAIL
@MIT_CSAIL

Describe programming in only six words.

We'll RT all the best ones.

Ours:

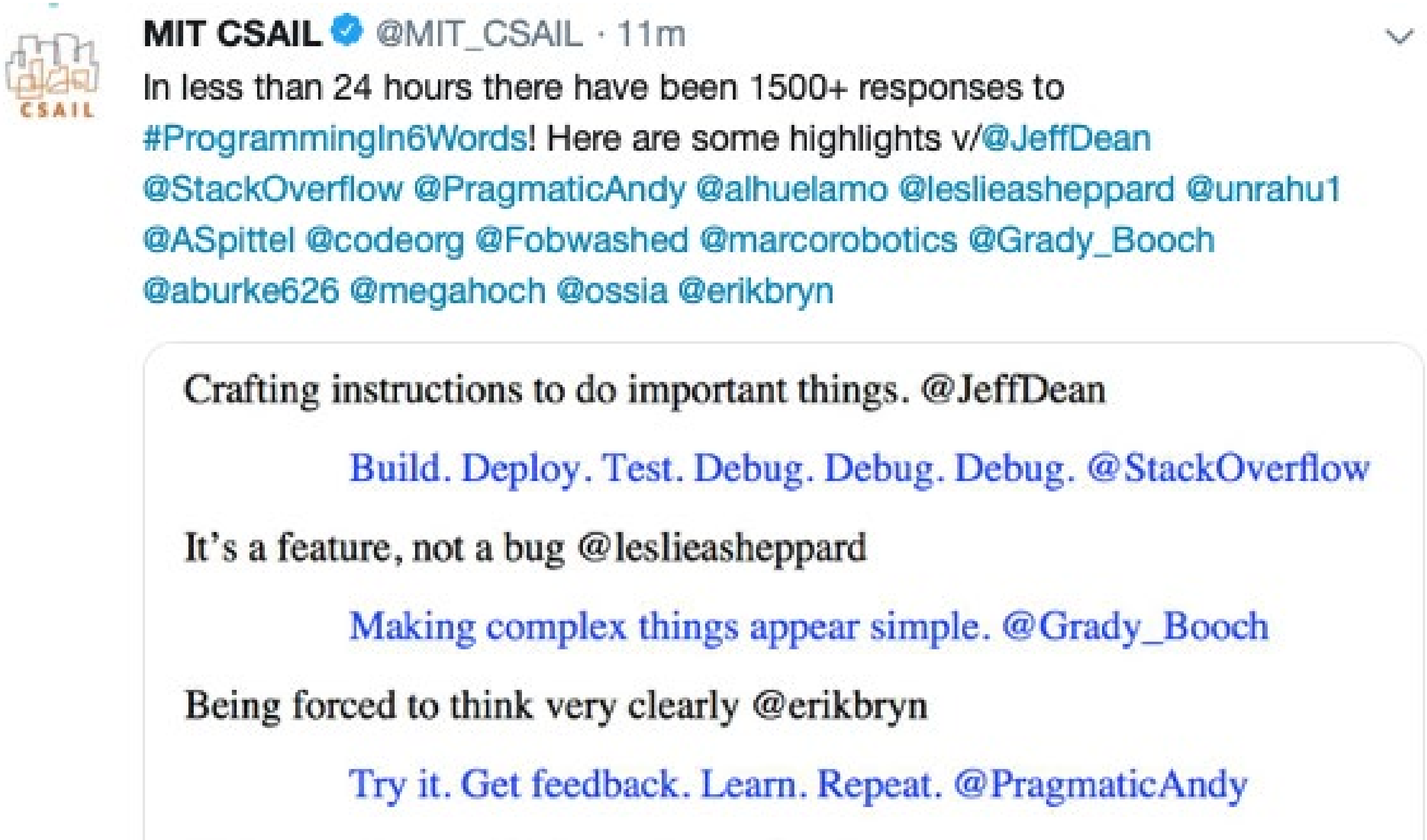
Turning ideas and caffeine into code.

#ProgrammingIn6Words #wednesdaywisdom

8:00 AM - 26 Jun 2019

326 Retweets 928 Likes

Programming in Six Words...



MIT CSAIL @MIT_CSAIL · 11m

In less than 24 hours there have been 1500+ responses to #ProgrammingIn6Words! Here are some highlights v/@JeffDean @StackOverflow @PragmaticAndy @alhuelamo @leslieasheppard @unrahu1 @ASpittel @codeorg @Fobwashed @marcorobotics @Grady_Booch @aburke626 @megahoch @ossia @erikbryn

Crafting instructions to do important things. @JeffDean

Build. Deploy. Test. Debug. Debug. Debug. @StackOverflow

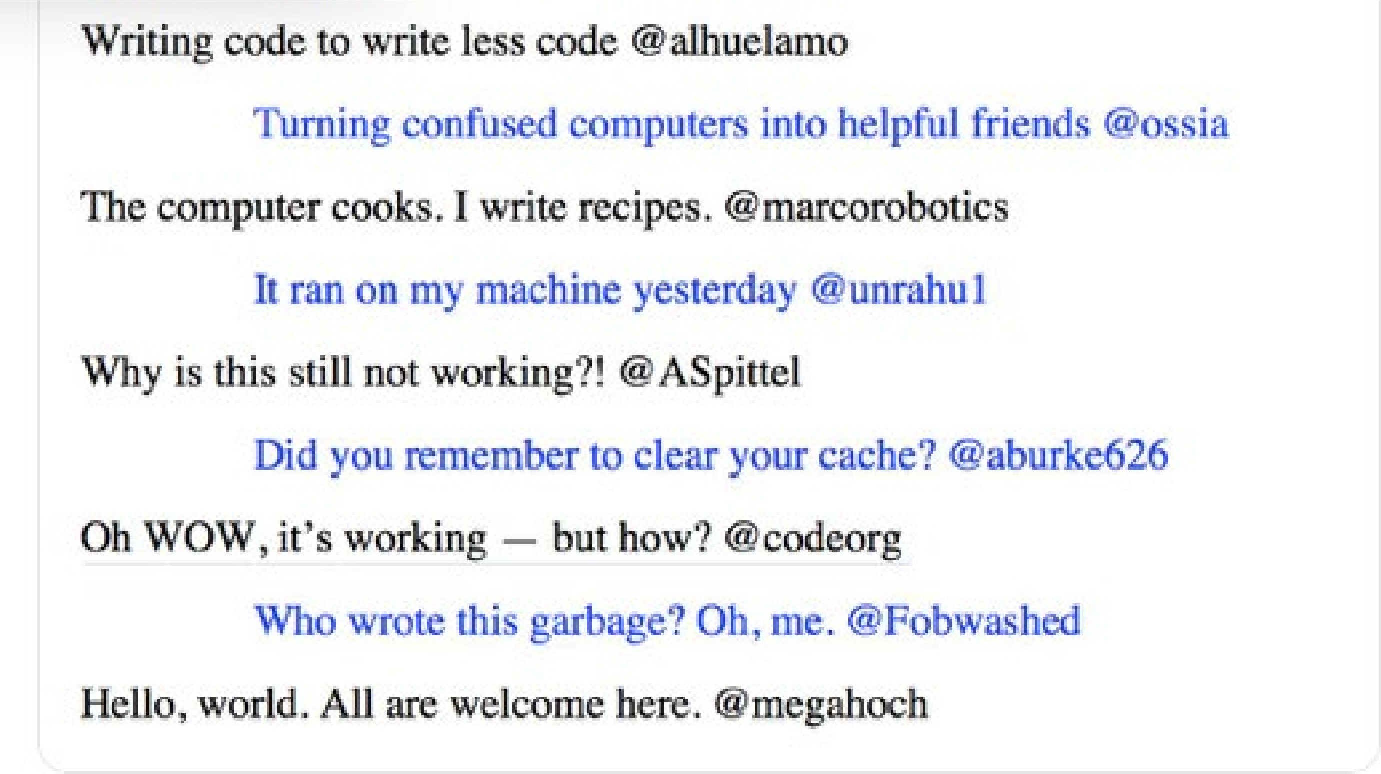
It's a feature, not a bug @leslieasheppard

Making complex things appear simple. @Grady_Booch

Being forced to think very clearly @erikbryn

Try it. Get feedback. Learn. Repeat. @PragmaticAndy

Programming in Six Words...



Writing code to write less code @alhuelamo

Turning confused computers into helpful friends @ossia

The computer cooks. I write recipes. @marcorobotics

It ran on my machine yesterday @unrahu1

Why is this still not working?! @ASpittel

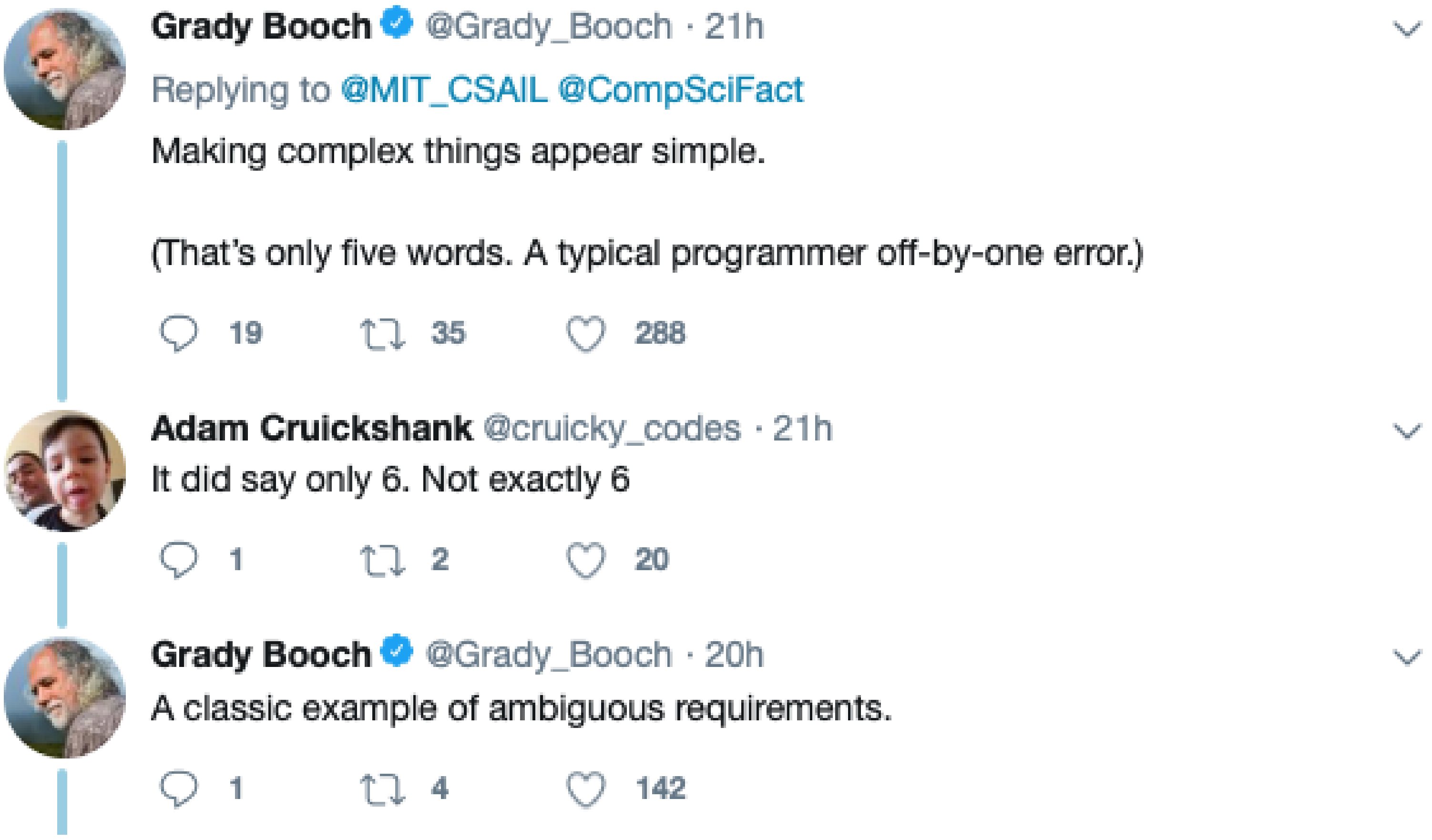
Did you remember to clear your cache? @aburke626

Oh WOW, it's working — but how? @codeorg

Who wrote this garbage? Oh, me. @Fobwashed

Hello, world. All are welcome here. @megahoch

Programming in Six Words...



Grady Booch @Grady_Booch · 21h

Replying to @MIT_CSAIL @CompSciFact

Making complex things appear simple.

(That's only five words. A typical programmer off-by-one error.)

Adam Cruickshank @cruicky_codes · 21h

It did say only 6. Not exactly 6

Grady Booch @Grady_Booch · 20h

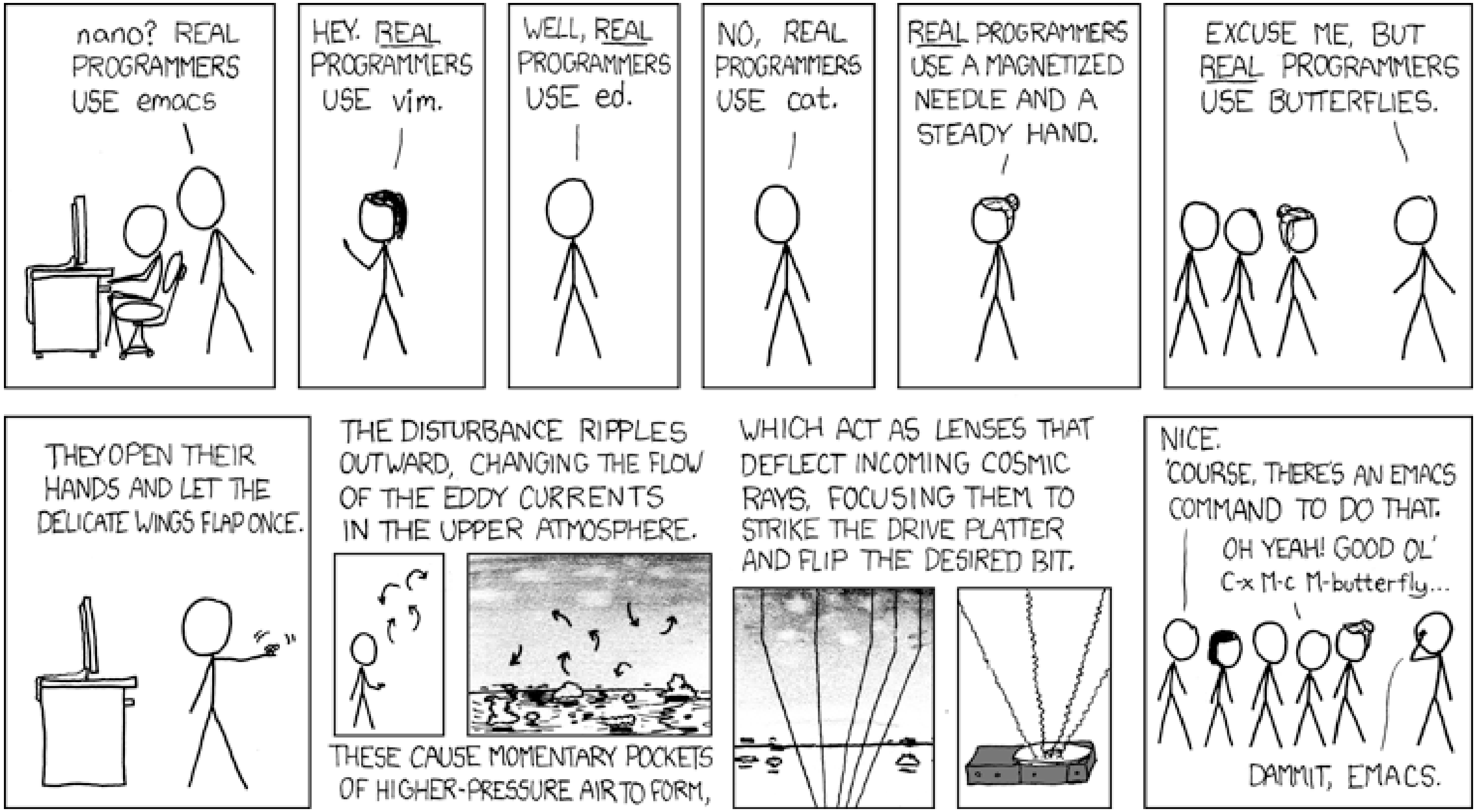
A classic example of ambiguous requirements.

https://en.wikipedia.org/wiki/Grady_Booch

Programming

- Computers are really very dumb **machines** -- they only do what they are told to do.

Programming


nano? REAL PROGRAMMERS USE emacs
HEY. REAL PROGRAMMERS USE vim.
WELL, REAL PROGRAMMERS USE ed.
NO, REAL PROGRAMMERS USE cat.
REAL PROGRAMMERS USE A MAGNETIZED NEEDLE AND A STEADY HAND.
EXCUSE ME, BUT REAL PROGRAMMERS USE BUTTERFLIES.
THEY OPEN THEIR HANDS AND LET THE DELICATE WINGS FLAP ONCE.
THE DISTURBANCE RIPPLES OUTWARD, CHANGING THE FLOW OF THE EDDY CURRENTS IN THE UPPER ATMOSPHERE.
THESE CAUSE MOMENTARY POCKETS OF HIGHER-PRESSURE AIR TO FORM,
WHICH ACT AS LENSES THAT DEFLECT INCOMING COSMIC RAYS, FOCUSING THEM TO STRIKE THE DRIVE PLATTER AND FLIP THE DESIRED BIT.
NICE.
'COURSE, THERE'S AN EMACS COMMAND TO DO THAT.
OH YEAH! GOOD OL' C-x M-c M-butterfly...
DAMMIT, EMACS.

Real programmers set the universal constants at the start such that the universe evolves to contain the disk with the data they want.

https://xkcd.com/378/

Programming

- Most computers perform their **operations** on a very primitive level.

- The basic operations of a computer system is called the computer's **instruction set**.

- In order to solve a problem using a computer, we must express the solution to that problem in a **language** that the computer can understand -- through the instruction set.

DILBERT™
by SCOTT ADAMS

Programming

WHEN I STARTED PROGRAMMING, WE DIDN'T HAVE ANY OF THESE SISSY "ICONS" AND "WINDOWS."
© 1992 United Feature Syndicate, Inc.
S.Adams
ALL WE HAD WERE ZEROS AND ONES -- AND SOMETIMES WE DIDN'T EVEN HAVE ONES.
9-8
I WROTE AN ENTIRE DATABASE PROGRAM USING ONLY ZEROS.
YOU HAD ZEROS? WE HAD TO USE THE LETTER "O."

https://dilbert.com/search_results?page=1&terms=Agile+Programming

Retrieved on August 8, 2019
Dilbert © 2019, Andrews McMeel Syndication

Programming

- A computer **program** is just a collection of the instructions necessary to solve a specific problem.

- The approach or method that we use to solve the problem is called an **algorithm**.

- To develop a program to solve a particular problem, we first express the solution to the problem in terms of an algorithm.

- With the algorithm in hand, we can then write the instructions necessary to implement the algorithm on a particular computer system.

"A computer is like a violin. You can imagine a novice trying first a phonograph and then a violin. The latter, he says, sounds terrible. That is the argument we have heard from our humanists and most of our computer scientists. Computer programs are good, they say, for particular purposes, but they aren’t flexible. Neither is a violin, or a typewriter, until you learn how to use it.”

— Marvin Minsky, “Why Programming Is a Good Medium for Expressing Poorly-Understood and Sloppily-Formulated Ideas”

Example - Algorithms

- Problem:

  Develop a program that tests if a number is even or odd.

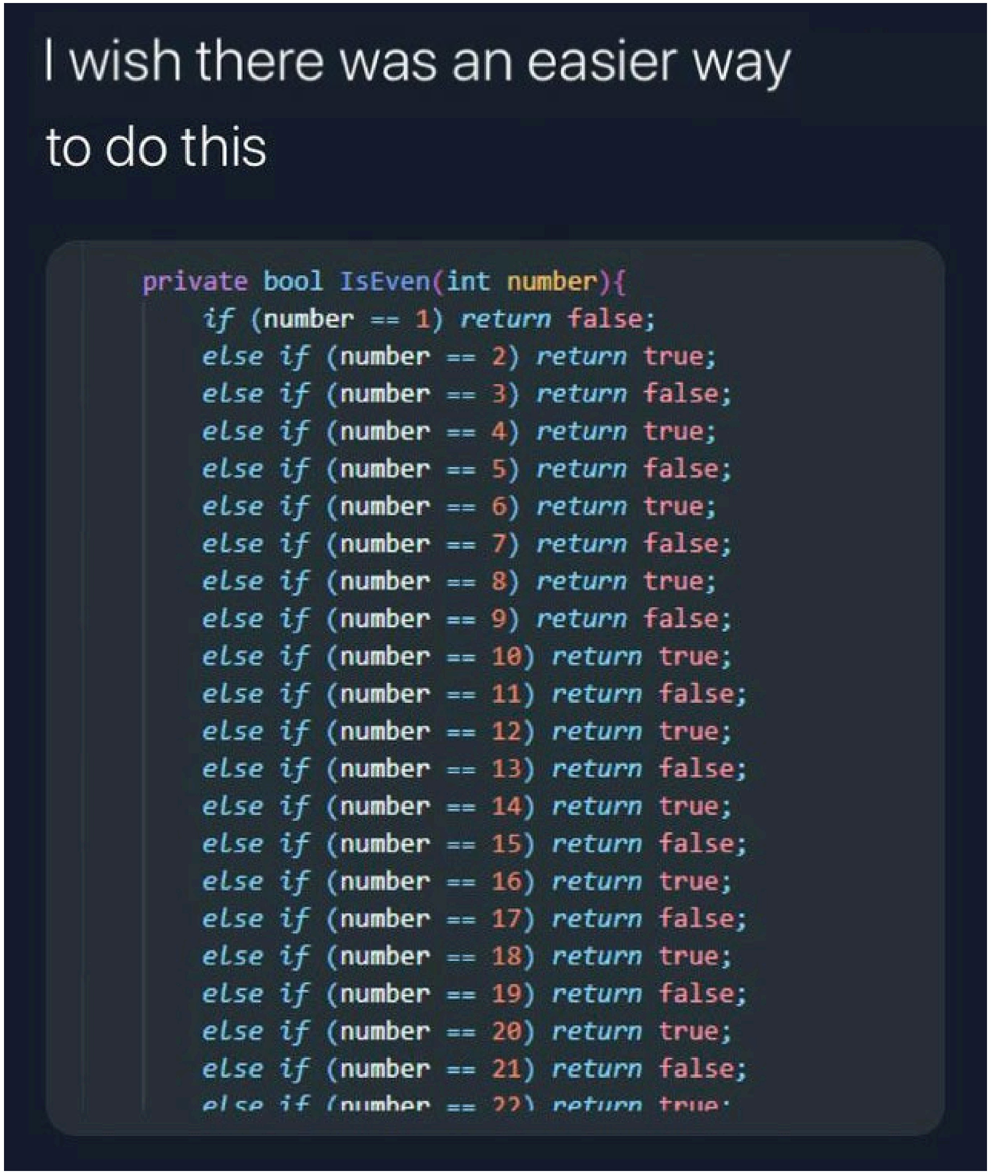
Example - Algorithms

- Problem:
  Develop a program that tests if a number is even or odd.

- Some Mathematical Theory:
  - https://mathworld.wolfram.com/EvenNumber.html
  - https://mathworld.wolfram.com/OddNumber.html

- Problem:

  Develop a program that tests if a number is even or odd.



I wish there was an easier way to do this

```
private bool IsEven(int number){
    if (number == 1) return false;
    else if (number == 2) return true;
    else if (number == 3) return false;
    else if (number == 4) return true;
    else if (number == 5) return false;
    else if (number == 6) return true;
    else if (number == 7) return false;
    else if (number == 8) return true;
    else if (number == 9) return false;
    else if (number == 10) return true;
    else if (number == 11) return false;
    else if (number == 12) return true;
    else if (number == 13) return false;
    else if (number == 14) return true;
    else if (number == 15) return false;
    else if (number == 16) return true;
    else if (number == 17) return false;
    else if (number == 18) return true;
    else if (number == 19) return false;
    else if (number == 20) return true;
    else if (number == 21) return false;
    else if (number == 22) return true;
```

https://www.reddit.com/r/ProgrammerHumor/comments/i0mwif/_/

Example - Algorithms

- Problem:
  Develop a program that tests if a number is even or odd.

- Some Digital Theory:
  - https://en.wikipedia.org/wiki/Binary_number
  - https://en.wikipedia.org/wiki/Two%27s_complement

Example - Algorithms

- Problem:
  Develop a program that tests if a number is even or odd.

- Algorithm #1:
  Divide the number by two.
  - If the remainder is zero, then the number is even.
  - Otherwise (the remainder is one), the number is odd.

- Algorithm #2:
  Test the least significant bit of the number.
  - If the bit is one, then the number is odd.
  - Otherwise (the bit is zero), the number is even.

Higher-Level Languages

- When computers were first developed, the only way they could be programmed was with **binary numbers** that corresponded directly to the machine instructions and locations in the computer's memory. (**Machine language**).

- **Assembly language** enabled the programmer to work with the machine on a slightly higher level.

- A special program called an **assembler** translates the assembly language programs from its symbolic format into machine language.

- Because a one-to-one correspondence exists between assembly language instructions and machine language instructions, assembly language is called a **low-level language**.

Higher-Level Languages

- The programmer must still learn the instruction set of the particular computer system in order to write a program.

- Assembly language programs are not **portable** -- they will not run on a different type of computer without being rewritten (**machine-dependent**).

- Operations of a **higher-level language** are much more sophisticated -- one **statement** would result in many different machine instructions being executed.

- Standardization of the syntax of a higher-level language mean that a program could be written to be **machine independent** -- a program could run on any machine that supported the language with few or no changes.

- A **compiler** is a special program that translates the statements of a program developed in a higher-level language into machine language.

Operating Systems

- An **operating system** is a program that controls the entire operation of a computer system.

- All **input/output (I/O)** operations are channeled through the operating system.

- The operating system must also manage the computer’s resources and must handle the **execution** of programs.

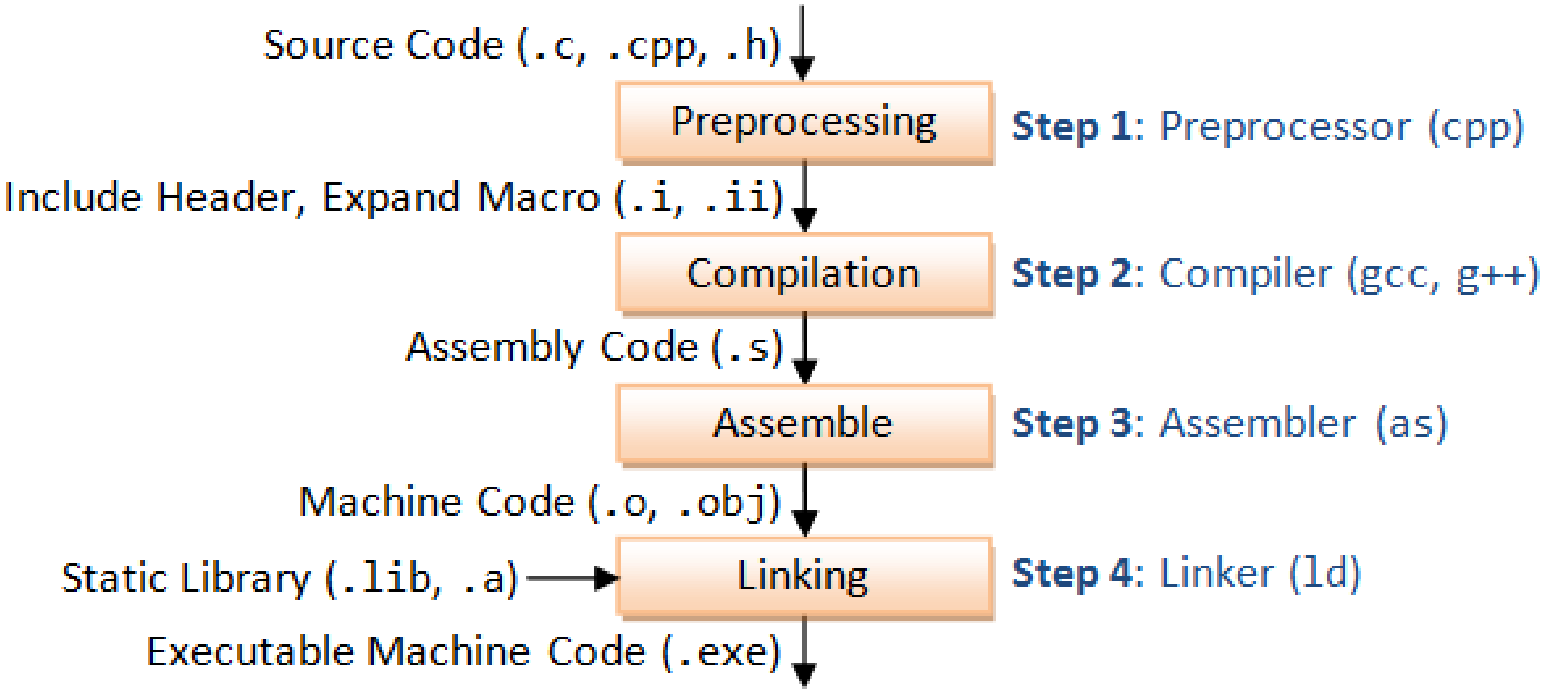
Compiling Programs

- A **compiler** is a program that analyzes a program developed in a particular computer language and then translates it into a form that is suitable for execution on your particular computer system.
- The program that is to be compiled is first typed into a **file** on the computer.
- A **text editor** must usually be used to enter a program into a file.
- The program that is entered into the file is known as the **source program** or **source code**.
- Once the source program has been entered into a file, you can then have it **compiled**.

Compiling Programs

- The compilation process is initiated by typing a special command. When this command is entered, the name of the file that contains the source program must also be specified.
- Step 1: The compiler examines each statement in the source program and checks for **syntax** and **semantic** errors.
- Step 2: The compiler translates each statement to a "lower" form -- usually to assembly language, or directly to machine language. This translation is called **object code**.
- Step 3: After the program has been translated into object code, it is ready to be **link**ed -- the program is joined with other programs that have been previously written and programs from the operating system's **library**.
- Step 4: The final linked file is called an **executable** file, and is stored in another file on the system -- ready to run or be executed.

C Compilation Process


Source Code (.c, .cpp, .h)
Preprocessing
Step 1: Preprocessor (cpp)
Include Header, Expand Macro (.i, .ii)
Compilation
Step 2: Compiler (gcc, g++)
Assembly Code (.s)
Assemble
Step 3: Assembler (as)
Machine Code (.o, .obj)
Static Library (.lib, .a)
Linking
Step 4: Linker (ld)
Executable Machine Code (.exe)

https://medium.com/@monkeyninja147/how-gcc-compiles-c-files-4a91318ae22c

C Compilation Process

- To *build* a C program means to compile source code from one or more files and then link those files into an executable file (.exe), a dynamic-load library (.dll) or a static library (.lib).

- Basic C compilation involves three main steps:

  - The C preprocessor transforms all the #directives and macro definitions in each source file. This creates a *translation unit*.

  - The C compiler compiles each translation unit into object files (.obj), applying whatever compiler options have been set.

  - The *linker* merges the object files into a single executable, applying the linker options that have been set.

https://docs.microsoft.com/en-us/cpp/build/projects-and-build-systems-cpp?view=vs-2019

C Compilation Process

- The Microsoft C/C++ compiler, linker, standard libraries, and related utilities comprise the MSVC compiler **toolset** (also called a **toolchain** or "build tools").
- These are included in Visual Studio.
- You can build simple programs by invoking the MSVC compiler (**cl.exe**) directly from the command line.
  - Note that the compiler automatically invokes the C preprocessor and the linker to produce the final output file.

https://docs.microsoft.com/en-us/cpp/build/projects-and-build-systems-cpp?view=vs-2019

Running Programs

- To **execute** (**run**) the program, all you have to do is type in the name of the **executable file**. This **load**s the program into the computer's **memory** and starts execution.

- When the program is executed, each of the statements of the program is **sequentiall**y executed.

- If the program requests any data from the user, called **input**, the program will temporarily suspend its execution and wait for the user to **enter** the data.

- Results that are **display**ed by the program are called **output**. (Normally displayed on the screen or terminal).

- If all goes well (and it probably won't the first time the program is executed), the program will do what it is supposed to do.

Debugging Programs

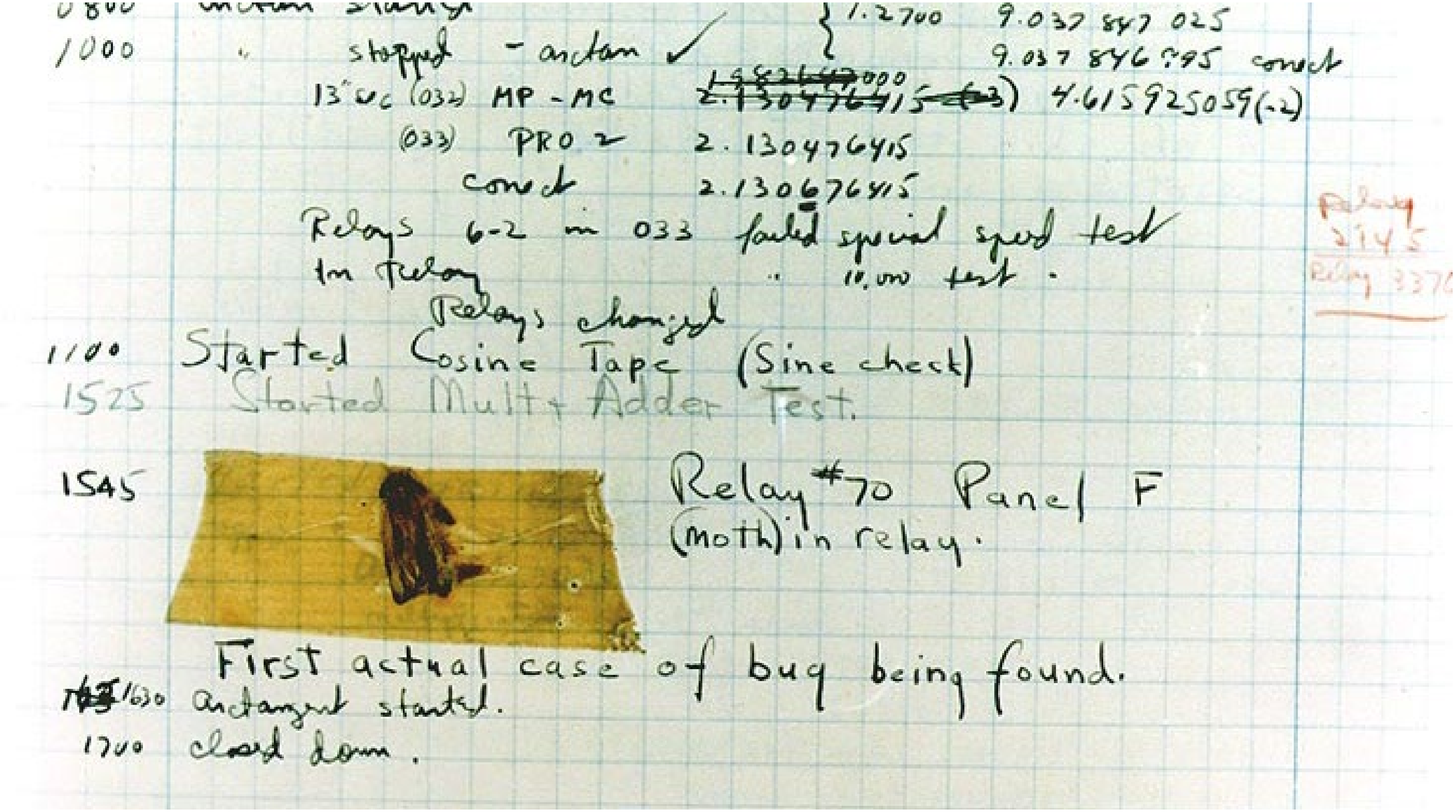
- If things go wrong, it will be necessary to go back and re-analyze the program’s logic. This is called **debugging** -- you try to remove all the **bugs** (features?) from the program.

- In order to **debug** a program, the source code usually has to be changed. Then the entire process of compiling, linking, and executing the program is repeated...

```
“99 bugs in the code,
    99 little bugs
Fix one bug, recompile
100 bugs in the code.”
```

:-)

Debugging Programs



0800 arctan started

1000 " stopped - arctan ✓ } 1.2700 9.037 847 025
9.037 846 995 correct

13"uc (032) MP - MC ~~1.982147000~~ ~~2.130476415 (-3)~~) 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.

~~163~~1630 Arctangent started.

1700 closed down.

Relay 2145
Relay 337

References

- "H96566k" by Courtesy of the Naval Surface Warfare Center, Dahlgren, VA., 1988. - U.S. Naval Historical Center Online Library Photograph NH 96566-KN. Licensed under Public Domain via Wikimedia Commons - https://commons.wikimedia.org/wiki/File:H96566k.jpg#mediaviewer/File:H96566k.jpg
- https://en.wikipedia.org/wiki/Grace_Hopper
- http://sniff.numachi.com/pages/tiPROGBUGS;ttBOT99.html
- https://en.wikipedia.org/wiki/Marvin_Minsky

References

- https://docs.microsoft.com/en-us/cpp/c-language/?view=vs-2019
  - C language documentation
- https://docs.microsoft.com/en-us/cpp/?view=vs-2019
  - Microsoft C++, C, and Assembler documentation
- https://docs.microsoft.com/en-us/cpp/build/walkthrough-compile-a-c-program-on-the-command-line?view=vs-2019
  - Walkthrough: Compile a C program on the command line
- https://docs.microsoft.com/en-us/cpp/get-started/tutorial-console-cpp?view=vs-2019
  - Create a console calculator in C++