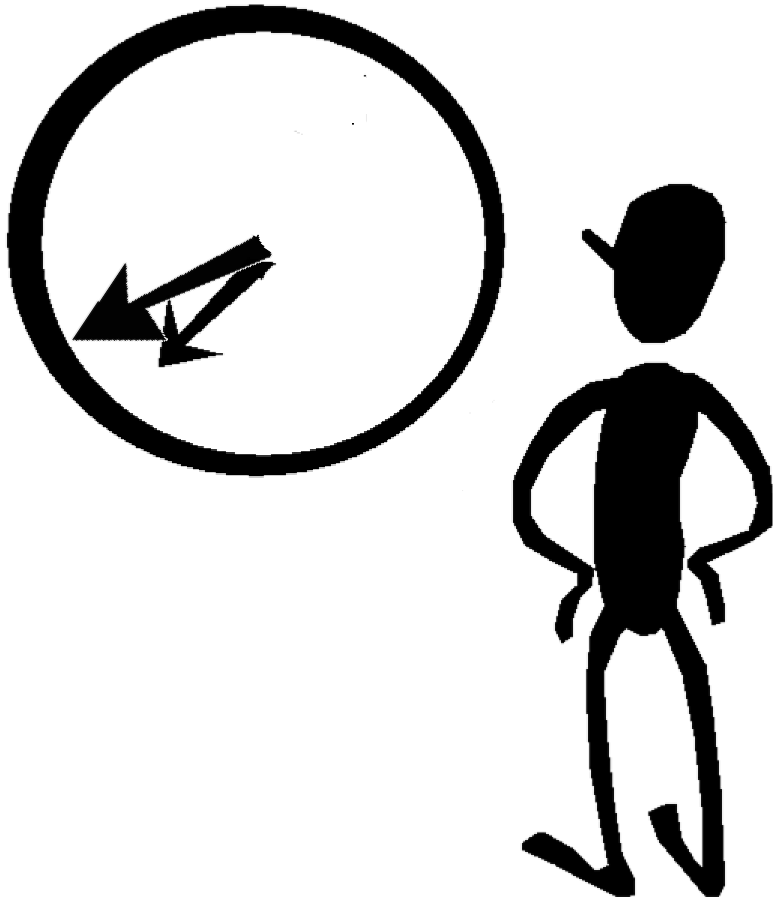
I will make you a better C# developer 2018 edition Debugging

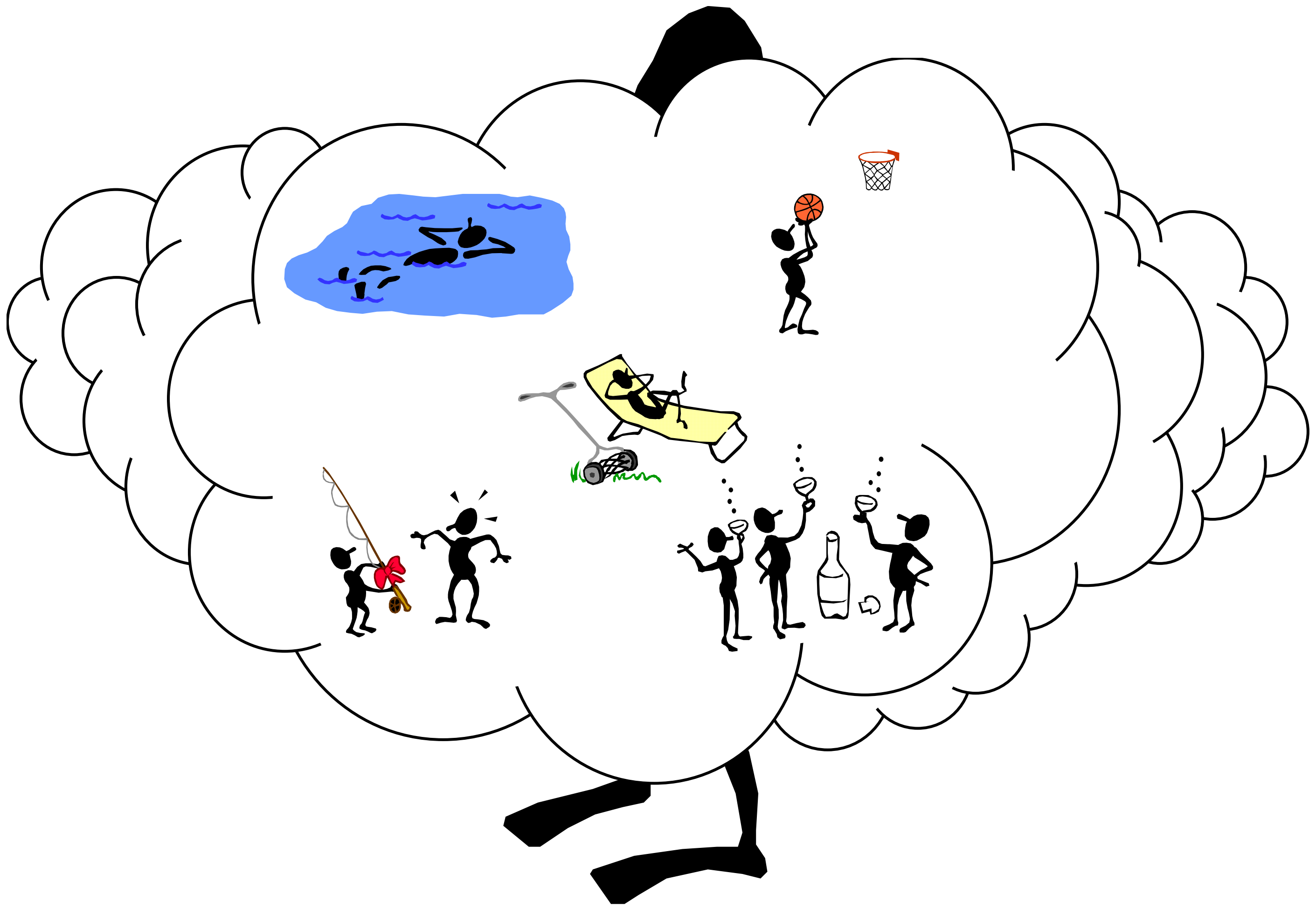
Kathleen Dollard

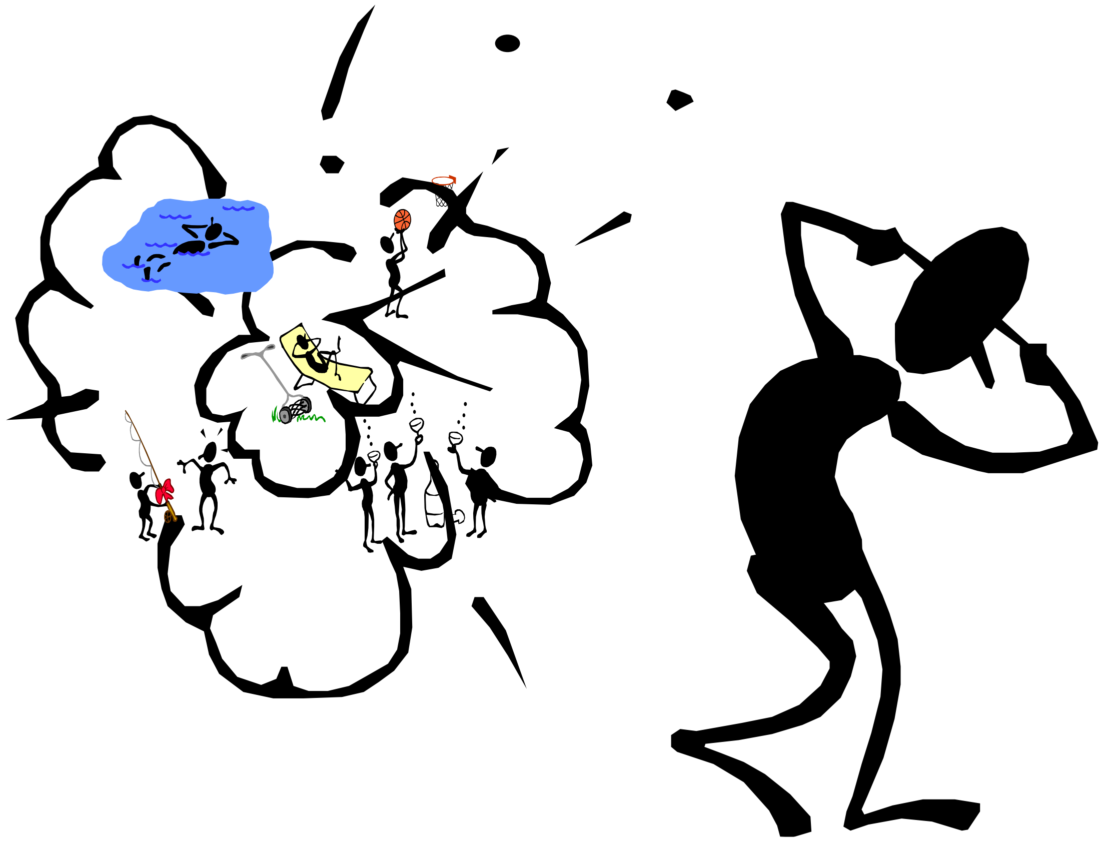
Principal Program Manager, Microsoft

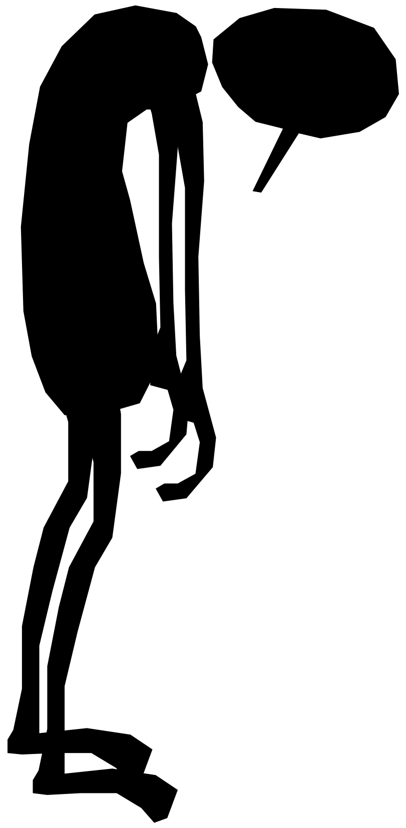
kdollard@microsoft.com

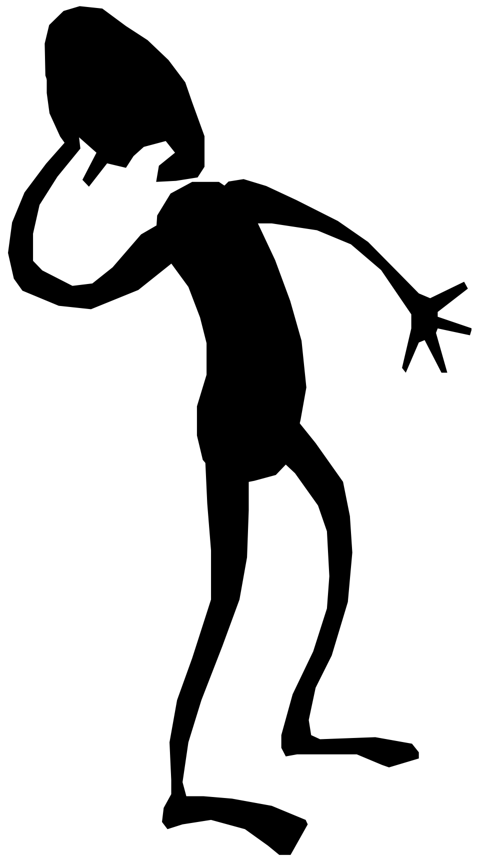
Debugging

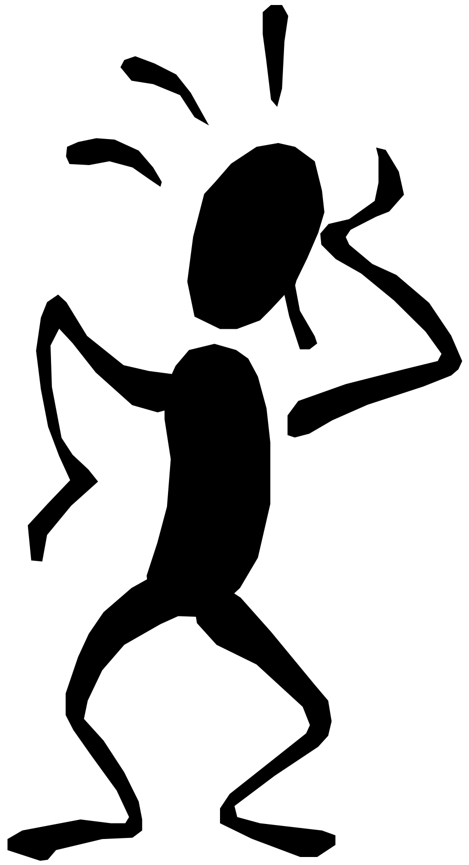


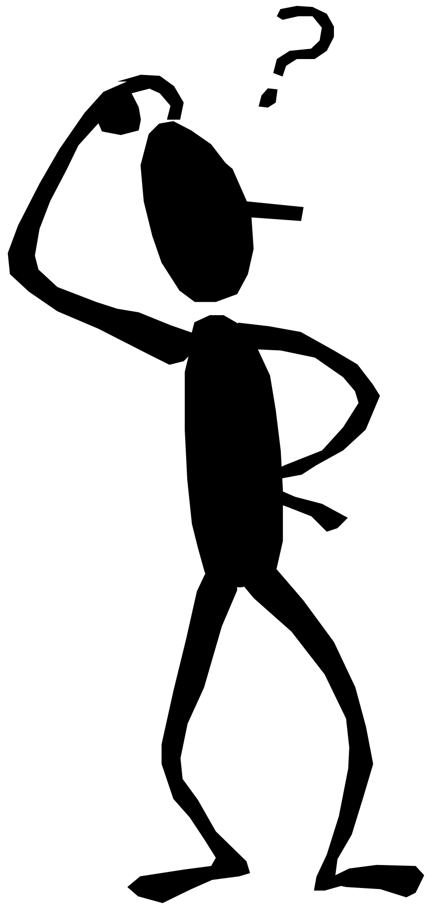












Debugging is a game of strategy.

The rules are set by the computer, your debugger, and requirements or user expectations.

You may enter the contest expecting a trivial opponent, only to find it like Hydra with two new problems sprouting for each one that you solve.

It’s a critical game because we fix bugs from the time we first check in code, and the cost of each bug tends to increase across the project lifecycle.

Bug?

Something you have to fix
in existing code

Code is broken.
Where to start?

```
public static int v1(int count)
{
    var a = 0;
    if (count == 0) return a;
    var b = 1;
    var i = 0;
    if (CheckReturn(++i, count, b)) return b;
    var val2 = a + b;
    a = b;
    b = val2;
    if (CheckReturn(++i, count, b)) return b;
    b = a + b;
    if (CheckReturn(++i, count, b)) return b;
    var val1 = b;
    b = a + val1;
    if (CheckReturn(++i, count, b)) return b;
    return default(int); // should never get here, so value doesn't matter
}

private static bool CheckReturn(int i, int count, int val)
{ return (i == count); }
```

Meaningless names

Unary increment

No bracket on return

Similar, but not identical code

Random claim in comment without guard

Valid return value on failure

Unused parameter

No value in method

Code is broken.
Where to start?

```
public static int v1(int pos)
{
    if (pos == 0) { return 0; }
    var a = 0;
    var b = 1;
    for (int i = 0; i < pos; i++)
    {
        var temp = b;
        b = b + a;
        a = temp;
    }
    return b;
}
```

Meaningless names

Meaningless names

Code is fixed.
Still not easy reading

```
public static int v1(int pos)
{
    if (pos == 0) { return 0; }
    if (pos == 1) { return 1; }
    var a = 0;
    var b = 1;
    for (int i = 1; i < pos; i++)
    {
        var temp = b;
        b = b + a;
        a = temp;
    }
    return b;
}
```

Redundant action

Non-zero iterator start

Happy Days!

```
public static int Fib(int count)
{
    if (count <=1) { return count; }
    var valPrevious = 0;
    var valThis = 1;
    // first two values returned above
    for (int i = 2; i <= count ; i++)
    {
        var temp = valThis;
        valThis = valThis + valPrevious;
        valPrevious = temp;
    }
    return valThis;
}
```

```
public static int Fib(int pos)
{
    if (pos <= 1) { return pos; }
    return Fib(pos - 2) + Fib(pos - 1);
}
```

What's wrong with this code?

Hey, it works! What's the problem?

O2^n and many iterations may overflow the stack

Code is hard to debug when you can’t read the code

Code is hard to debug when the symptom (issue)
is distant from the problem (bug)
in time or space

Code is hard to debug when the context
causing the bug is transitory

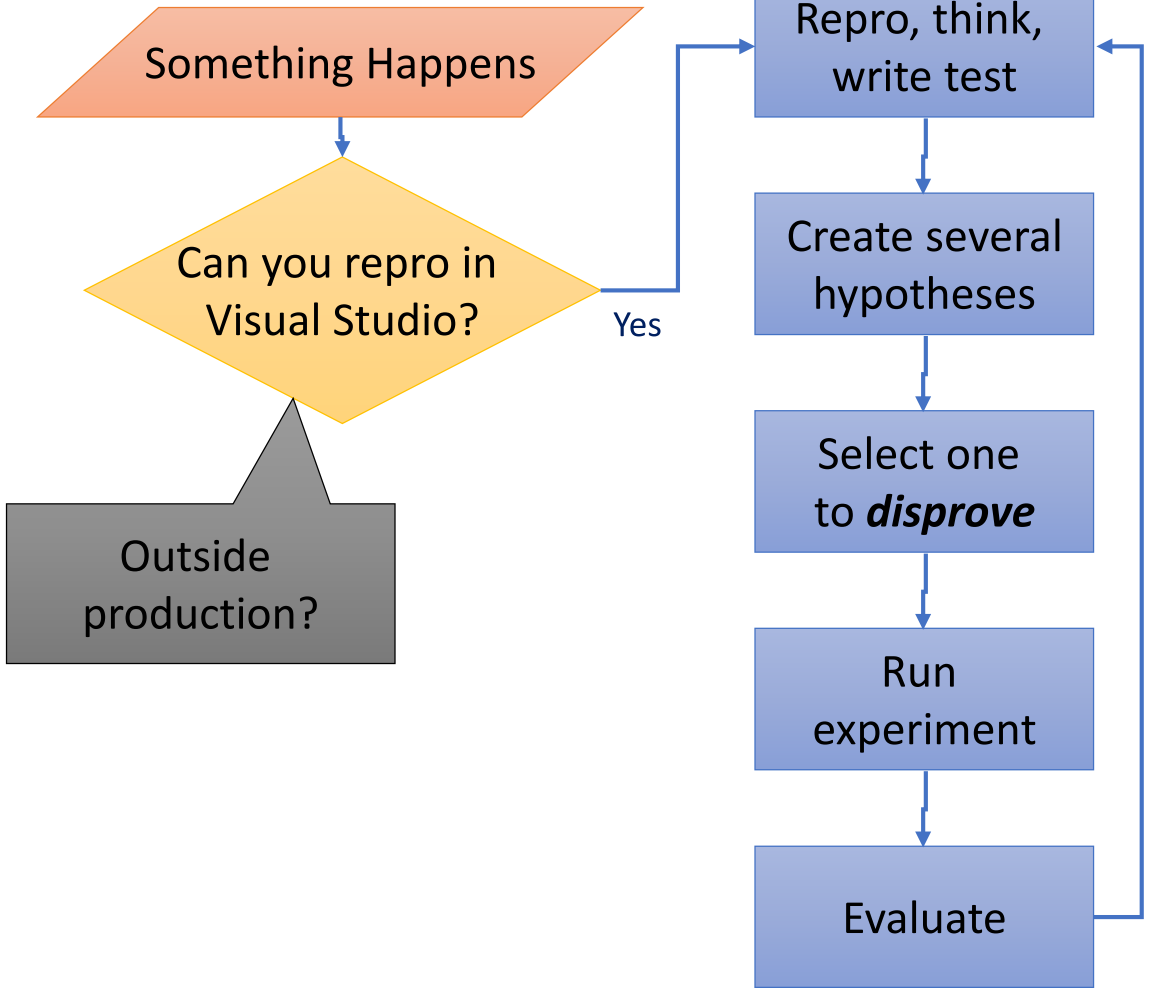
Code is hard to debug when someone gave you
incorrect or misleading information

Debugging Strategies

- **Divide and Conquer**
- **Scientific Method**

A good debugging strategy is any strategy you can articulate and repeat

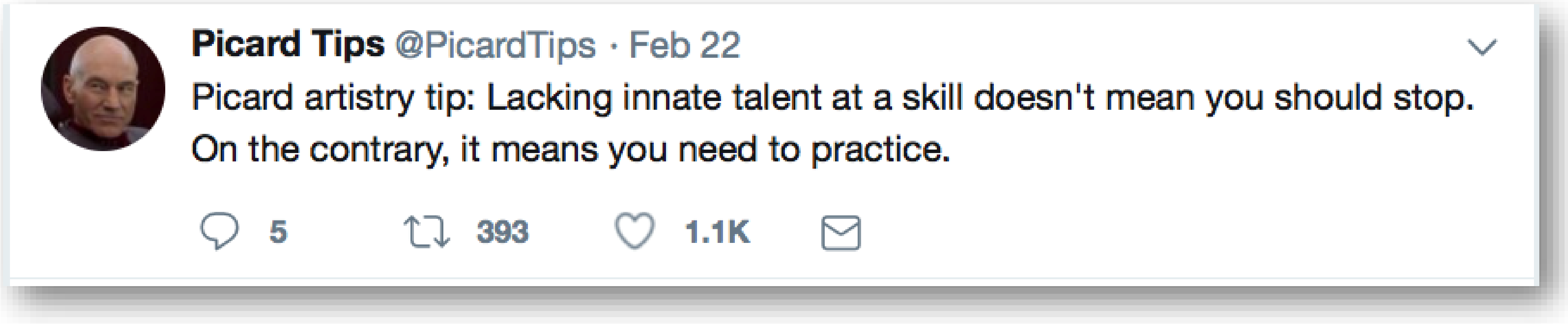
A Debugging Strategy
Scientific Method



Scientific Method Walkthrough

Collaborative Debugging Game

1. **One person imagines a bug. Be VERY specific. Know exactly the broken code and exactly what would happen**
    1. Helps if it's a bug you've stumbled with
2. **Everyone else works together to solve the bug by asking what would happen if they ran certain VERY specific tests**
3. **The person imagining the bug may well make mistakes. Be patient. You'll also make mistakes and go down rabbit holes in the real world.**
4. **Take turns imagining the bug**



Picard Tips @PicardTips · Feb 22

Picard artistry tip: Lacking innate talent at a skill doesn't mean you should stop. On the contrary, it means you need to practice.

5 393 1.1K

Break