Object Oriented Programming with Java

COMSM0103

Dr Simon Lock & Dr Sion Hannuna

Meet The Team - Unit Directors

Simon

Sion

Aim of this Unit

Our main focus will be on programming with Java
An essential element of which is "Object Orientation"
(which is why the unit is called "OOP with Java" !)

Object Orientation is a MAJOR concept
It is not "owned" by Java - other languages use it

OO is a total paradigm shift (quite literally)
There is more to learning a language than syntax !

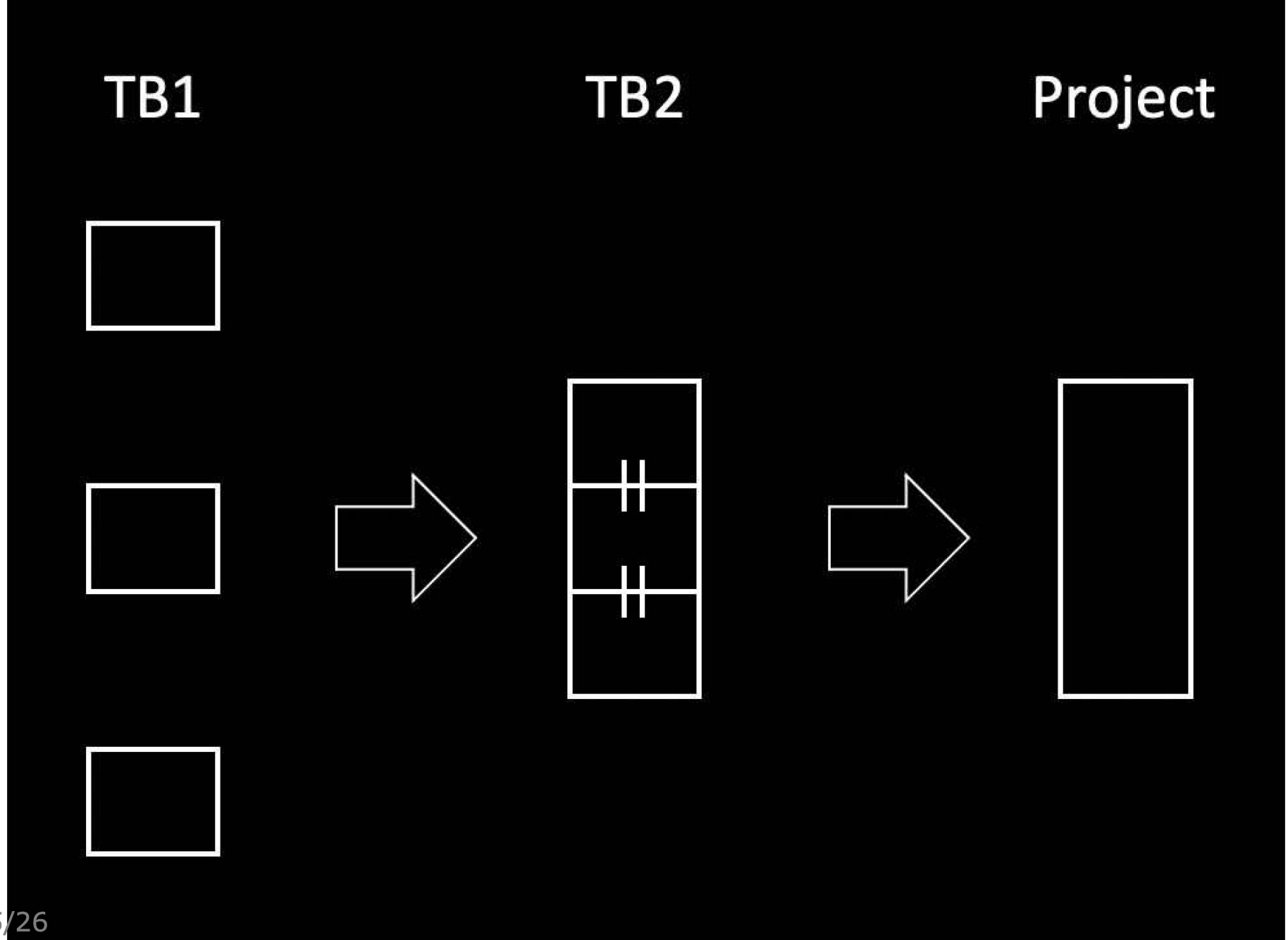
WARNING

This unit isn't JUST about "Coding"
(i.e. implementing a specification you are given)

It's NOT JUST a re-run of TB1 programming units
(just with a different language)

It takes a broader perspective on "Development"
Analysis, Spec., Design, Testing, Maintenance etc.

This unit uses tools from the Software Tools unit
As well as practices & processes from Software Eng.


TB1
TB2
Project

Weekly Workbooks

Teaching centres around series of weekly workbooks
Consisting of various practical tasks to work through

Workbooks take a "problem led" approach:
Tasks provide a reason/motivation for your learning

Templates/code examples provided as needed
Lecture video fragments integrated inline
"Everything in one place" to streamline consumption

Sandwich ?

This unit is designed to have a "sandwich" structure

Self study workbook: to gain initial understanding
Lecture: provide deeper insight & answer questions
Practical: get one-on-one help to resolve problems
Self study workbook: complete practical activities

Accessing Workbooks

Workbooks are released at the start of each week

They will be made available via GitHub

(easy to upload, easy to download)

Clone/pull workbooks and view them locally

This makes viewing linked documents much easier

Weekly Practical Labs

2 hour practical each week: Friday 1-3pm MVB2.11

A team of skilled teaching assistants to help !

Abel

Gene

Alex

Jerry

Recommended IDE

IntelliJ is the supported IDE for this unit

It is already installed on the lab machines

For this unit, you will need _at least_ Java 11...

Just pick the latest version if get a choice

Check out the unit "Welcome" page on Blackboard...

Video guides for installing on your own computer

Would be good to do this as soon as possible !

Assessment

Selected workbooks are assessed
(these will be clearly flagged when time comes !)
There are 3 assessed exercises in total...
But they span multiple weeks and workbooks

Assignments have codenames: OXO, DB and STAG
Each increasing in complexity and difficulty
Each increasing in weight to reflect this
(20%, 35%, 45%)

Questions ?

What is Java ?

Java is a _bit_ like C (syntax is quite similar)

There are however some important differences:

- No explicit pointers (no * & ->) yay !!!
- Automated memory management (no malloc/free)
- No seg faults (you'll get exceptions instead)
- Support for various Object Oriented constructs
- Greater platform independence (more later)

Hello World

Here is the traditional "Hello World" in Java:

```
class HelloWorld
{
    public static void main(String[] args)
    {
        System.out.println("Hello World!");
    }
}
```

Lots of stuff in there !

Some of it looks very similar to C

We'll revisit the key elements in later lectures

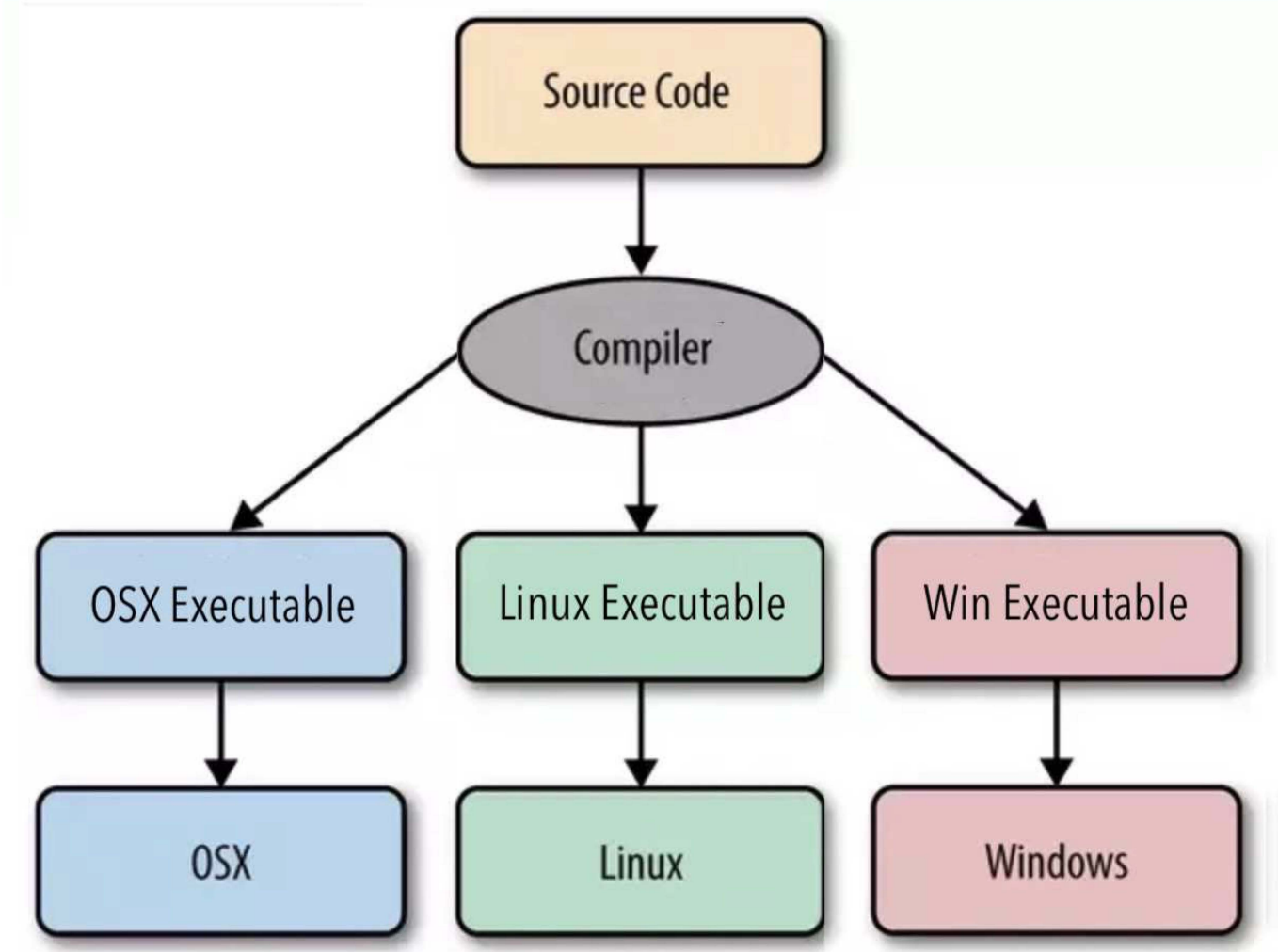
Write Once, Run Anywhere

Power feature: Java runs the same on all platforms
(Except Android Java - which is very different !)

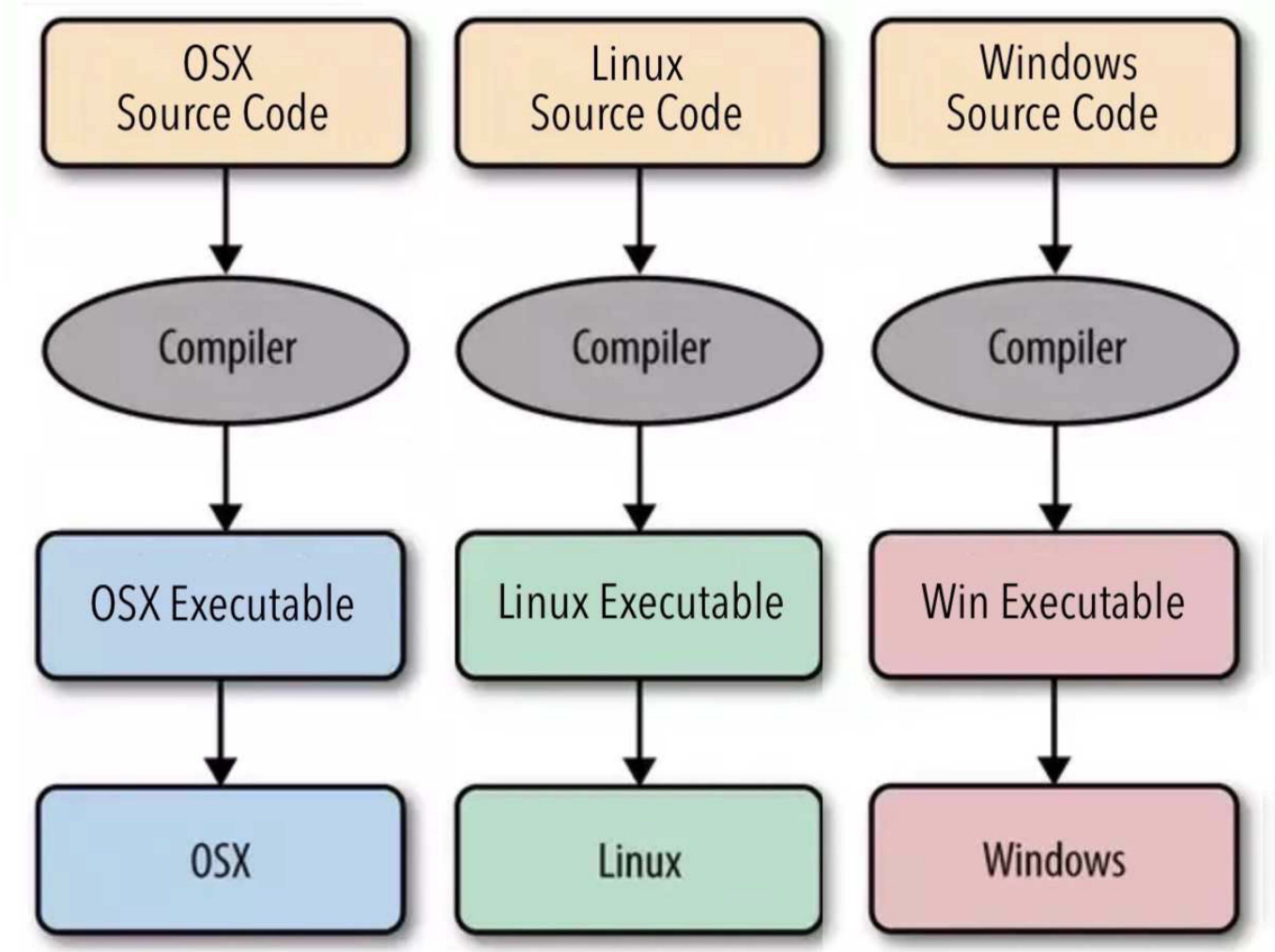
Source code compiled to cross-platform "bytecode"
(midway between source and binary executable)

Bytecode is then interpreted at runtime
By a standardised "Virtual Machine"
(That abstracts over the low-level detail of host OS)

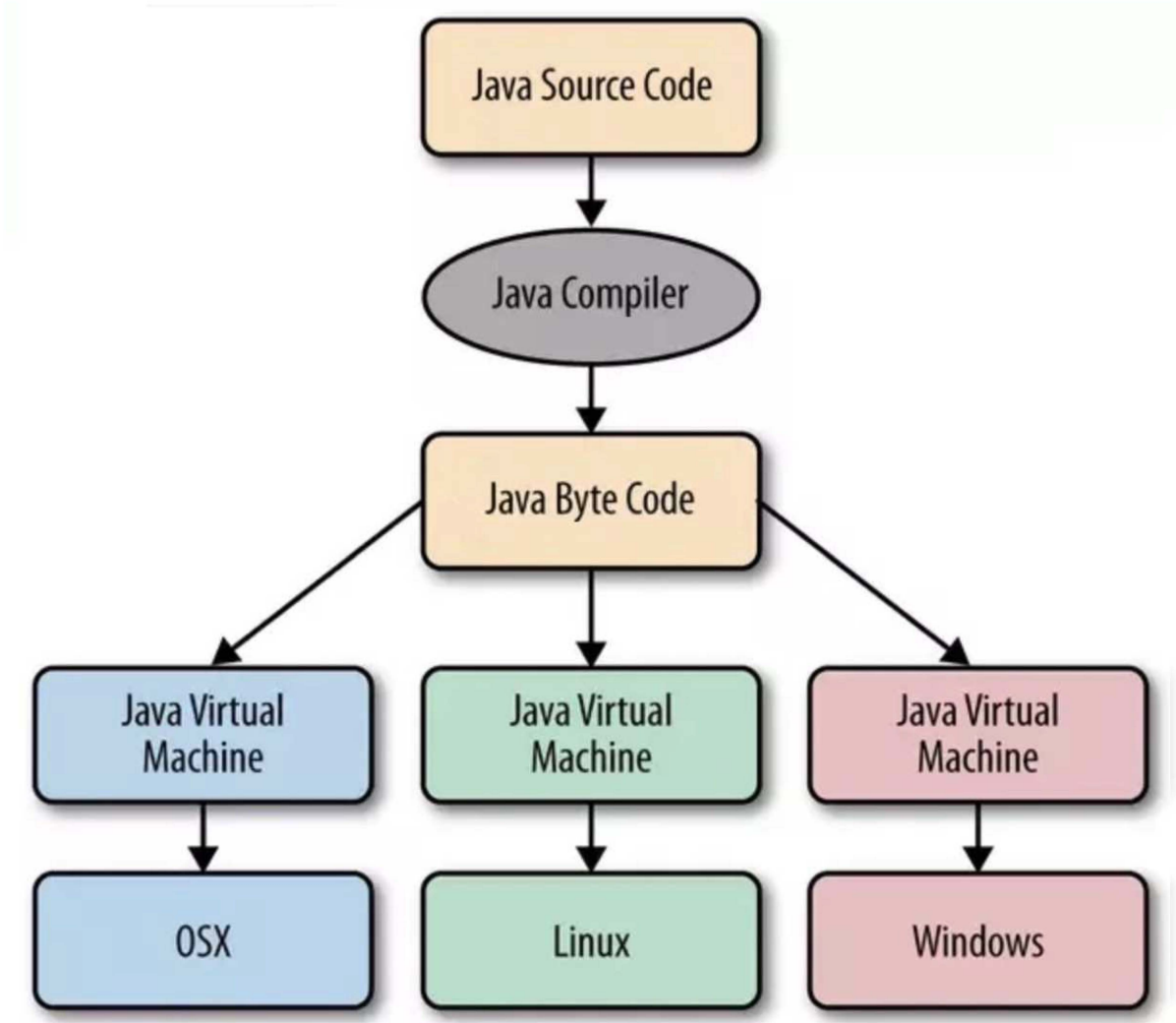
C Programming (in Theory)



Actual C Programming !



Java Programming



Performance

C has a reputation for being fast !

Java for being a little more "leisurely"

(Due to the overhead of Bytecode interpretation)

HOWEVER

Almost all Java Virtual Machines use "JIT" compilers

Convert bytecode into native executables at runtime

"Just-In-Time" to be executed

So the performance difference is not that big

Use of Java

Java is a very popular programming language
https://www.youtube.com/watch?v=Og847HVwRSI

Used for large servers, desktop applications,
mobile devices, embedded processors etc.

It has very little in common with "JavaScript" !
Other than a partially similar name
(and some common syntax)

What type of language ?

C is a procedural/imperative language
("do this and then do that")
With Functional Decomposition as main paradigm:
Functions *delegate* work to sub-functions

Java is also a procedural/imperative language
But its (intended) paradigm is Object Orientation:
Objects *collaborate* together to achieve objective

Difference might seem subtle, but impact is great

Functional Decomposition (what C is)

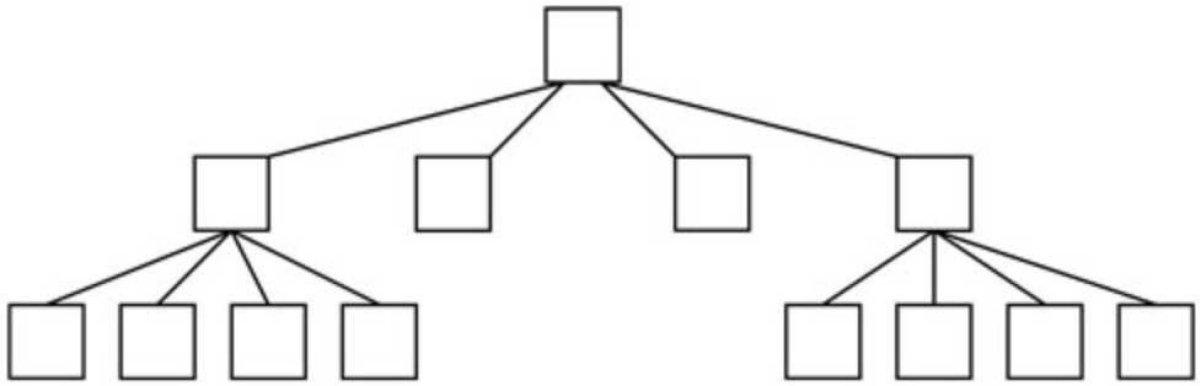
Main function is broken down into smaller functions

Forms a tree, with data flowing around everywhere

Although effective programs can be written this way

They tend to be monolithic (big and frightening)

Also "brittle" (resistant to long-term evolution)



Object Orientation

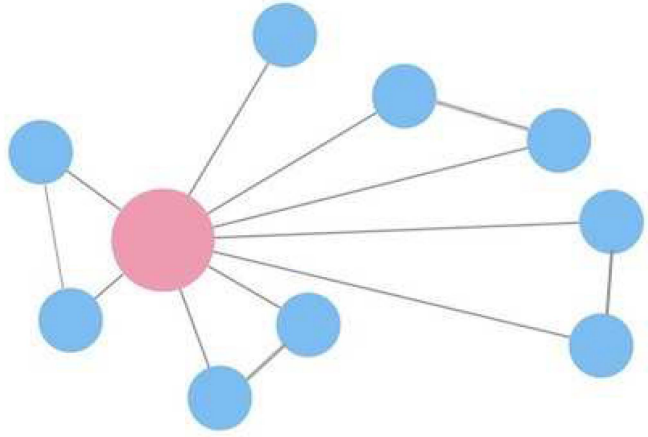
Program written as decentralised cooperating pieces

Each piece (Object) looks after its own internal data

Collaborate with each other to get the job done

Scales well to larger projects (if done properly !)

Works effectively for big teams (if done properly !)



What are Classes and Objects?

We've mentioned them already,
but what exactly ARE Classes and Objects ?

CLASSES: modules that divid up the source code
(Normally each file contains just a single Class)

OBJECTS: structures that divid up running programs
(Each Object encloses its own state and data)

Classes can be viewed as a template (cookie cutter)
from which we can "instantiate" live Objects

Key Characteristics of Object Orientation

Abstraction: sophistication, but with simple interface

Encapsulation: inner working locked away out-of-sight

Inheritance: hierarchies of Classes share behaviours

Polymorphism: like Classes can be treated the same

Yes, these are all very vague, high-level descriptions

But we will explore them all in more detail later !

Let's take a look at the first workbook:

https://github.com/drslock/JAVA2022