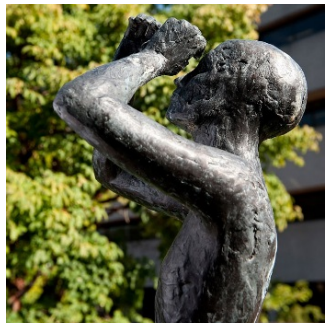


CS1117 – Introduction to Programming

Dr. Jason Quinlan,

School of Computer Science and Information Technology

A TRADITION OF INDEPENDENT THINKING

UCC

University College Cork, Ireland

Coláiste na hOllscoile Corcaigh

What is programming?

On Dictionary.com, it says:

- the process of writing computer programs

Not helpful, so let's search:

- "the process of writing computer programs"

On Dummies.com, it says:

- "Before you start writing a computer program, first take four critical steps to design it."

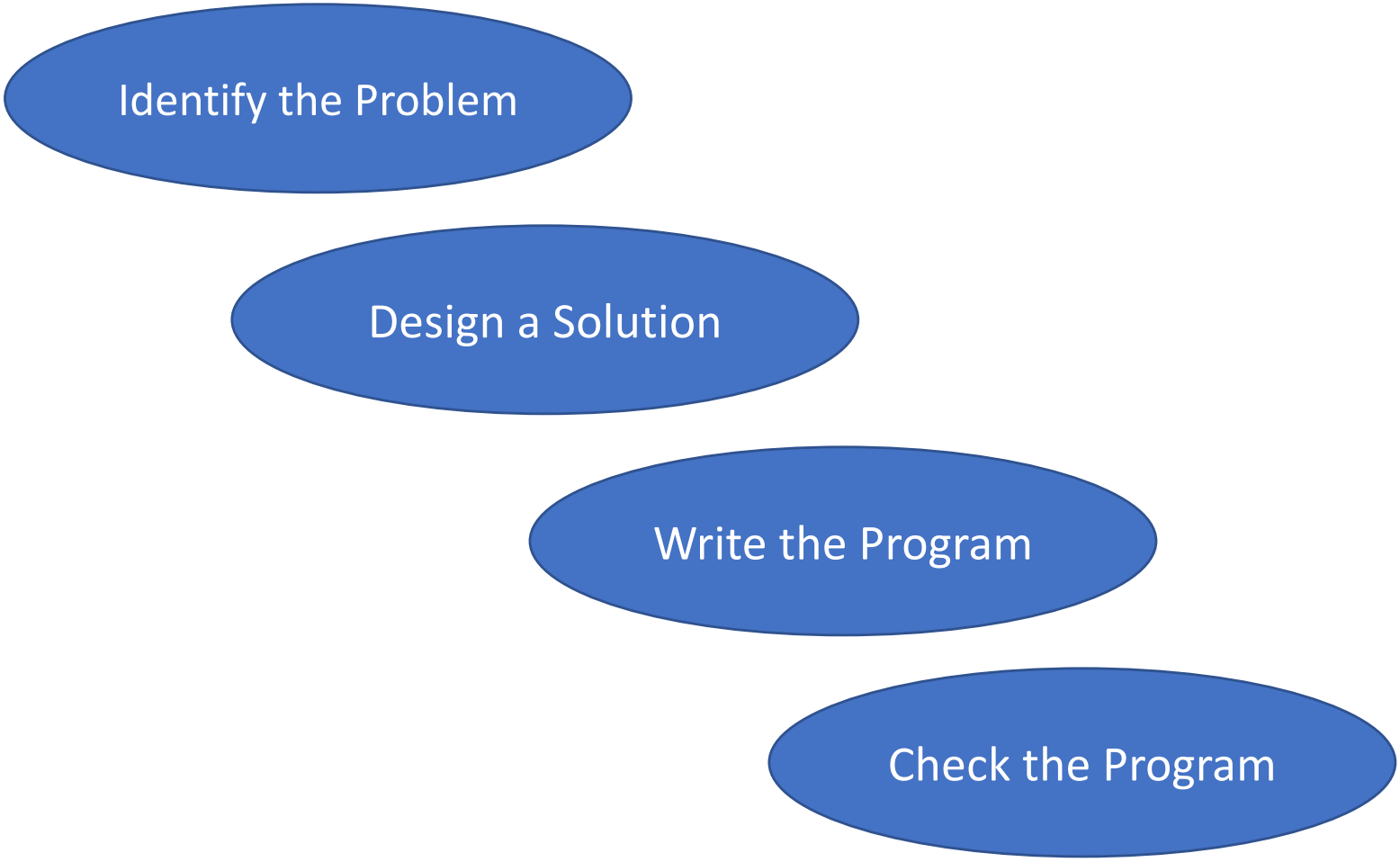
Now we're getting somewhere

What is programming?

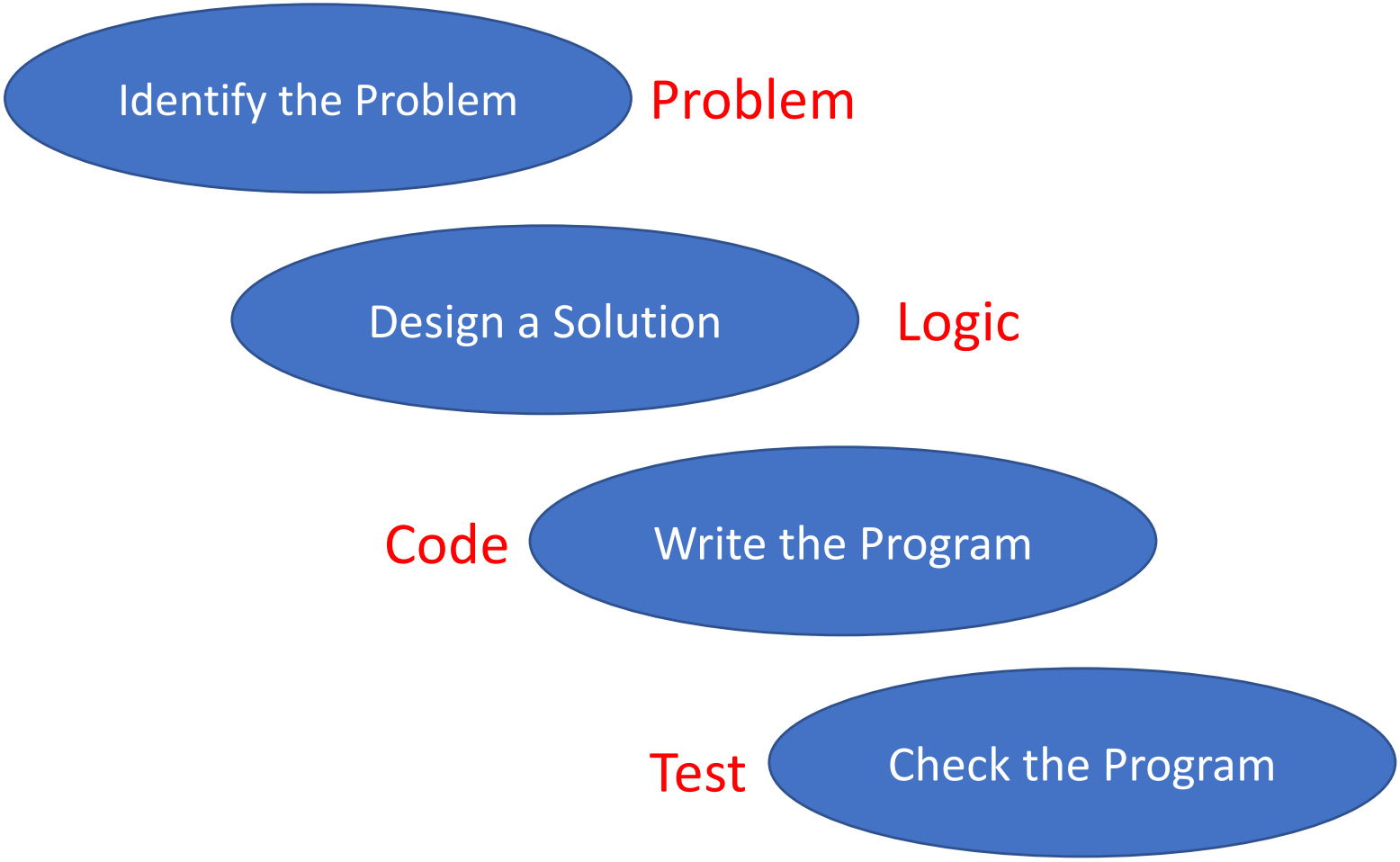
On Dummies.com, the four steps to design are:

- Identify the problem:
  - Every program solves a problem, or at least provides clarity to a part of the problem
  - If you don't identify all of the problem, we get BUGS
- Identify the user:
  - Will the person be technical, admin, maintenance, muggle….
  - Does the user just need an interface or demand low level admin rights for on the fly modifications
- Determine the target device (computer, tablet, phone, etc.):
  - iOS, Android, Apple, Google, Web Server, etc.
  - Different devices use different languages: iOS (Swift), Web (HTML), Google (golang)
- Determine your programming skills:
  - Who's going to write the program. Small team, large team
  - Small problem, large problem

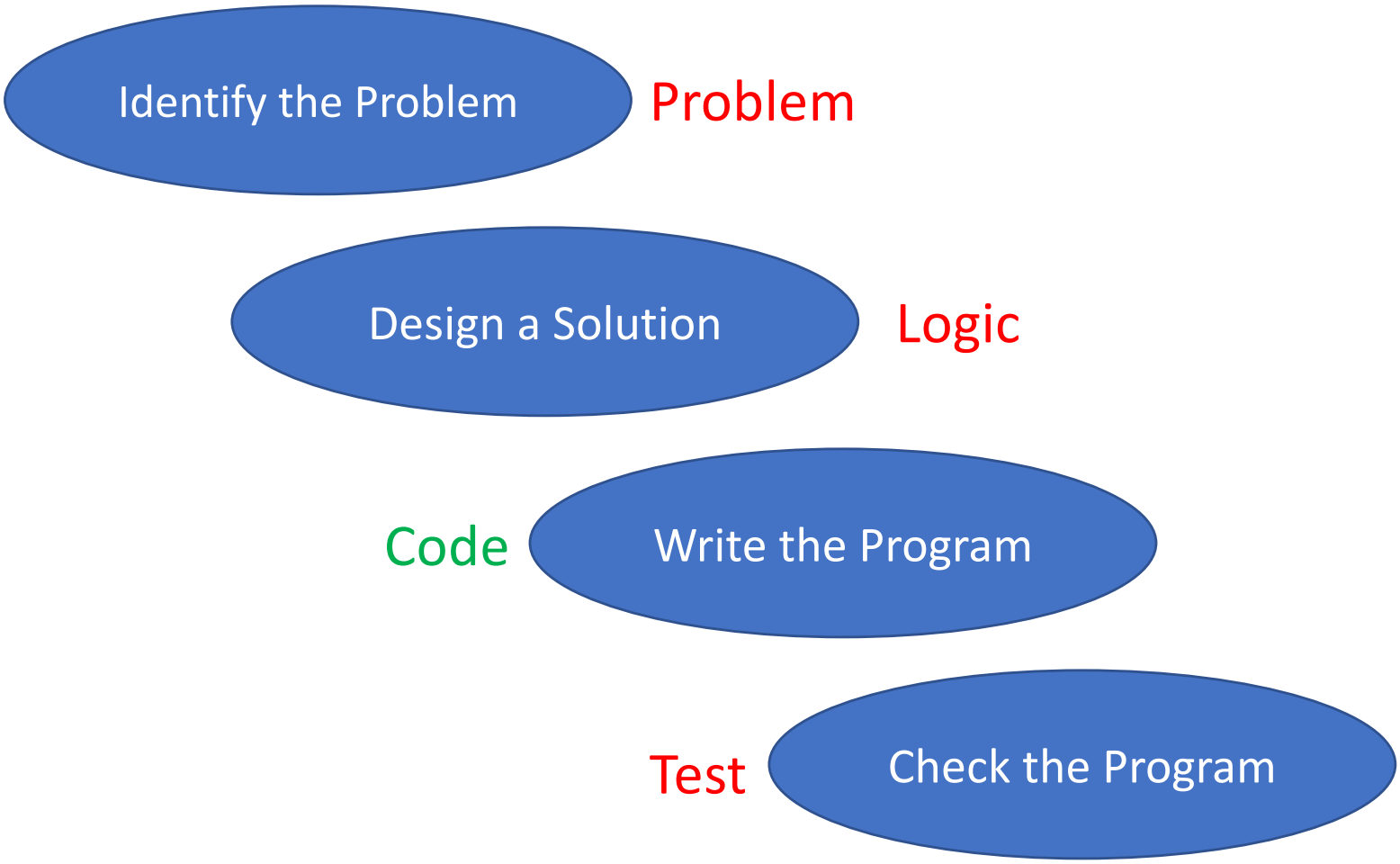
4 Pillars of Programming



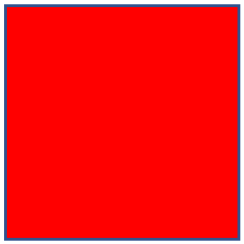
4 Pillars of Programming



4 Pillars of Programming



What is a program?



- A program is a set of instructions given to a computer written in a certain language

- **Application Programs**
  - Web applications e.g. Dropbox, Google Maps, etc.
  - Local host applications e.g. Word, Excel
  - Games on mobile devices e.g. PUBG, Clash of Clans, Fortnite…

- **System Programs**
  - Device drivers e.g. video/audio driver
  - An entire Operating System e.g. Windows 7
  - Plans of the Death Star

What is a program?

- **Categories of Computer Languages:**

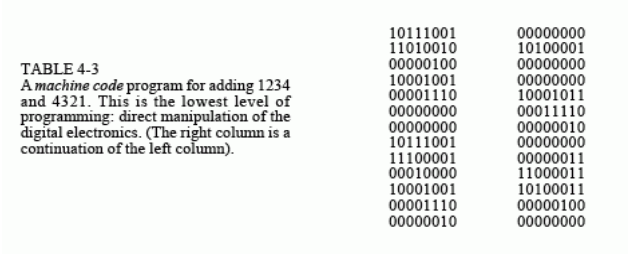


TABLE 4-3
A *machine code* program for adding 1234 and 4321. This is the lowest level of programming: direct manipulation of the digital electronics. (The right column is a continuation of the left column).

10111001	00000000
11010010	10100001
00000100	00000000
10001001	00000000
00001110	10001011
00000000	00011110
00000000	00000010
10111001	00000000
11100001	00000011
00010000	11000011
10001001	10100011
00001110	00000100
00000010	00000000

  - **Machine Language**
    - Binary number codes understood by a specific CPU
  - **Assembly Language**
    - Codes that correspond one-to-one with machine language instructions – CPU specific

      assembly code:

      MOV AL, 61h ; load AL with 97 decimal (61 hex)

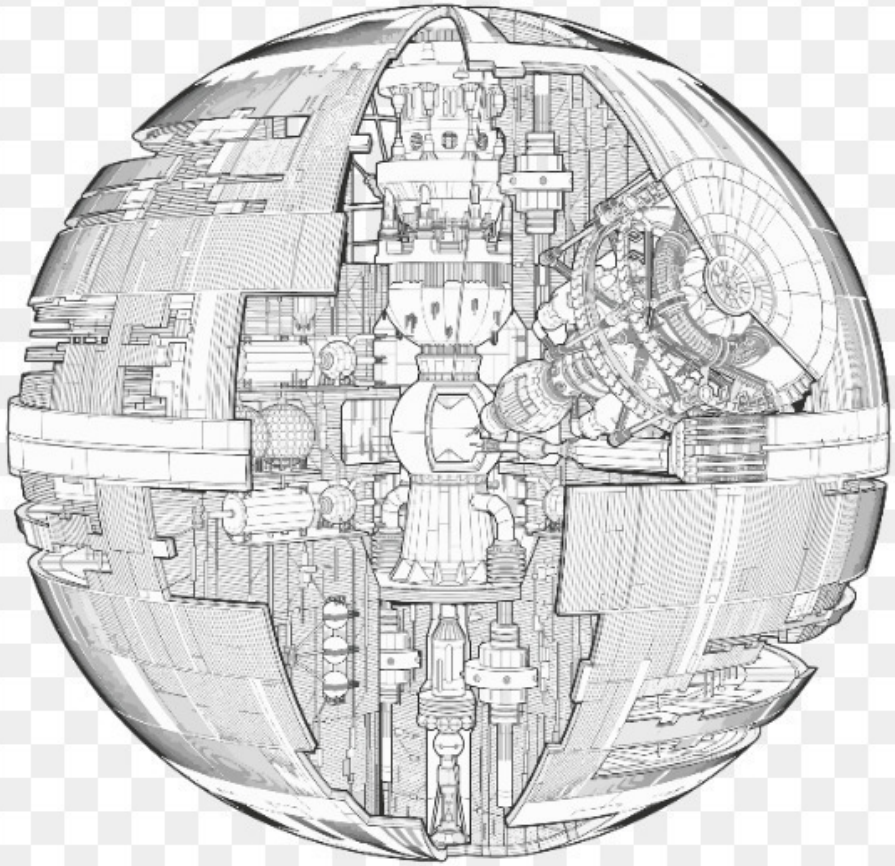
      binary code:

      10110000 01100001
  - **High Level Languages: "**Human Readable" Code
    - Machine independent programming language that combines algebraic expressions and English symbols.
    - Examples: C, C#, C++, Java, Python, Perl, PHP, Swift, Golang, etc.

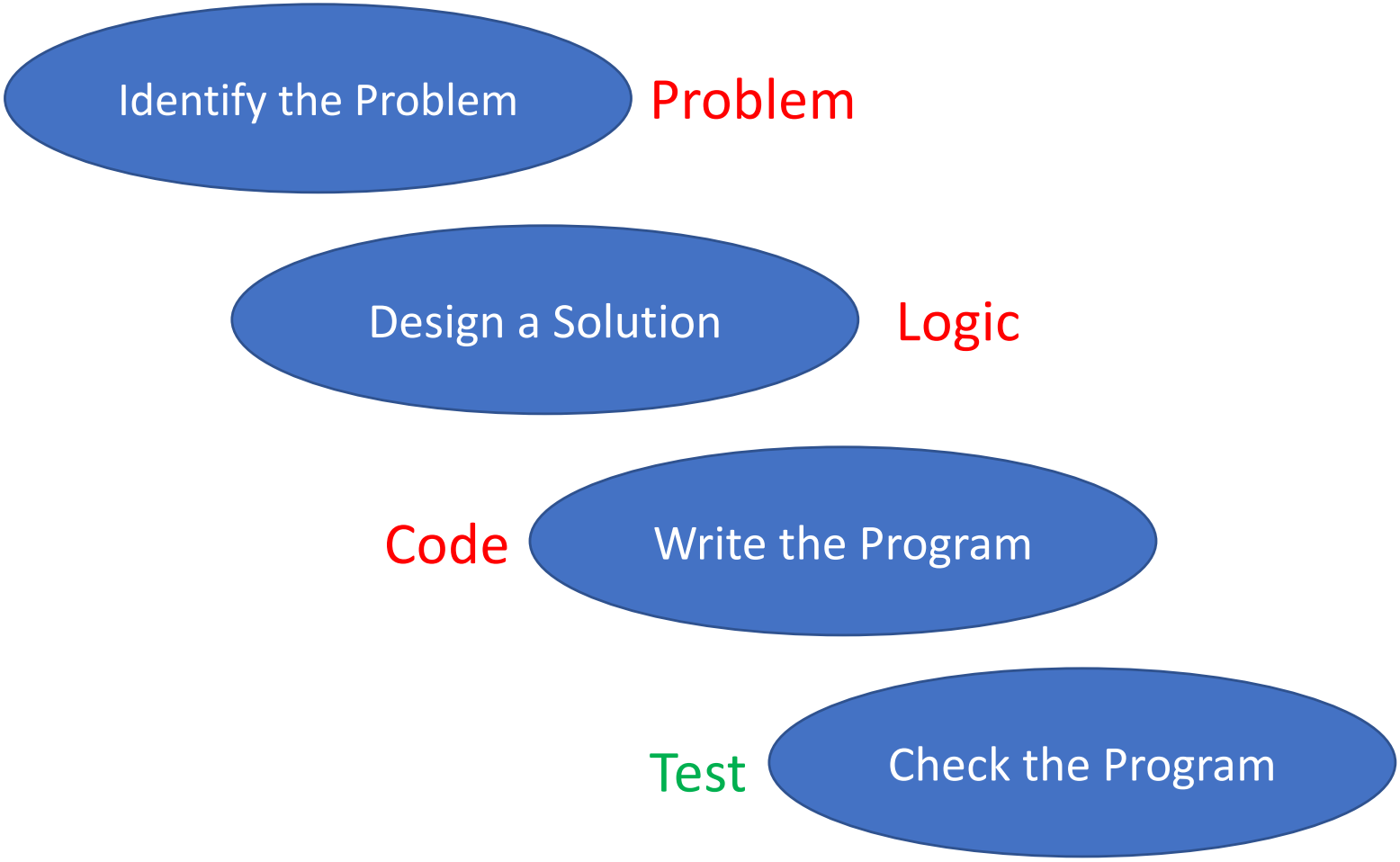
What is a program?

What is a program?

What is a program?



4 Pillars of Programming



Why do we need to test our code?

Roughly 2 kinds of errors:

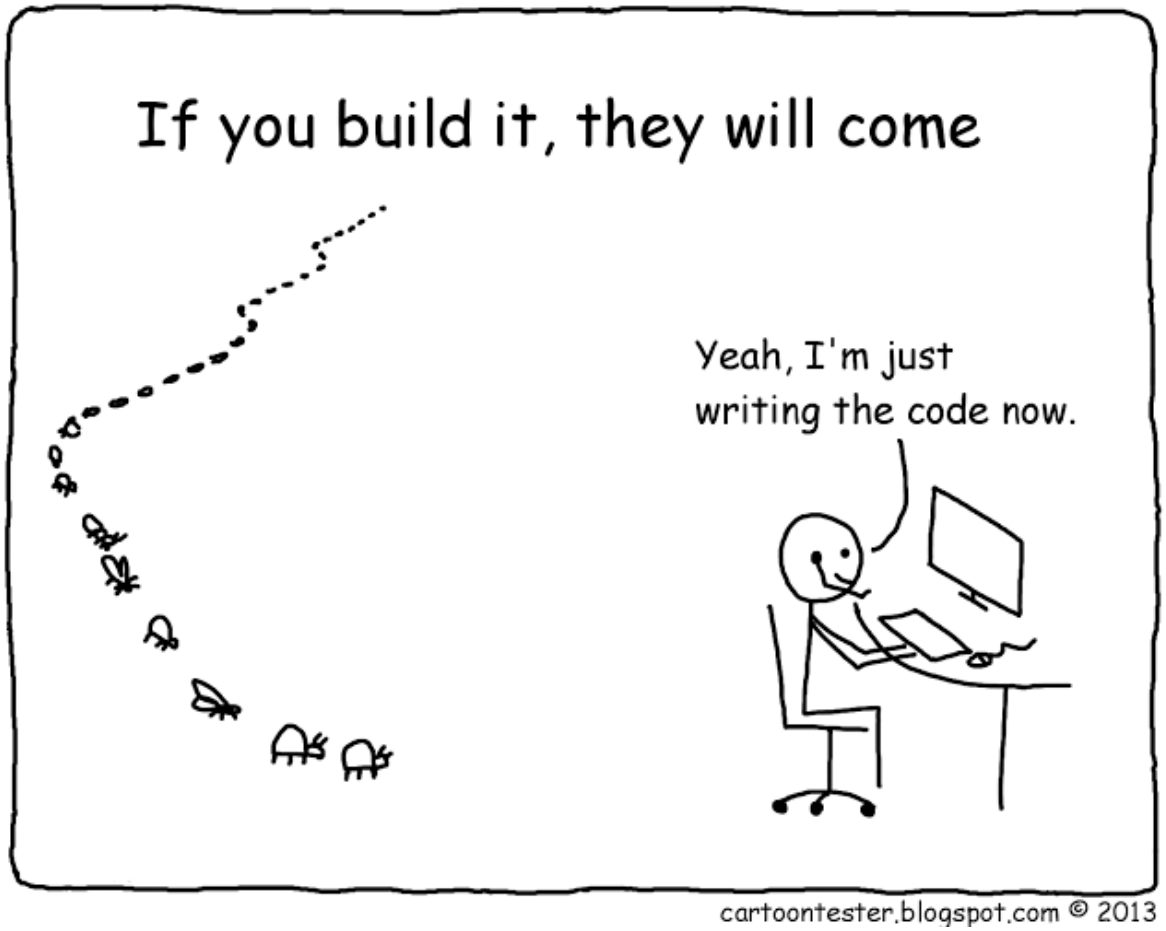
Syntactic:

- Compiler will stop program and issue alert
- Normally incorrect grammar (calling something by the wrong name)
- Calling something that doesn't exist
- Most if not all will be caught by a good IDE (Atom, PyCharm)

Semantic (logic):

- Compiler does not complain but builds and runs the code
- Does not crash or stop abnormally
- Code looks primarily fine
- But will stop if something specific happens (input, output, e.g., date)
- This is a BUG

Why do we need to test our code?



Why do we need to test our code?

- BUGS, BUGS, BUGS !!!!
- What is a bug:
  - As said, an error in the logic of the program
  - Something should go left but goes right
  - Something should go up but goes down
  - Output should save to a file but the files doesn't exist

Why do we need to test our code?

- On June 4th, 1996, the very first Ariane 5 rocket ignited its engines and began speeding away from the coast of French Guiana. 37 seconds later, the rocket flipped 90 degrees in the wrong direction, and less than two seconds later, aerodynamic forces ripped the boosters apart from the main stage at a height of 4km. This caused the self-destruct mechanism to trigger, and the spacecraft was consumed in a gigantic fireball of liquid hydrogen.
- The disastrous launch cost approximately $370m, led to a public inquiry, and through the destruction of the rocket's payload, delayed scientific research into workings of the Earth's magnetosphere for almost 4 years. The Ariane 5 launch is widely acknowledged as one of the most expensive software failures in history.

Why do we need to test our code?

- The fault was quickly identified as a software bug in the rocket's Inertial Reference System. The rocket used this system to determine whether it was pointing up or down, which is formally known as the horizontal bias, or informally as a BH value. This value was represented by a 64-bit floating variable, which was perfectly adequate.
- However, problems began to occur when the software attempted to stuff this 64-bit variable, which can represent billions of potential values, into a 16-bit integer, which can only represent 65,535 potential values. For the first few seconds of flight, the rocket's acceleration was low, so the conversion between these two values was successful. However, as the rocket's velocity increased, the 64-bit variable exceeded 65k, and became too large to fit in a 16-bit variable. It was at this point that the processor encountered an operand error, and populated the BH variable with a diagnostic value.
- **Not enough space to reach space**

Why do we need to test our code?

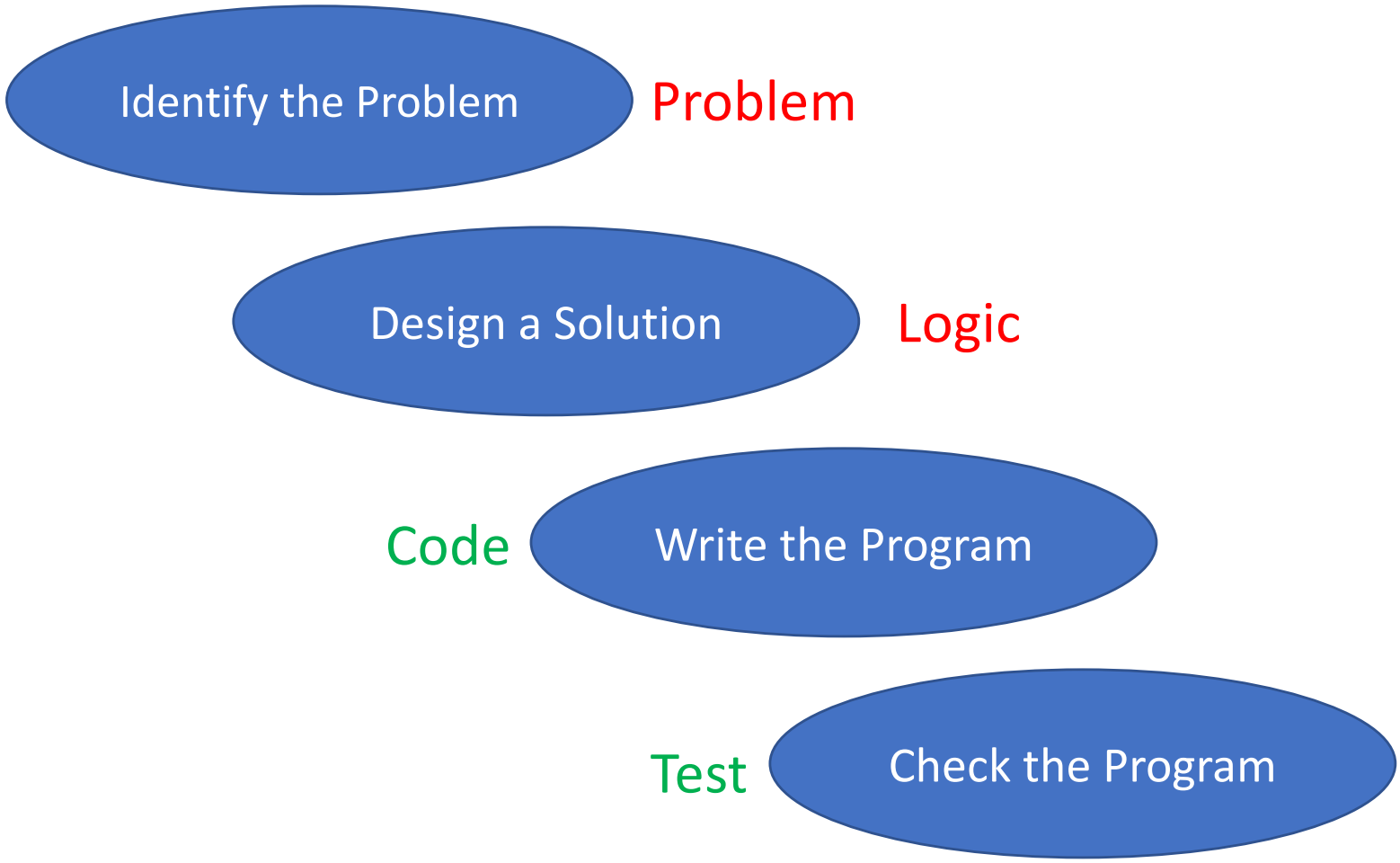
- At 1:41AM Pacific Time on Thursday, September 23, 1999, the Mars Climate Orbiter stowed its solar arrays, to protect them from a temporary descent into the upper Martian atmosphere. Nine minutes later, the craft reoriented itself using its Reaction Control systems, to line up its main engine retrograde to Mars. At 2:01AM, the engine ignited for its main burn, which initiated a 40 day stretch of complex aerobraking maneuvers in the upper Martian atmosphere, conserving precious rocket fuel. At 2:06AM, the orbiter passed behind Mars, and radio communications from Earth were blocked for a nail-biting twenty-one minutes. Once having successfully entered orbit, the satellite was supposed to relay photos of Mars to Earth for several years.
- But unfortunately, this did not happen as the craft was not on the correct trajectory. Instead, it had burned up in the Martian atmosphere nearly 24 hours earlier, lost without a trace. This is the story of the Rapid Unanticipated Disassembly of the Mars Climate Orbiter.

Why do we need to test our code?

- The first factor was a bug in the ground control software supplied by Lockheed Martin, which displayed the force for each AMD manoeuvre. The software calculated the value in an imperial unit, pound-seconds, whereas the software built by NASA expected the value to be in a metric unit, newton-seconds. As these values were not correctly converted, this led to small discrepancies in the position of the spacecraft, which compounded over a course of millions of miles.
- The second was human error. Quality Assurance had not found the use of an imperial unit in external software, despite the fact that NASA's coding standards at the time mandated use of metric units.
- In fact, for the first four months of navigation, NASA had relied on email notifications from the contractor to determine that the spacecraft was rotating. Calculations were also performed manually rather than using the supplied software, due to file format errors and miscellaneous bugs.

Why do we need to test our code?

Recap:4 Pillars of Programming



UCC
University College Cork, Ireland
Coláiste na hOllscoile Corcaigh