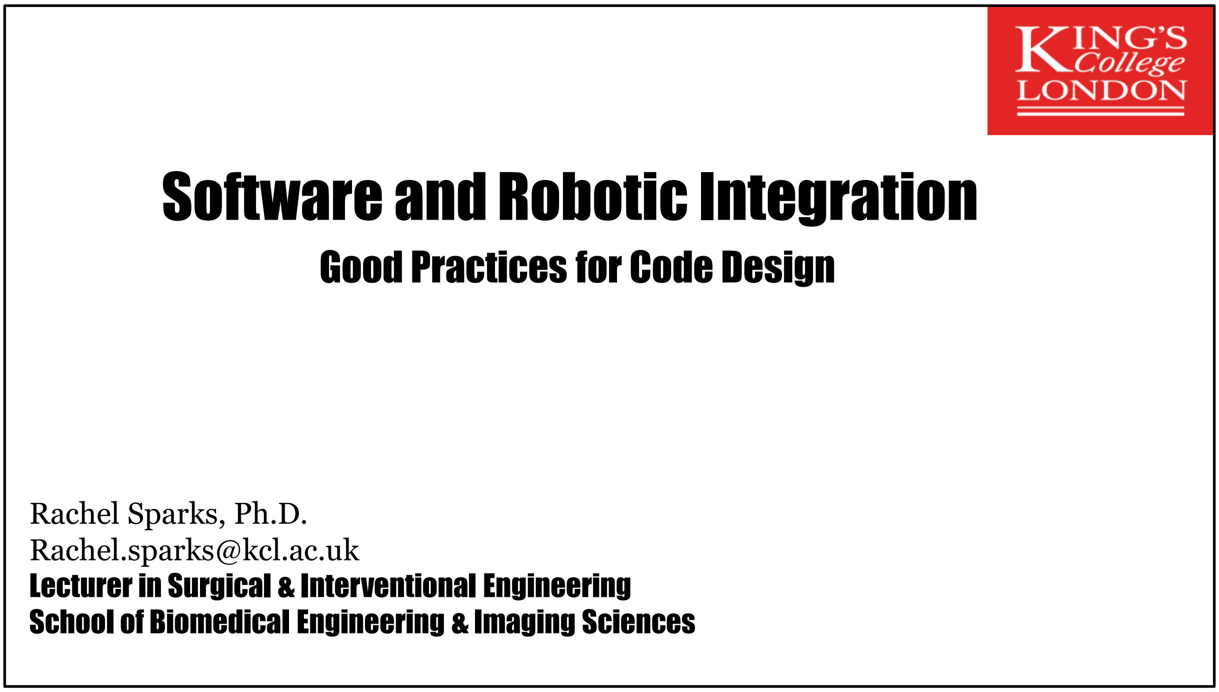


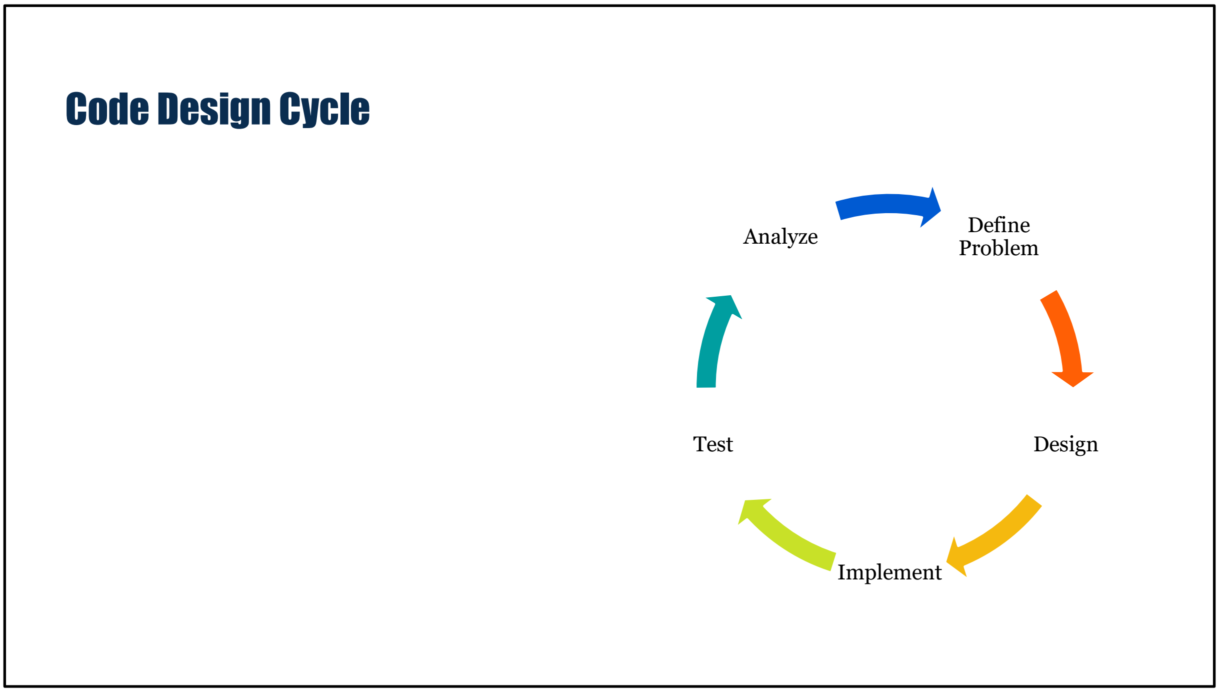
Software and Robotic Integration

Good Practices for Code Design

Rachel Sparks, Ph.D.
Rachel.sparks@kcl.ac.uk
Lecturer in Surgical & Interventional Engineering
School of Biomedical Engineering & Imaging Sciences

Code Design

- Systematic approach to design & implementing code
  - There are many possible methods – Agile, Waterfall, Scrum, Lean, Extreme
  - Some differences, but we are going to focus on the similarities
- Common trends
  - Decouple design (*how* code achieve aim) from implementation (specifics of *what* code does)



Code Design Cycle

Analyze

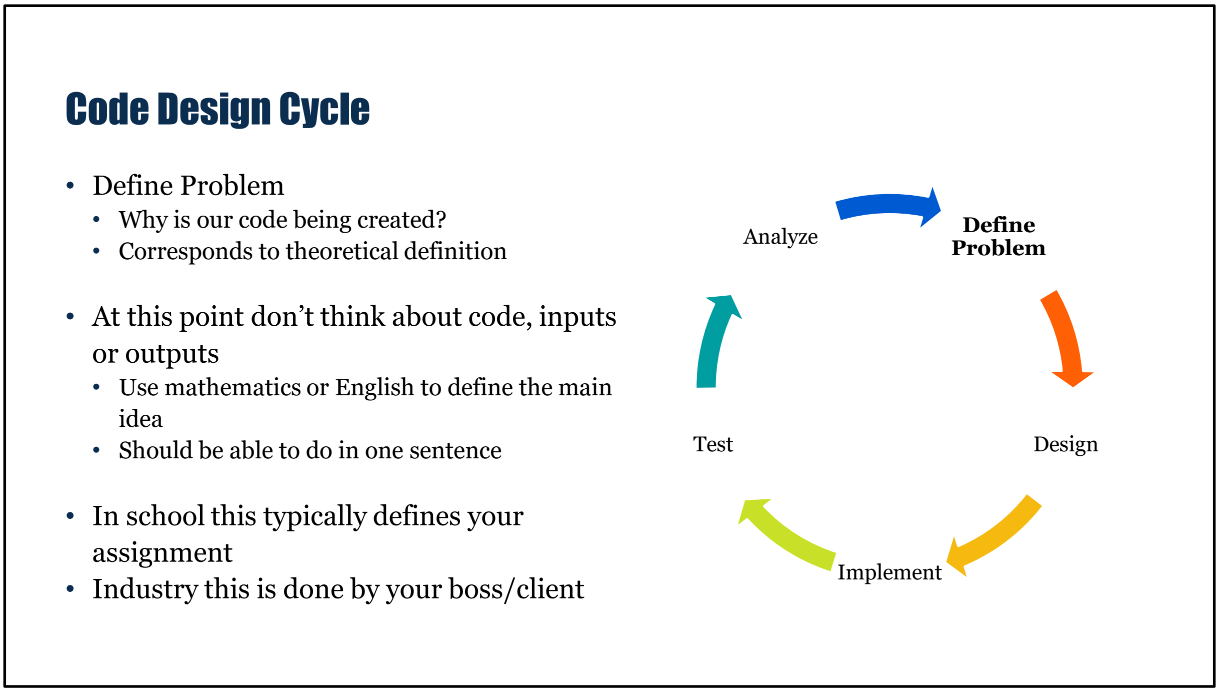
Define Problem

Design

Implement

Test

Have them define the problem theoretically



Code Design Cycle

- Define Problem
  - Why is our code being created?
  - Corresponds to theoretical definition
- At this point don't think about code, inputs or outputs
  - Use mathematics or English to define the main idea
  - Should be able to do in one sentence
- In school this typically defines your assignment
- Industry this is done by your boss/client

Define Problem

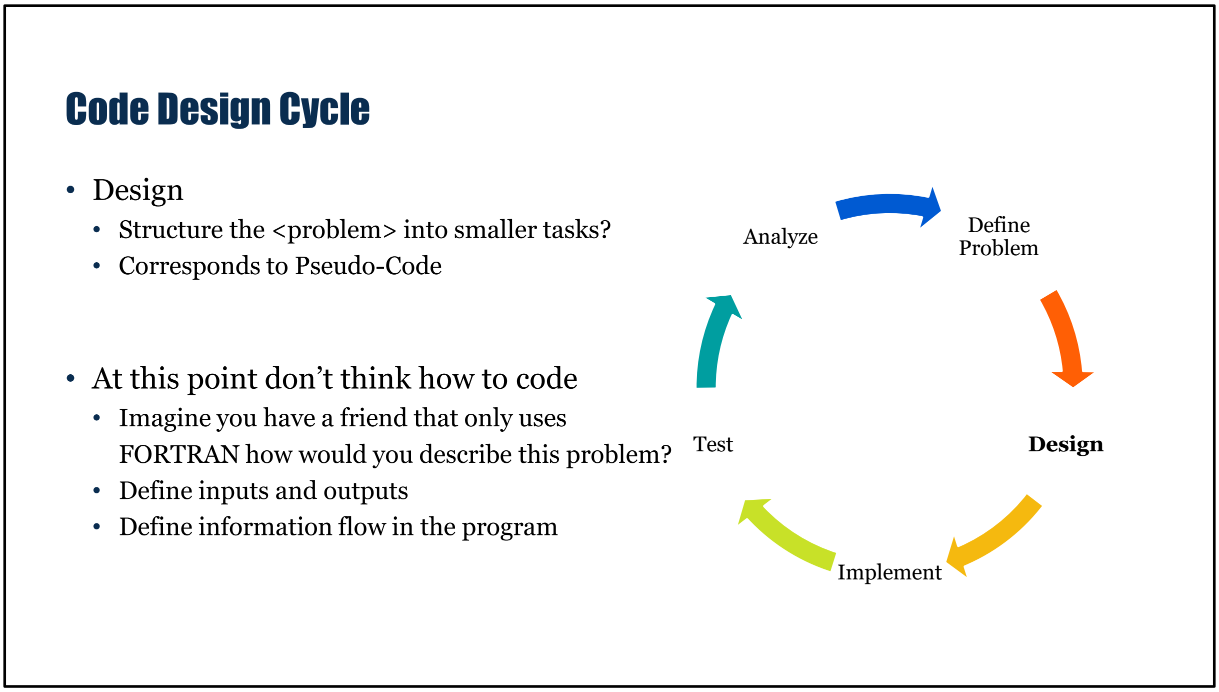
Design

Implement

Test

Analyze

The "why" we are coding -> "why" is this code going to solve a real world problem/automate an important task etc.



Code Design Cycle

- Design
  - Structure the <problem> into smaller tasks?
  - Corresponds to Pseudo-Code
- At this point don't think how to code
  - Imagine you have a friend that only uses FORTRAN how would you describe this problem?
  - Define inputs and outputs
  - Define information flow in the program

Analyze

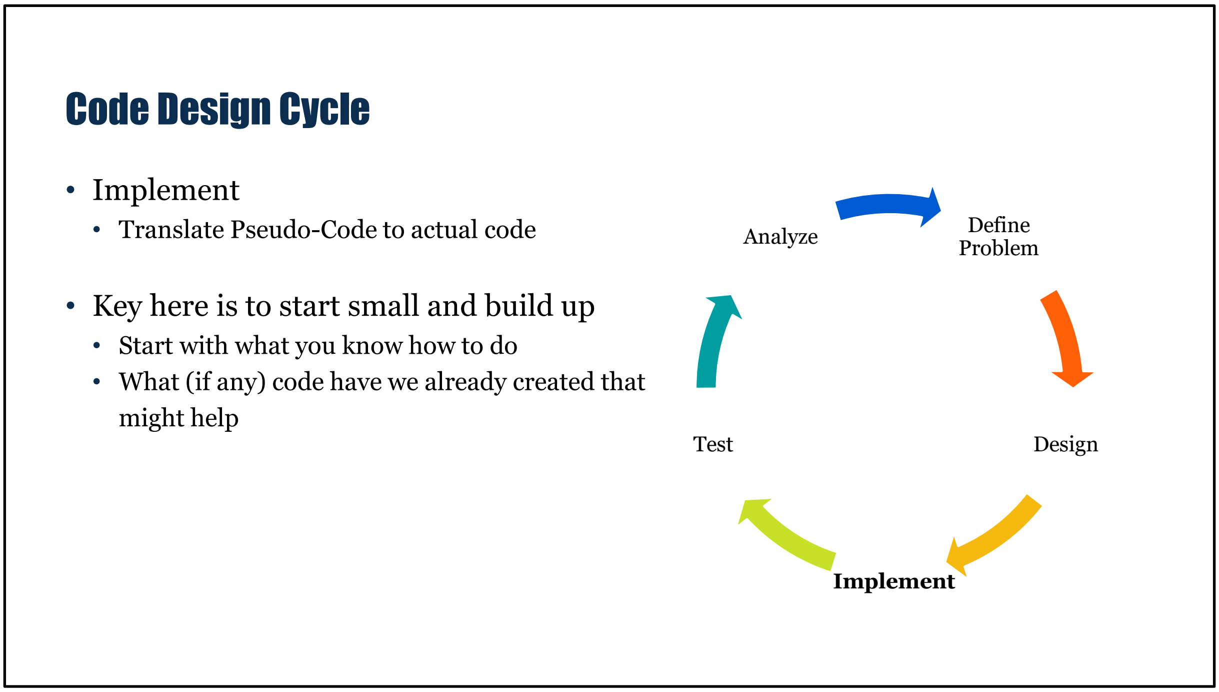
Define Problem

Design

Implement

Test

The "how" -> how are we going to solve the problem, does not need to be line by line but with enough granular detail we know which individual components we need and how they should be interacting



Code Design Cycle

- Implement
  - Translate Pseudo-Code to actual code
- Key here is to start small and build up
  - Start with what you know how to do
  - What (if any) code have we already created that might help

Analyze

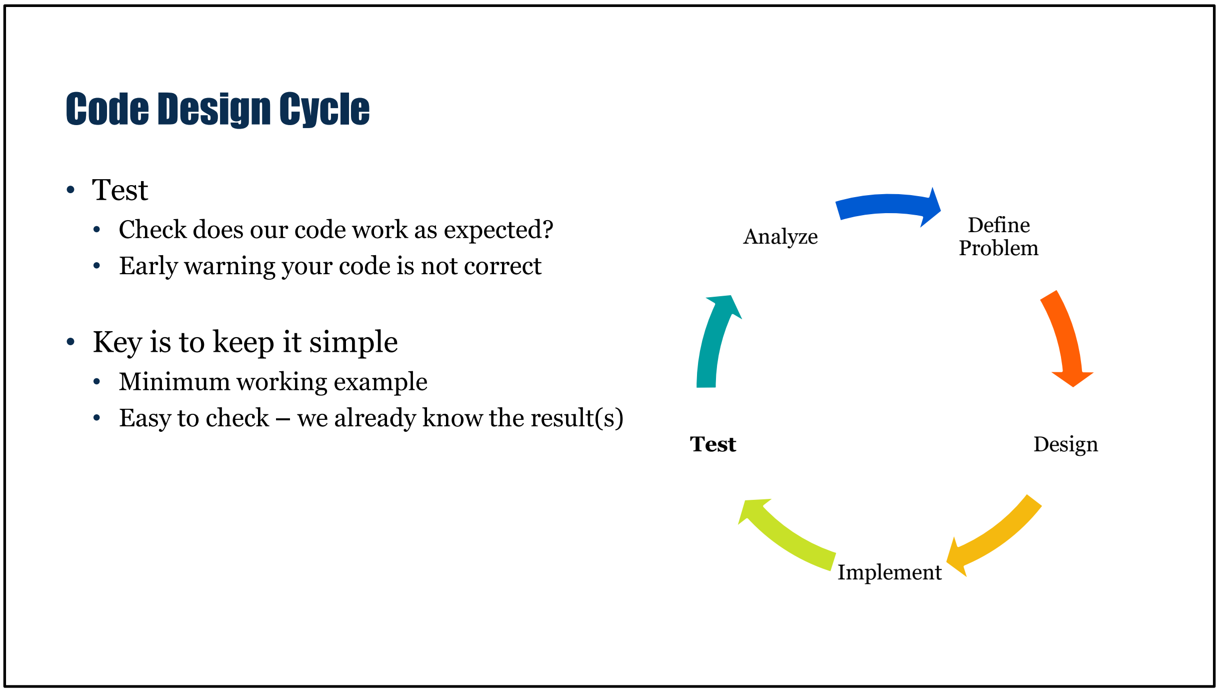
Define Problem

Design

Implement

Test

“What” -> once we know how we want to solve our problem we need to define exactly what the code will do to achieve our design.



Code Design Cycle

- Test
  - Check does our code work as expected?
  - Early warning your code is not correct
- Key is to keep it simple
  - Minimum working example
  - Easy to check – we already know the result(s)

Analyze

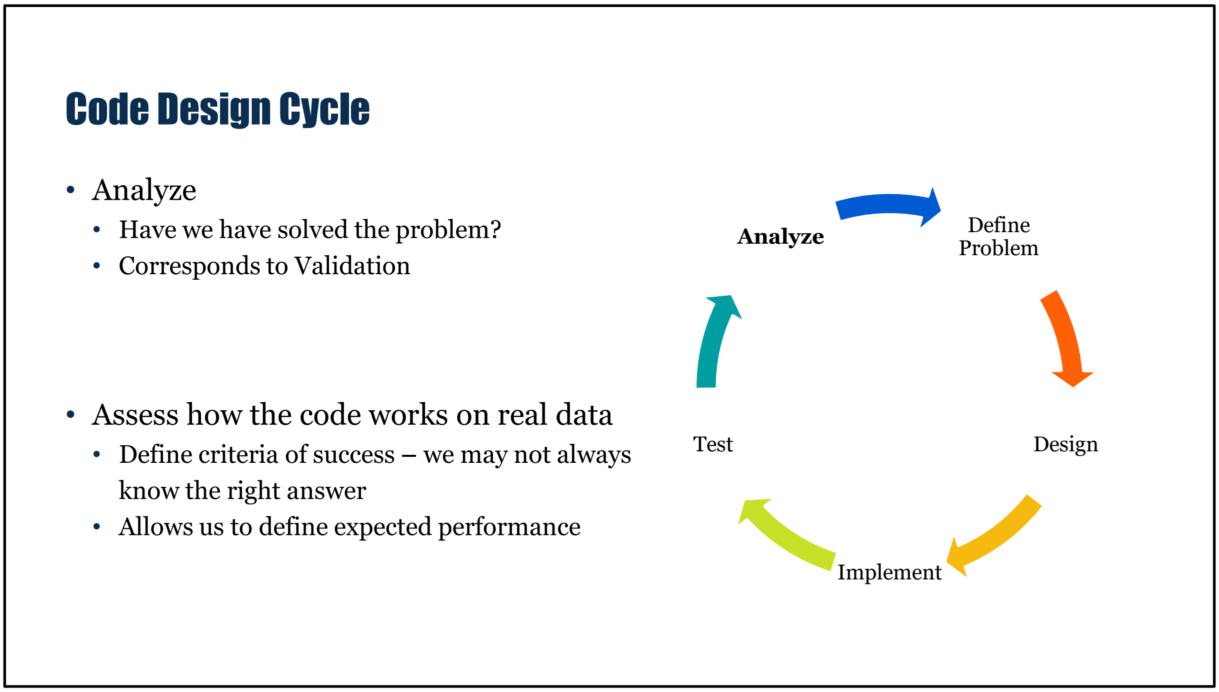
Define Problem

Design

Implement

Test

Is it working -> always try to define a very simple test that you can easily check. For instance give the program one positive example (code should work as expected and give a known output), one negative example (code should fail or this is an edge case where the result should be a known, but invalid output), and a few nonsense examples (for instance an empty input, a datatype you are not expecting, etc.)



Code Design Cycle

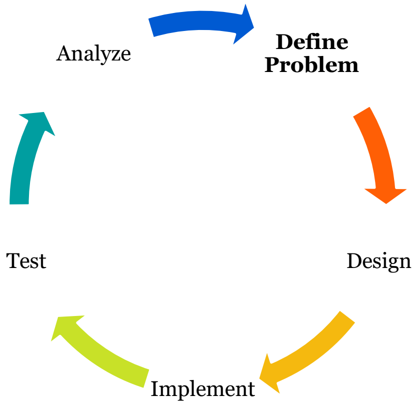
- Analyze
  - Have we have solved the problem?
  - Corresponds to Validation
- Assess how the code works on real data
  - Define criteria of success – we may not always know the right answer
  - Allows us to define expected performance

Analyze
Define Problem
Design
Implement
Test

This can be where we bring in edge cases – where the output is not known ahead of time but we want to see if the algorithm would work under a special scenario or on real world data (that might have noise or other “problems” not introduced during testing)

Code Design Cycle

- Define Problem
  - Rescope if your analysis shows your implementation is not good enough
  - Move onto another interesting problem if you have solved it



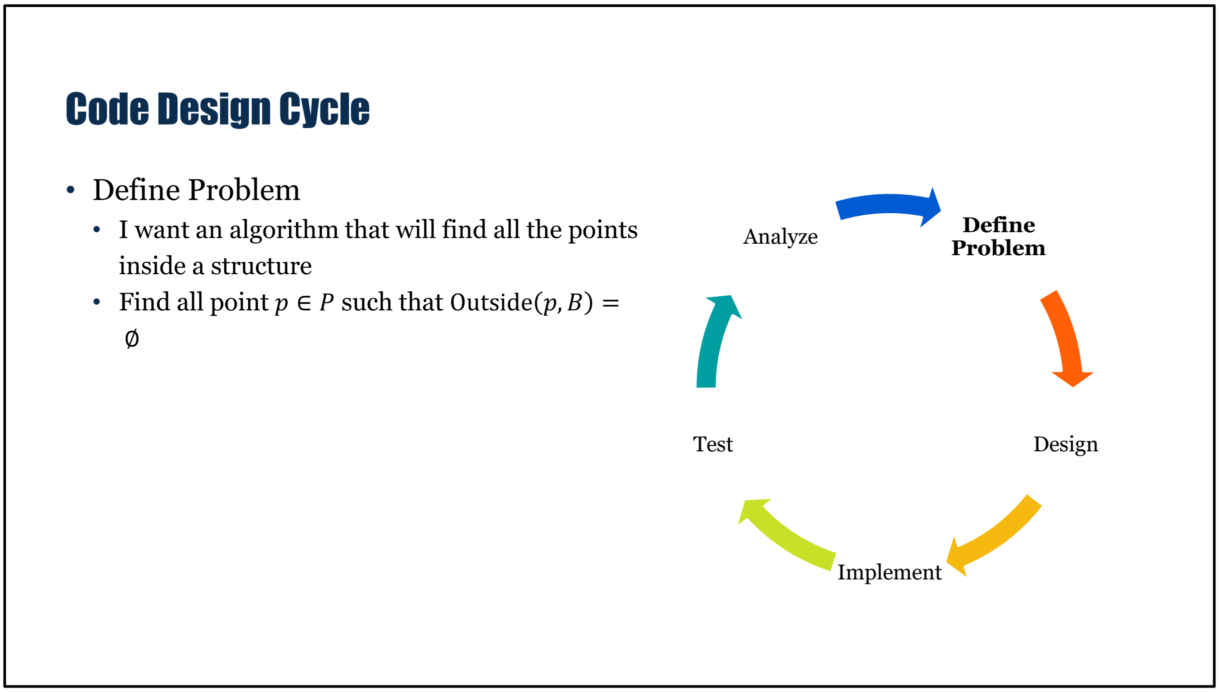
Analyze

Define Problem

Design

Implement

Test

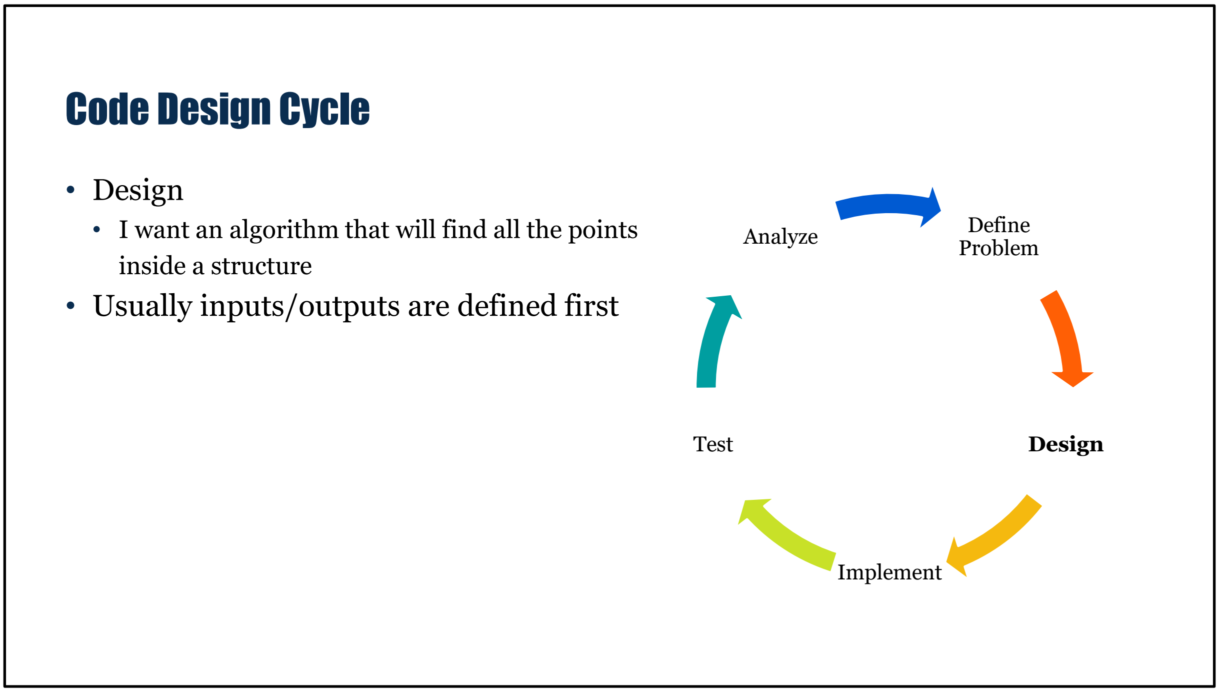


Code Design Cycle

- Define Problem
  - I want an algorithm that will find all the points inside a structure
  - Find all point $p \in P$ such that $\text{Outside}(p, B) = \emptyset$

Analyze
Define Problem
Design
Implement
Test

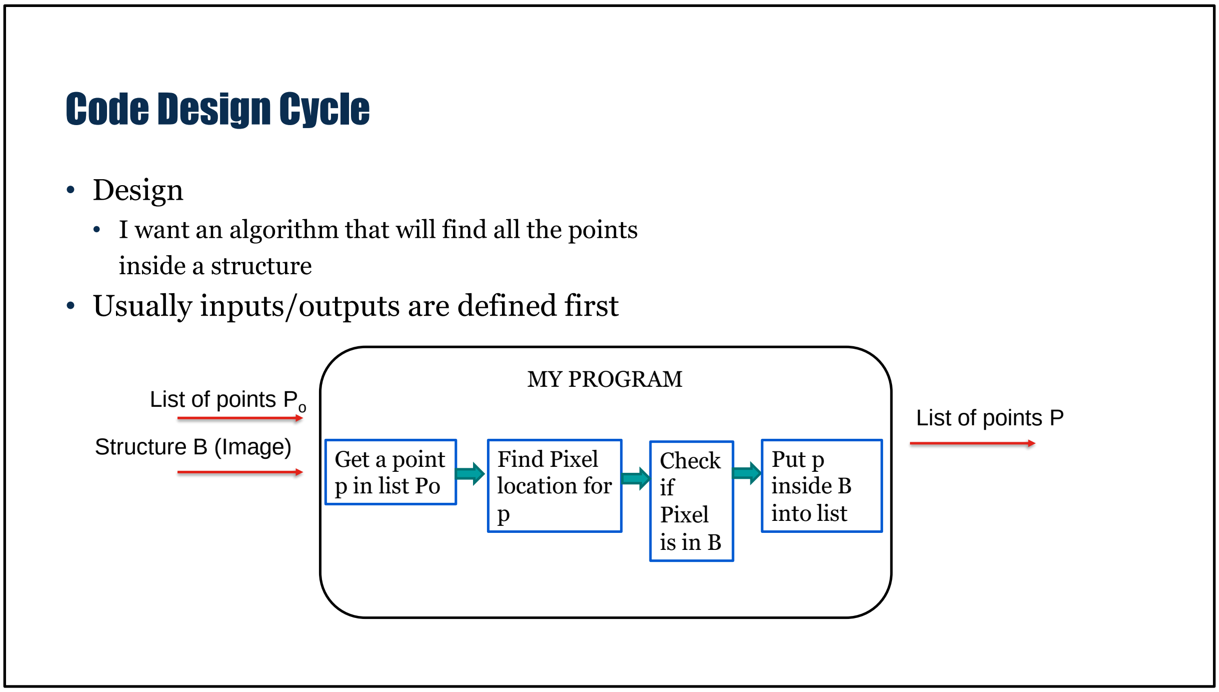
Have them define the problem theoretically



Code Design Cycle

- Design
  - I want an algorithm that will find all the points inside a structure
- Usually inputs/outputs are defined first

Have them define the problem theoretically



Code Design Cycle

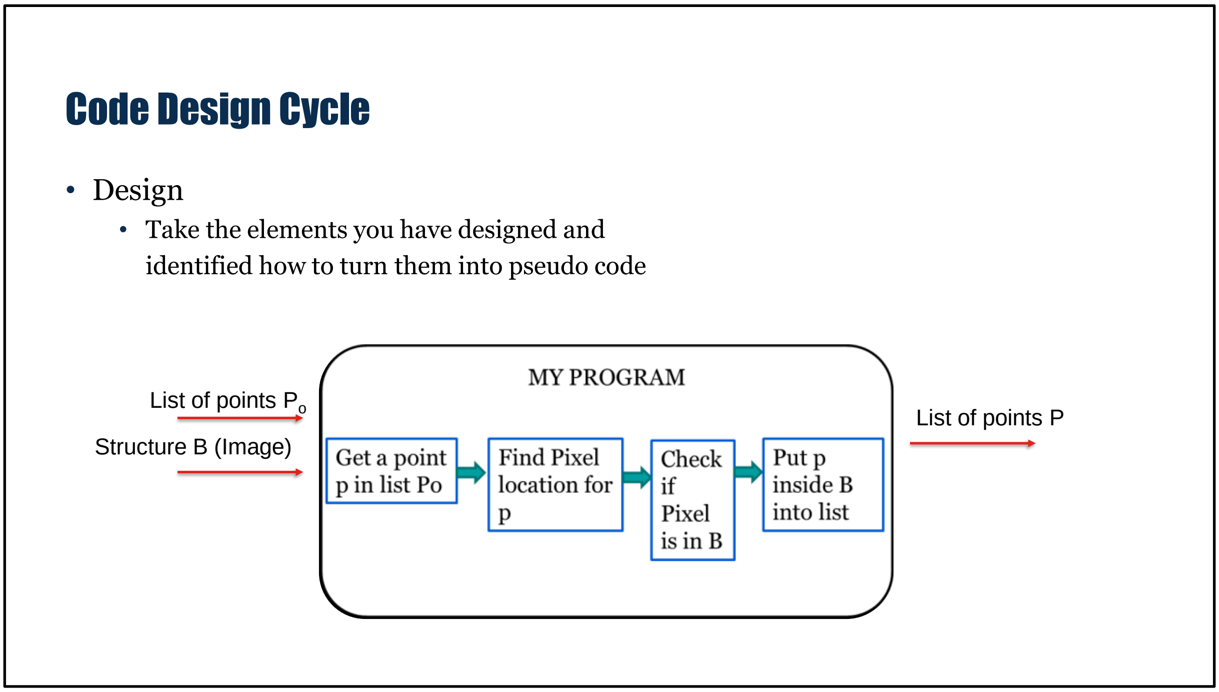
- Design
  - I want an algorithm that will find all the points inside a structure
- Usually inputs/outputs are defined first

What can we further define in the inputs

Think about the elements and how they need to be connected

Does not have to be linear just think about the minimum step you can take for a specific input or desired output

Then think about how to line up the boxes– don't worry about having to "add" boxes if you need to make a link!



Code Design Cycle

- Design
  - Take the elements you have designed and identified how to turn them into pseudo code

MY PROGRAM

List of points P_o

Structure B (Image)

Get a point p in list Po

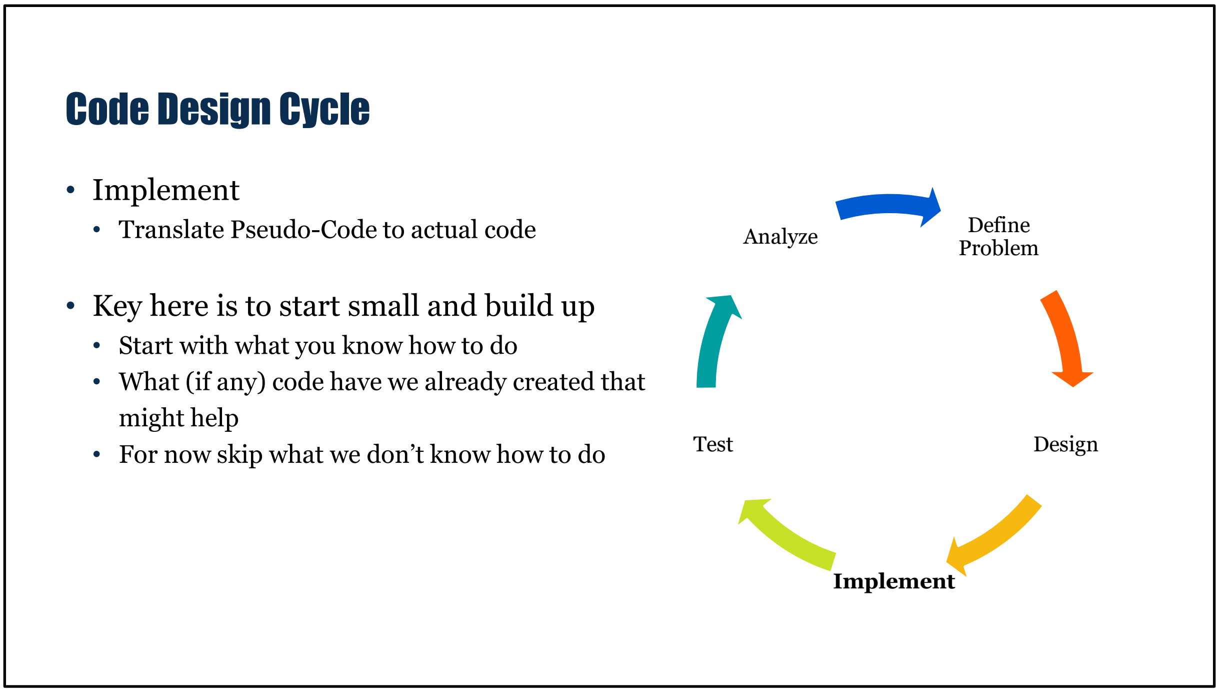
Find Pixel location for p

Check if Pixel is in B

Put p inside B into list

List of points P

What can we further define in the inputs



Code Design Cycle

- Implement
  - Translate Pseudo-Code to actual code
- Key here is to start small and build up
  - Start with what you know how to do
  - What (if any) code have we already created that might help
  - For now skip what we don't know how to do

Analyze

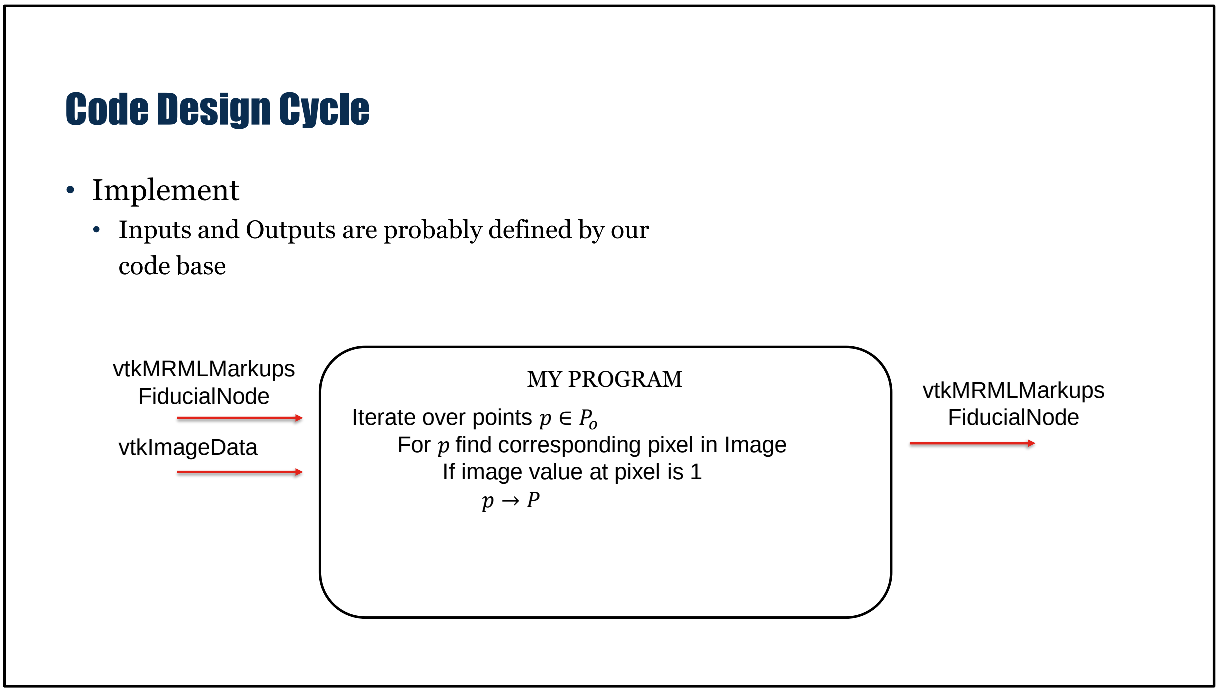
Define Problem

Design

Implement

Test

Have them define the problem theoretically



Code Design Cycle

- Implement
  - Inputs and Outputs are probably defined by our code base

vtkMRMLMarkups FiducialNode →

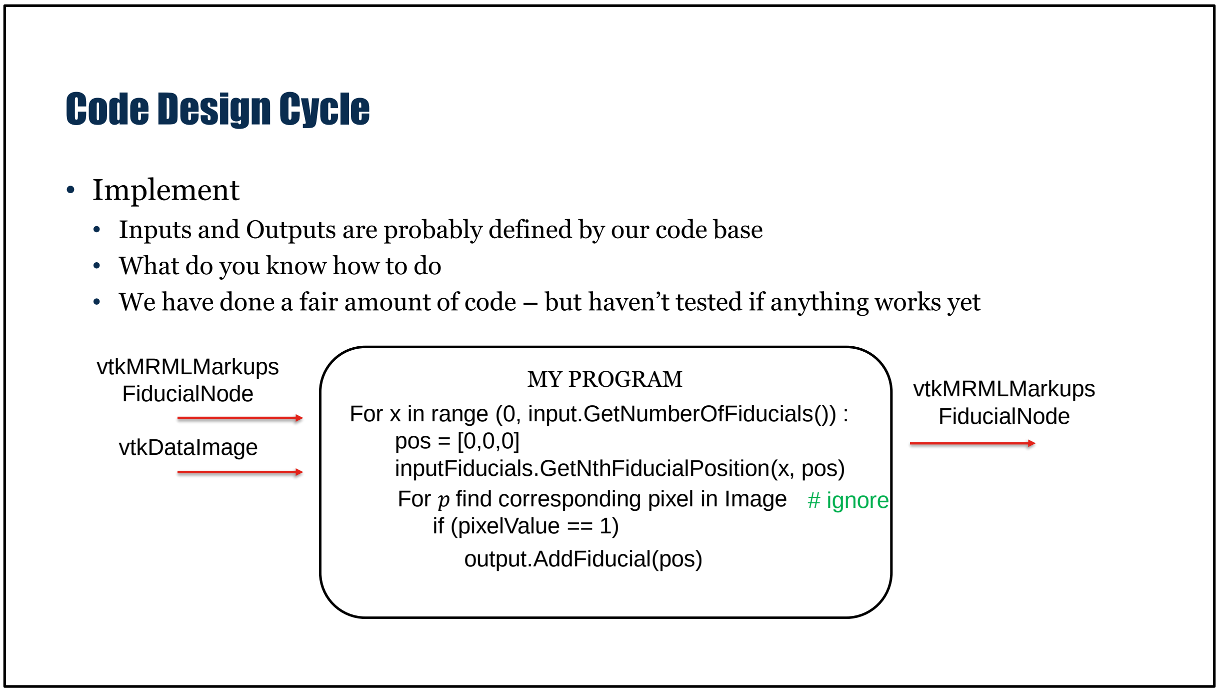
vtkImageData →

MY PROGRAM

```
Iterate over points $p \in P_o$
    For $p$ find corresponding pixel in Image
        If image value at pixel is 1
            $p \rightarrow P$
```

→ vtkMRMLMarkups FiducialNode

What can we further define in the inputs



Code Design Cycle

- Implement
  - Inputs and Outputs are probably defined by our code base
  - What do you know how to do
  - We have done a fair amount of code – but haven’t tested if anything works yet

vtkMRMLMarkups FiducialNode

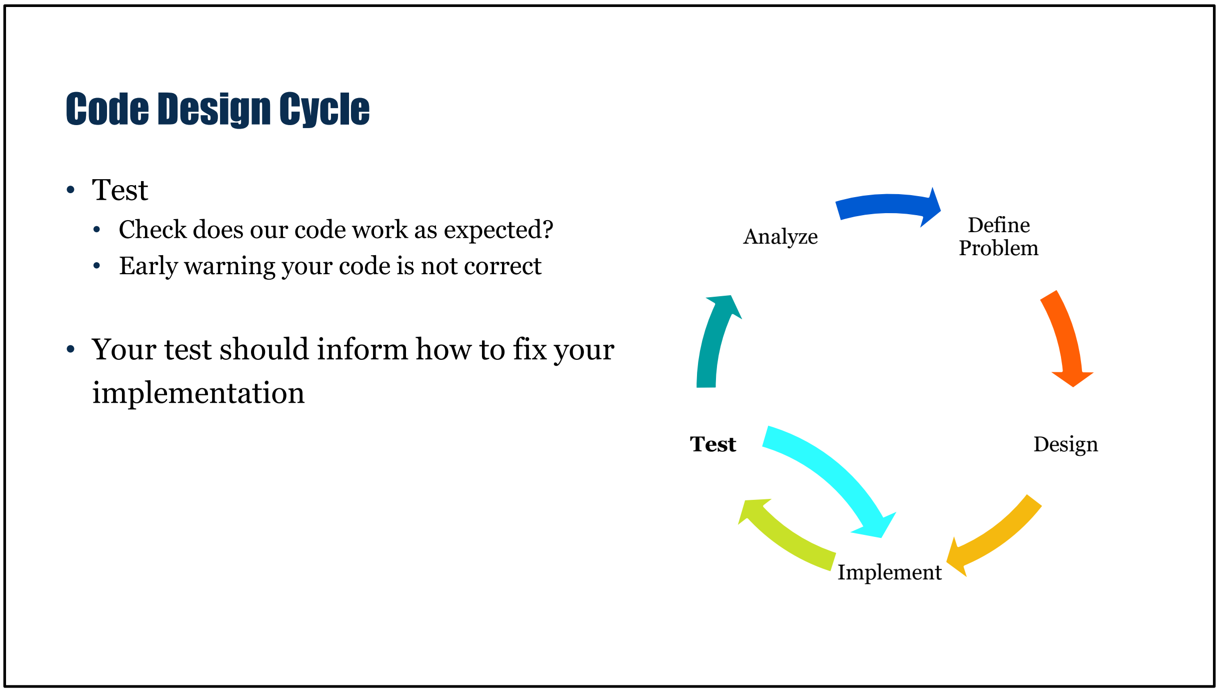
vtkDataImage

MY PROGRAM

```
For x in range (0, input.GetNumberOfFiducials()) :
    pos = [0,0,0]
    inputFiducials.GetNthFiducialPosition(x, p‍os)
    For p find corresponding pixel in Image    # ignore
        if (pixelValue == 1)
            output.AddFiducial(pos)
```

vtkMRMLMarkups FiducialNode

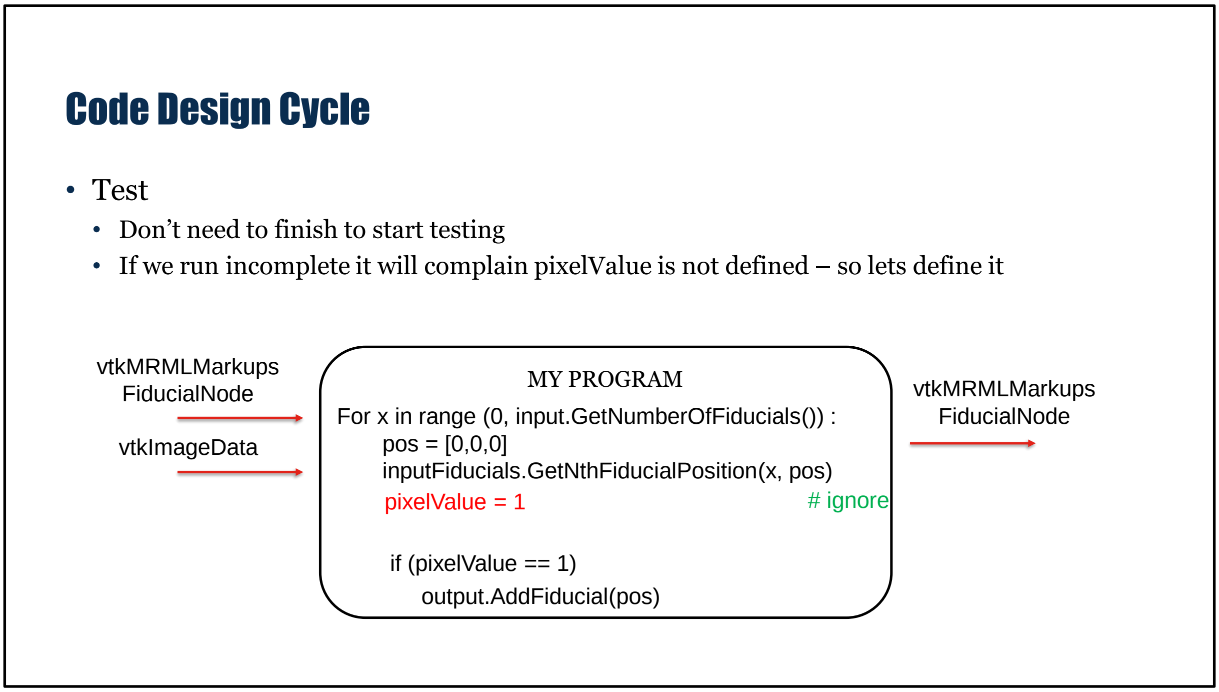
Start adding in simple things we know how to do ignoring those things we don’t know how to do



Code Design Cycle

- Test
  - Check does our code work as expected?
  - Early warning your code is not correct
- Your test should inform how to fix your implementation

What can we test and how?



Code Design Cycle

- Test
  - Don’t need to finish to start testing
  - If we run incomplete it will complain pixelValue is not defined – so lets define it

vtkMRMLMarkups FiducialNode

vtkImageData

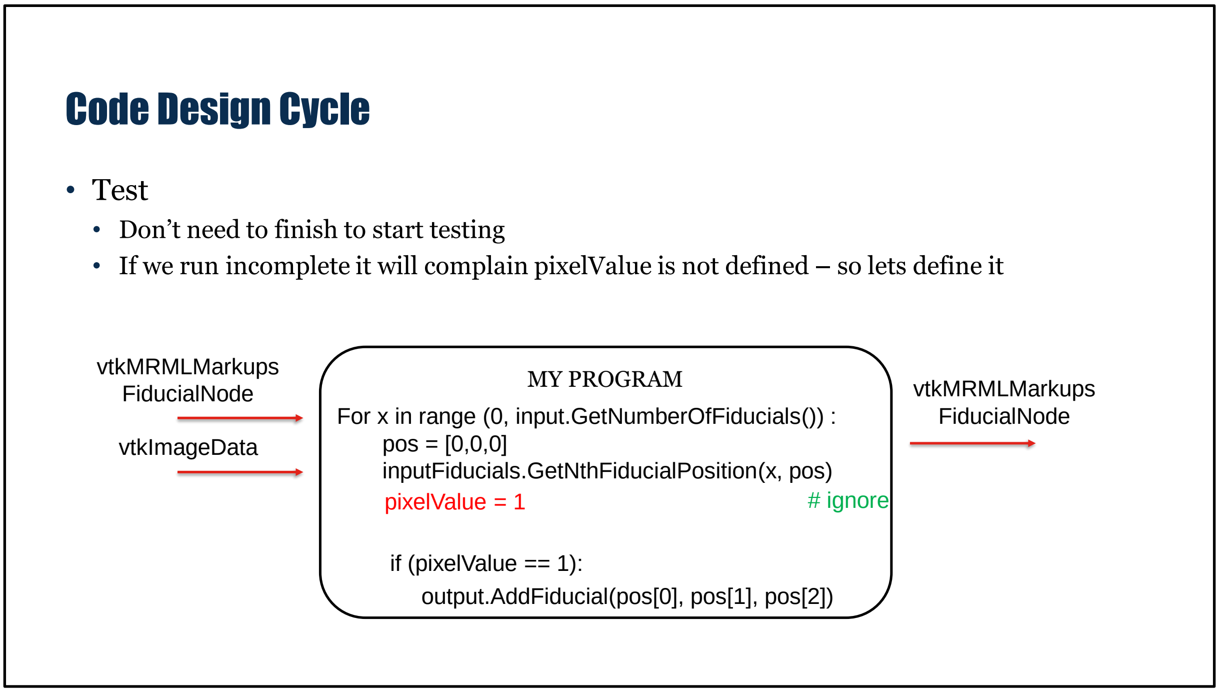
MY PROGRAM

```
For x in range (0, input.GetNumberOfFiducials()) :
      pos = [0,0,0]
      inputFiducials.GetNthFiducialPosition(x, pos)
      pixelValue = 1                                # ignore

       if (pixelValue == 1)
           output.AddFiducial(pos)
```

vtkMRMLMarkups FiducialNode

Fill in misc values/variables that we don’t know how to properly complete yet for the purpose of testing



Code Design Cycle

- Test
  - Don’t need to finish to start testing
  - If we run incomplete it will complain pixelValue is not defined – so lets define it

vtkMRMLMarkups FiducialNode

vtkImageData

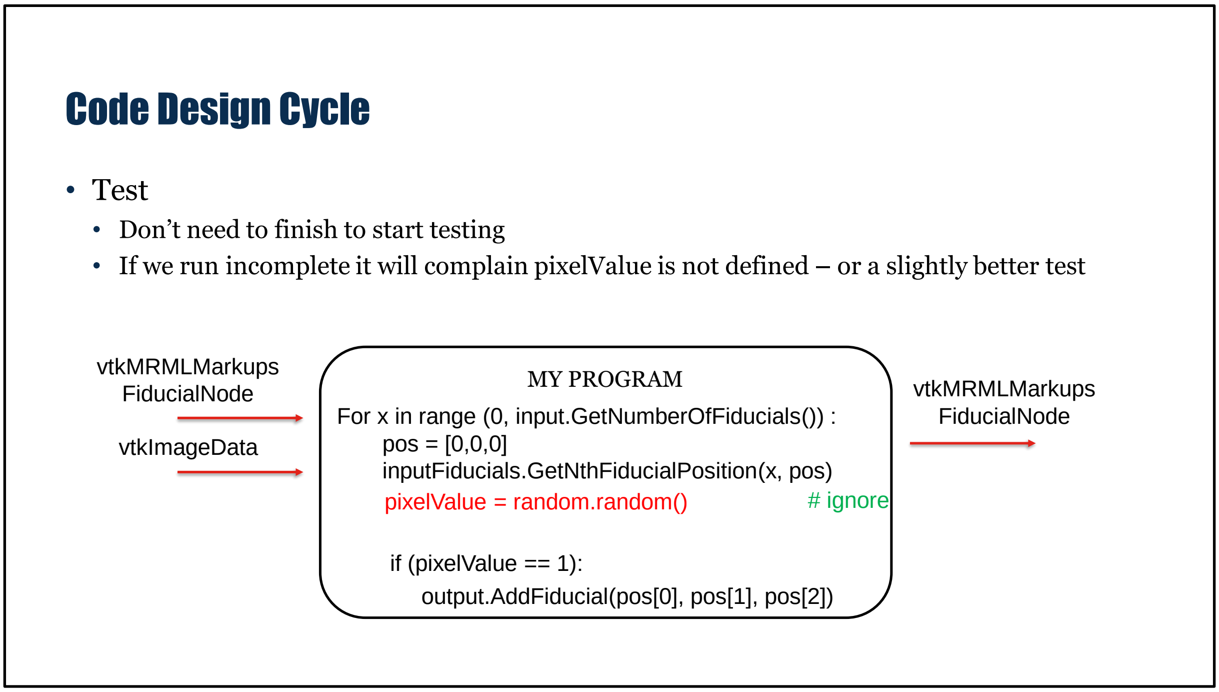
MY PROGRAM

```
For x in range (0, input.GetNumberOfFiducials()) :
      pos = [0,0,0]
      inputFiducials.GetNthFiducialPosition(x, pos)
      pixelValue = 1                                   # ignore

       if (pixelValue == 1):
           output.AddFiducial(pos[0], pos[1], pos[2])
```

vtkMRMLMarkups FiducialNode

Fix any mistakes so that we return what we would expect



Code Design Cycle

- Test
  - Don’t need to finish to start testing
  - If we run incomplete it will complain pixelValue is not defined – or a slightly better test

vtkMRMLMarkups FiducialNode

vtkImageData

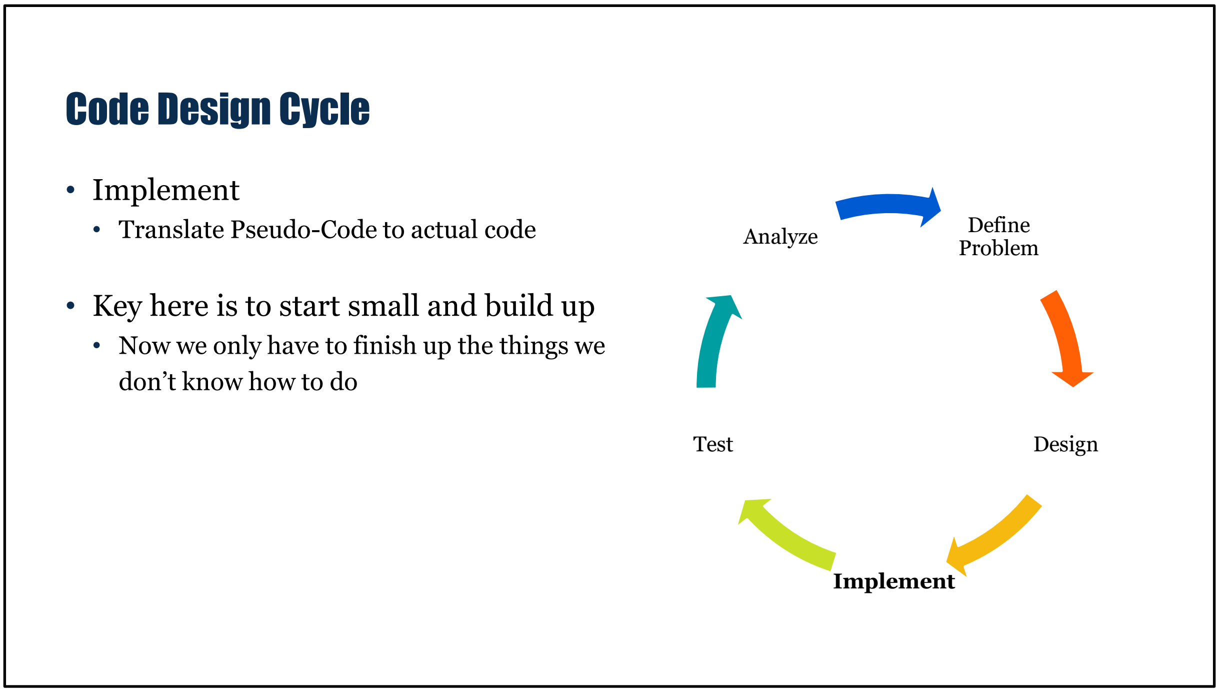
MY PROGRAM

```
For x in range (0, input.GetNumberOfFiducials()) :
      pos = [0,0,0]
      inputFiducials.GetNthFiducialPosition(x, pos)
      pixelValue = random.random()                 # ignore

       if (pixelValue == 1):
           output.AddFiducial(pos[0], pos[1], pos[2])
```

vtkMRMLMarkups FiducialNode

Maybe try creating a slightly better check – just to see that our if statement is working correctly.



Code Design Cycle

- Implement
  - Translate Pseudo-Code to actual code
- Key here is to start small and build up
  - Now we only have to finish up the things we don’t know how to do

Analyze

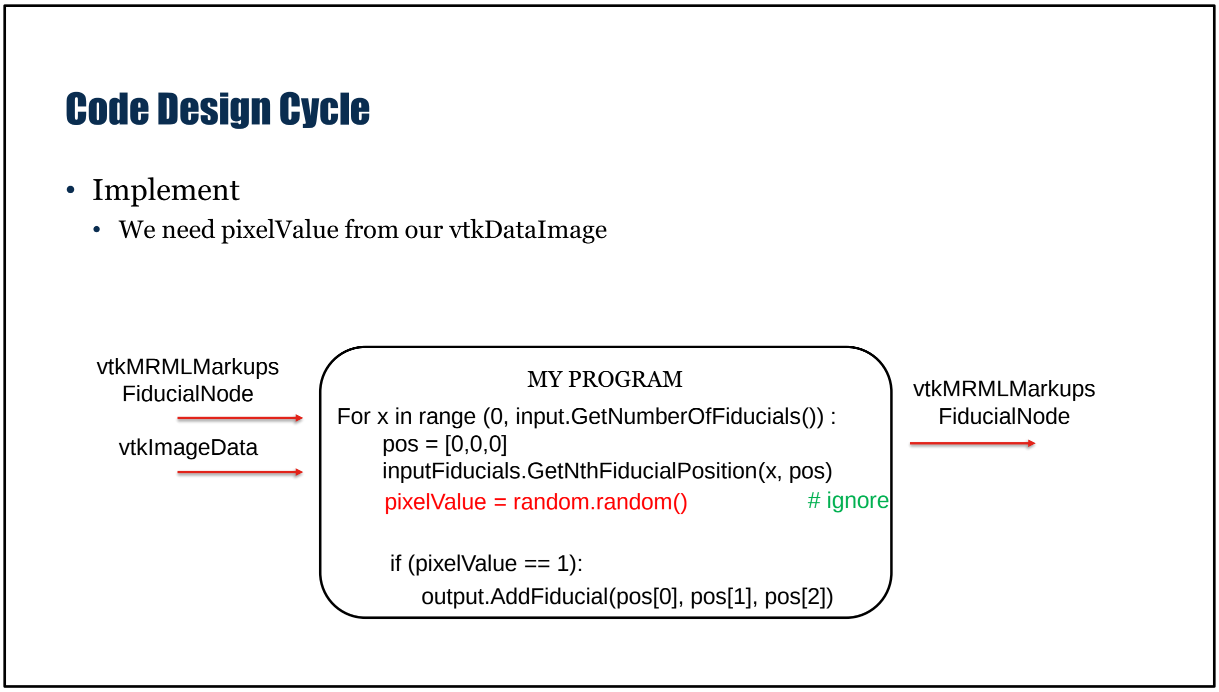
Define Problem

Design

Implement

Test

Have them define the problem theoretically



Code Design Cycle

- Implement
  - We need pixelValue from our vtkDataImage

vtkMRMLMarkups FiducialNode

vtkImageData

MY PROGRAM

```
For x in range (0, input.GetNumberOfFiducials()) :
      pos = [0,0,0]
      inputFiducials.GetNthFiducialPosition(x, pos)
      pixelValue = random.random()                # ignore

       if (pixelValue == 1):
           output.AddFiducial(pos[0], pos[1], pos[2])
```

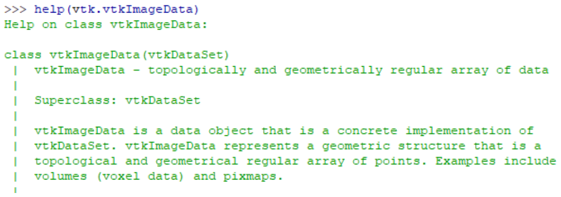
vtkMRMLMarkups FiducialNode

Finally we can focus on filling in the line of code we have chosen to ignore.

vtkImageData

There are many ways to do this

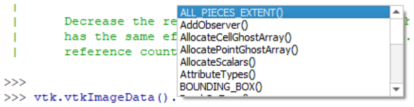
- Documentation: https://vtk.org/doc/nightly/html/classvtkImageData.html
- Inline help



```
>>> help(vtk.vtkImageData)
Help on class vtkImageData:

class vtkImageData(vtkDataSet)
 |  vtkImageData - topologically and geometrically regular array of data
 |
 |  Superclass: vtkDataSet
 |
 |  vtkImageData is a data object that is a concrete implementation of
 |  vtkDataSet. vtkImageData represents a geometric structure that is a
 |  topological and geometrical regular array of points. Examples include
 |  volumes (voxel data) and pixmaps.
 |
```

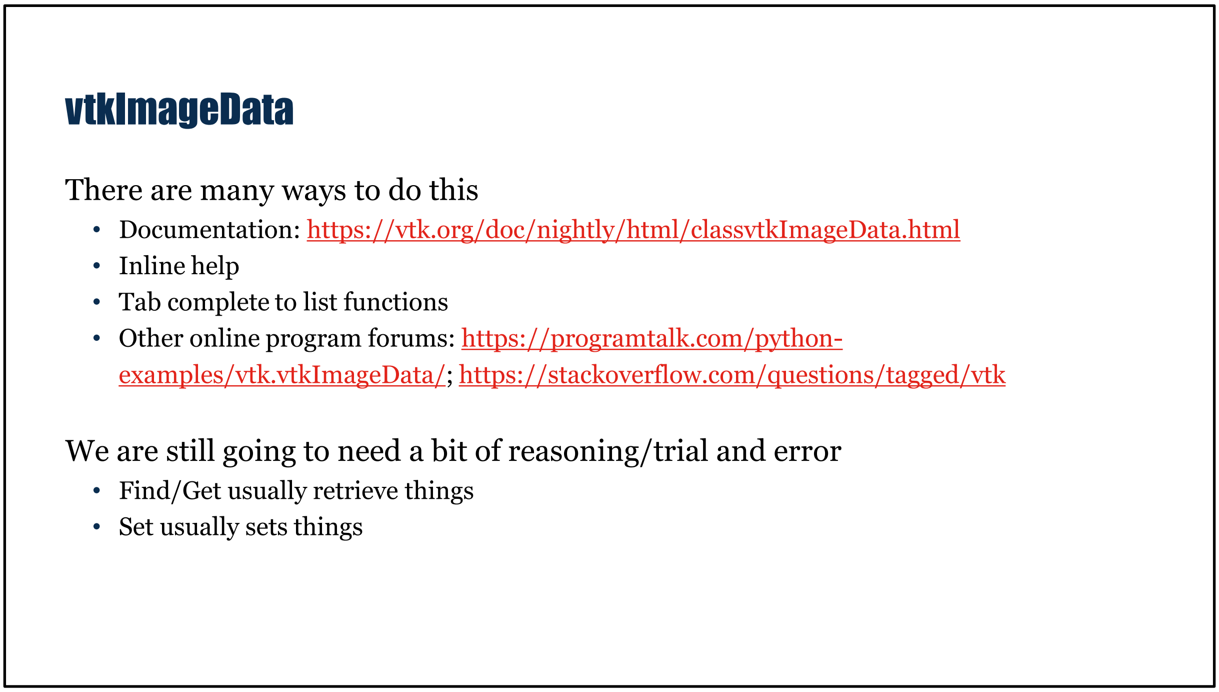
- Tab complete to list functions



```
 |
 |      Decrease the re
 |      has the same ef
 |      reference count

>>>
>>> vtk.vtkImageData().
```

First determine where you should look to find the functions/calls



vtkImageData

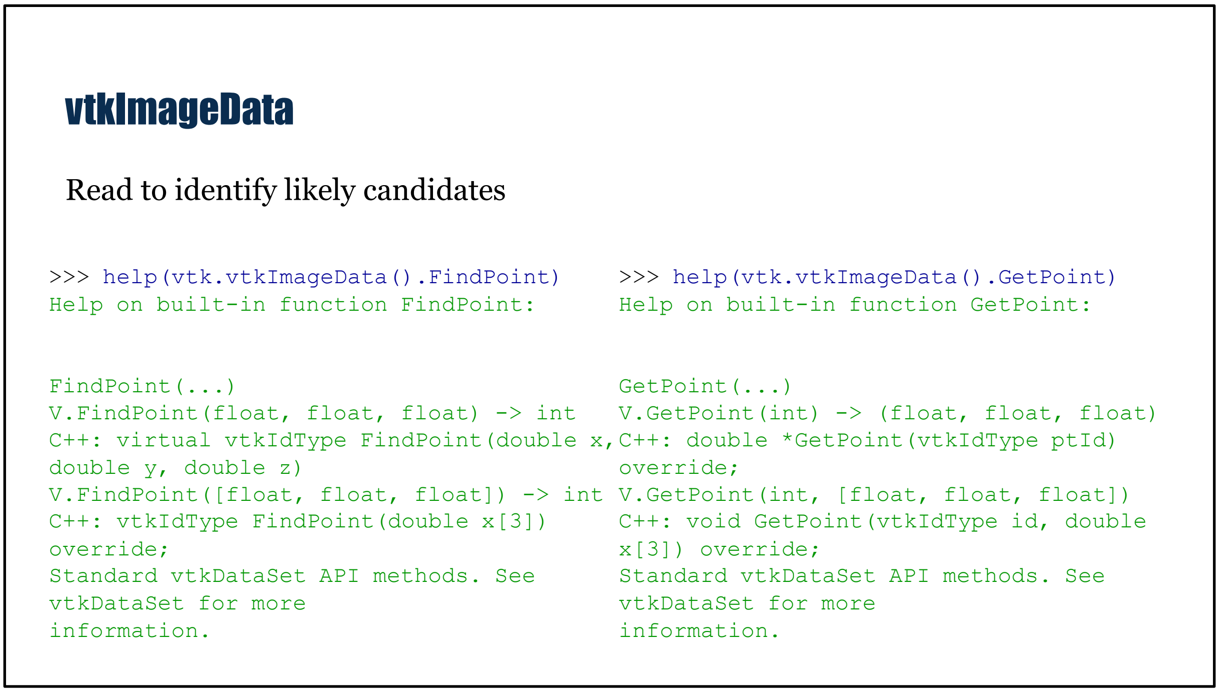
There are many ways to do this

- Documentation: https://vtk.org/doc/nightly/html/classvtkImageData.html
- Inline help
- Tab complete to list functions
- Other online program forums: https://programtalk.com/python-examples/vtk.vtkImageData/; https://stackoverflow.com/questions/tagged/vtk

We are still going to need a bit of reasoning/trial and error

- Find/Get usually retrieve things
- Set usually sets things

Use some knowledge/logic to help figure out where to start your investigation



vtkImageData

Read to identify likely candidates

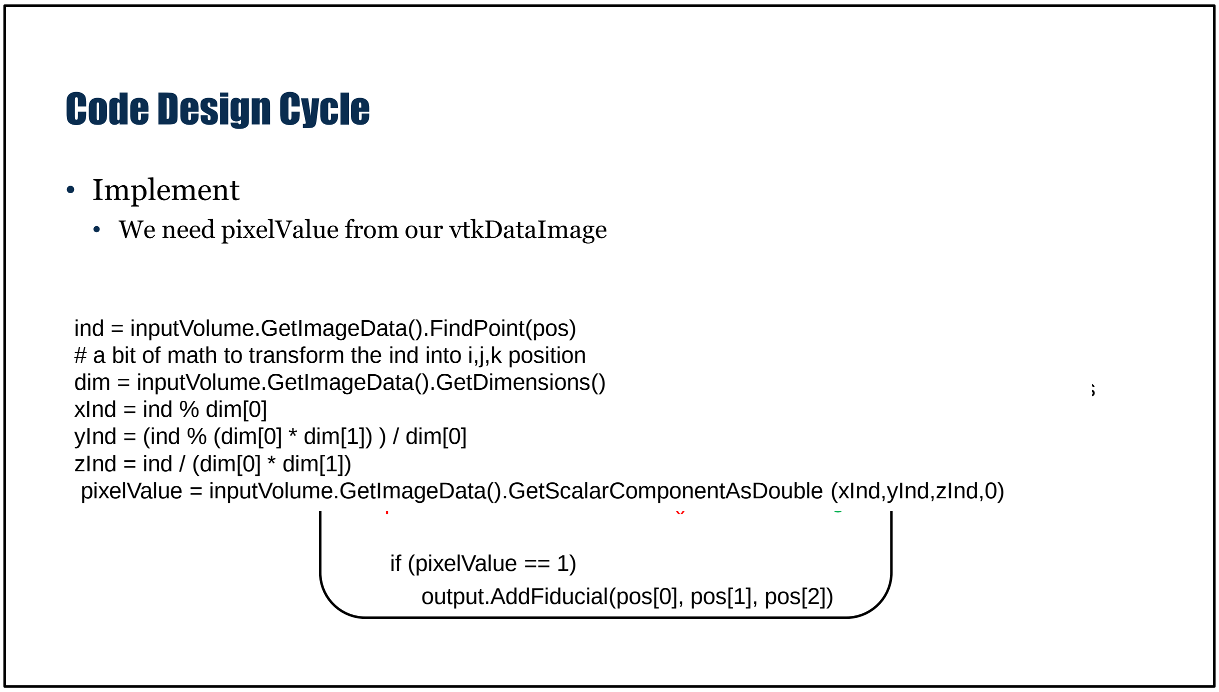
```
>>> help(vtk.vtkImageData().FindPoint)
Help on built-in function FindPoint:


FindPoint(...)
V.FindPoint(float, float, float) -> int
C++: virtual vtkIdType FindPoint(double x,
double y, double z)
V.FindPoint([float, float, float]) -> int
C++: vtkIdType FindPoint(double x[3])
override;
Standard vtkDataSet API methods. See
vtkDataSet for more
information.
```

```
>>> help(vtk.vtkImageData().GetPoint)
Help on built-in function GetPoint:


GetPoint(...)
V.GetPoint(int) -> (float, float, float)
C++: double *GetPoint(vtkIdType ptId)
override;
V.GetPoint(int, [float, float, float])
C++: void GetPoint(vtkIdType id, double
x[3]) override;
Standard vtkDataSet API methods. See
vtkDataSet for more
information.
```

Help paired with your knowledge of how the code is named can help identify possible functions to use



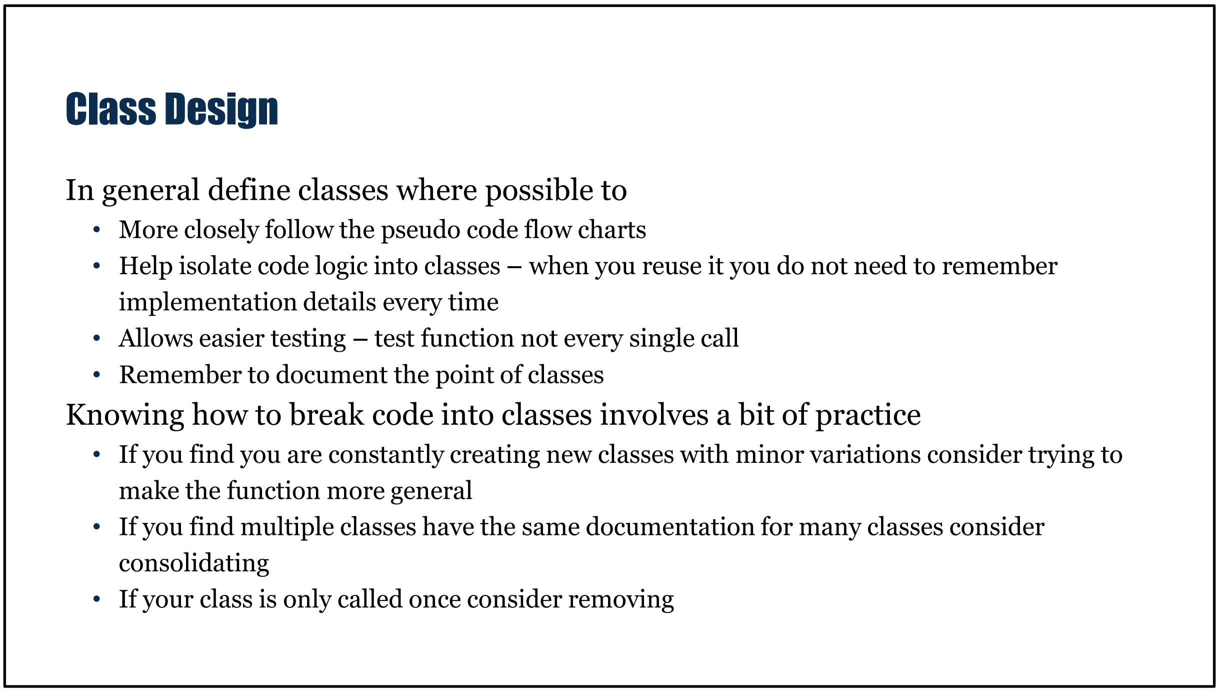
Code Design Cycle

- Implement
  - We need pixelValue from our vtkDataImage

```
ind = inputVolume.GetImageData().FindPoint(pos)
# a bit of math to transform the ind into i,j,k position
dim = inputVolume.GetImageData().GetDimensions()
xInd = ind % dim[0]
yInd = (ind % (dim[0] * dim[1]) ) / dim[0]
zInd = ind / (dim[0] * dim[1])
 pixelValue = inputVolume.GetImageData().GetScalarComponentAsDouble (xInd,yInd,zInd,0)
```

```
if (pixelValue == 1)
    output.AddFiducial(pos[0], pos[1], pos[2])
```

After some trial and error we can figure out how to translate our point into a pixel



Class Design

In general define classes where possible to

- More closely follow the pseudo code flow charts
- Help isolate code logic into classes – when you reuse it you do not need to remember implementation details every time
- Allows easier testing – test function not every single call
- Remember to document the point of classes

Knowing how to break code into classes involves a bit of practice

- If you find you are constantly creating new classes with minor variations consider trying to make the function more general
- If you find multiple classes have the same documentation for many classes consider consolidating
- If your class is only called once consider removing

Code Refactoring

Don’t be afraid to rewrite code.

- Once you have implemented your entire algorithm you may realise there are unnecessary calls or multiple similar lines of code
- Similar code may indicate you need to create a class
- If after implementation you think you have a simpler way to achieve the same results try it
  - Fewer lines of code can be easier to debug
  - More readable code can be easier to use and debug later