Class Granularity

It is important to use encapsulation appropriately

We need to hide state in suitably sized classes

A class with a single attribute might well be to small

A class with hundreds of attributes might be too big

Remember that classes can consist of other classes

We can have a nested hierarchy of encapsulation !

Structural Cohesion

Classes should be "cohesive":

"A logical & coherent cluster of data & behaviour"

Is

a tumble dryer

that also makes coffee

cohesive ?

Structural Coupling

Classes should also be distinct and independent

There should be no tight coupling between classes

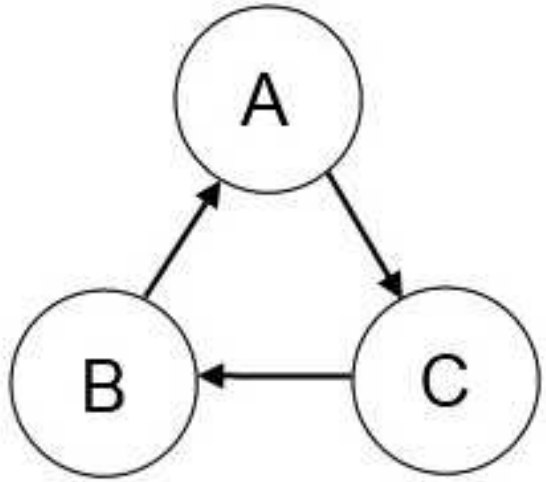
Cyclic Dependency

It is good to have a clear hierarchy of responsibility

Like management structures in an organisation

It is best not to have cyclic loops in these structures

What if your boss was managed by your subordinate !



Problems with Cyclic Dependency

It makes it uncertain who is in control/command
Unclear allocation of functionality to responsible class

Who's job is it to complete a task ?
Who's job to make sure the task has been done ?

Also makes it difficult to replace parts of the system
(Due to the many linkages between classes)

Global Variable Referencing

A key feature/benefit of OOP is encapsulation
We hide complexity away inside objects

It is "bad form" to break that encapsulation...
"Reach in" and "interfere" with attributes directly

Only access internal state through defined methods