```
// Killing Cancer with Code
// Richard Popple

isCancerKilled = false;
while ( !isCancerKilled )
{
    isCancerKilled = cancer.Kill();
}
```

Why scripting?

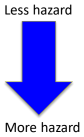
- Automate planning tasks
- Automate QA
- Data mining

Script types

- Macros
- Read only
- Full read/write

Script types

- Macros
- Read only
- Full read/write


Less hazard

More hazard

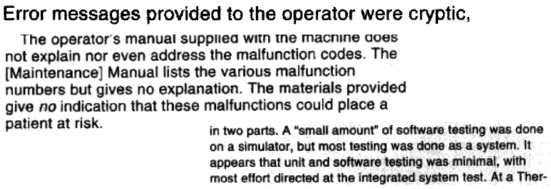
QA and safety

- Wild West
- No AAPM guidance
- Follow general standards of practice

Clinical scripts are the same as other clinical software

- Acceptance testing
- Commissioning
- Training and documentation
- Routine QA
- Version validation
  - New version of script
  - New version of host API/TPS

The Therac-25: every physicist’s nightmare



Error messages provided to the operator were cryptic,

The operator’s manual supplied with the machine does not explain nor even address the malfunction codes. The [Maintenance] Manual lists the various malfunction numbers but gives no explanation. The materials provided give *no* indication that these malfunctions could place a patient at risk.

in two parts. A “small amount” of software testing was done on a simulator, but most testing was done as a system. It appears that unit and software testing was minimal, with most effort directed at the integrated system test. At a Ther-

Leveson, N. G., & Turner, C. S. (1993). An investigation of the Therac-25 accidents. Computer, 26(7), 18-41.
https://www.cs.umd.edu/class/spring2003/cmsc838p/Misc/therac.pdf

DO NOT circumvent API restrictions

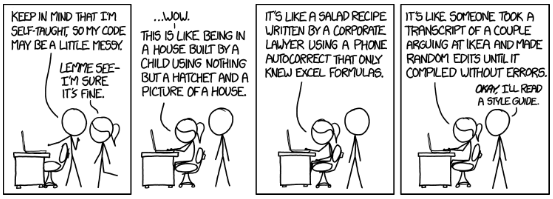
- Even if you are “smart” and “careful”

“Computer hacking refers to the practice of modifying or altering computer software and hardware to accomplish a goal that is considered to be outside of the creator’s original objective.”

“…computer hacking is somewhat ambiguous and difficult to define.”

http://cyber.laws.com/hacking

Basic principles


KEEP IN MIND THAT I'M SELF-TAUGHT, SO MY CODE MAY BE A LITTLE MESSY.
LEMME SEE– I'M SURE IT'S FINE.
...WOW.
THIS IS LIKE BEING IN A HOUSE BUILT BY A CHILD USING NOTHING BUT A HATCHET AND A PICTURE OF A HOUSE.
IT'S LIKE A SALAD RECIPE WRITTEN BY A CORPORATE LAWYER USING A PHONE AUTOCORRECT THAT ONLY KNEW EXCEL FORMULAS.
IT'S LIKE SOMEONE TOOK A TRANSCRIPT OF A COUPLE ARGUING AT IKEA AND MADE RANDOM EDITS UNTIL IT COMPILED WITHOUT ERRORS.
OKAY, I'LL READ A STYLE GUIDE.

Basic principles

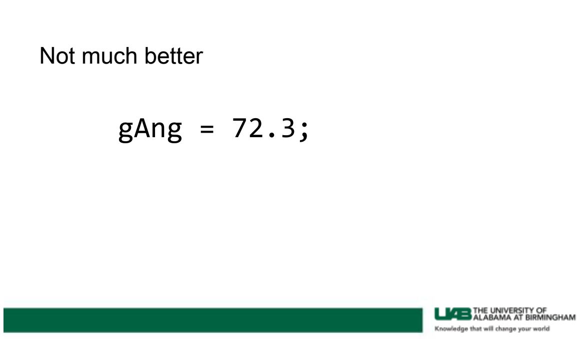
- Good variable naming
- Avoid global variables like poison
- Comments
- Short functions
- Documentation
- Tests

Basic principles

- https://www.python.org/dev/peps/pep-0008/
- https://msdn.microsoft.com/library/ff926074.aspx
- https://msdn.microsoft.com/library/ms229045(v=vs.100).aspx

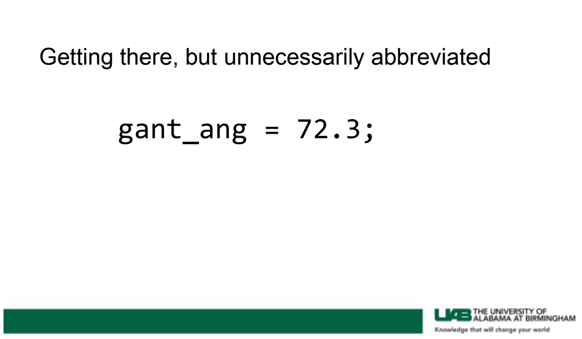
Cryptic

```
g = 72.3;
```



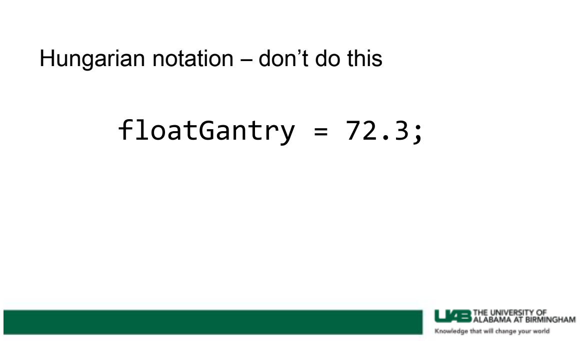
Not much better

```
gAng = 72.3;
```



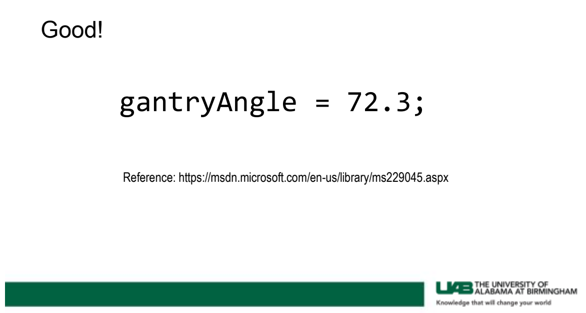
Getting there, but unnecessarily abbreviated

```
gant_ang = 72.3;
```



Hungarian notation – don’t do this

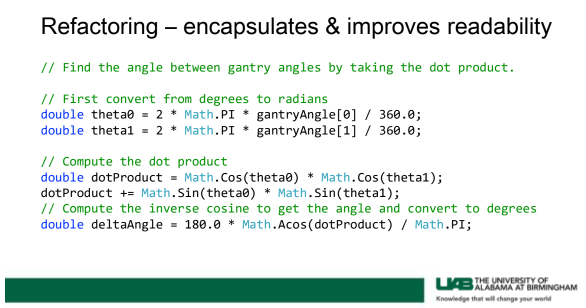
```
floatGantry = 72.3;
```



Good!

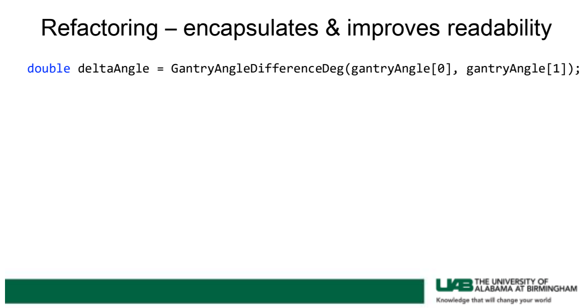
```
gantryAngle = 72.3;
```

Reference: https://msdn.microsoft.com/en-us/library/ms229045.aspx



Refactoring – encapsulates & improves readability

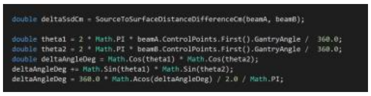
```
// Find the angle between gantry angles by taking the dot product.

// First convert from degrees to radians
double theta0 = 2 * Math.PI * gantryAngle[0] / 360.0;
double theta1 = 2 * Math.PI * gantryAngle[1] / 360.0;

// Compute the dot product
double dotProduct = Math.Cos(theta0) * Math.Cos(theta1);
dotProduct += Math.Sin(theta0) * Math.Sin(theta1);
// Compute the inverse cosine to get the angle and convert to degrees
double deltaAngle = 180.0 * Math.Acos(dotProduct) / Math.PI;
```



Refactoring – encapsulates & improves readability

```
double deltaAngle = GantryAngleDifferenceDeg(gantryAngle[0], gantryAngle[1]);
```

And now what you came for!



```
double deltaSsdCm = SourceToSurfaceDistanceDifferenceCm(beamA, beamB);

double theta1 = 2 * Math.PI * beamA.ControlPoints.First().GantryAngle /  360.0;
double theta2 = 2 * Math.PI * beamB.ControlPoints.First().GantryAngle /  360.0;
double deltaAngleDeg = Math.Cos(theta1) * Math.Cos(theta2);
deltaAngleDeg += Math.Sin(theta1) * Math.Sin(theta2);
deltaAngleDeg = 360.0 * Math.Acos(deltaAngleDeg) / 2.0 / Math.PI;
```