Object Oriented Programming with Java

COMSM0086

Introduction and Orientation

Meet The Team - Unit Directors

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 !

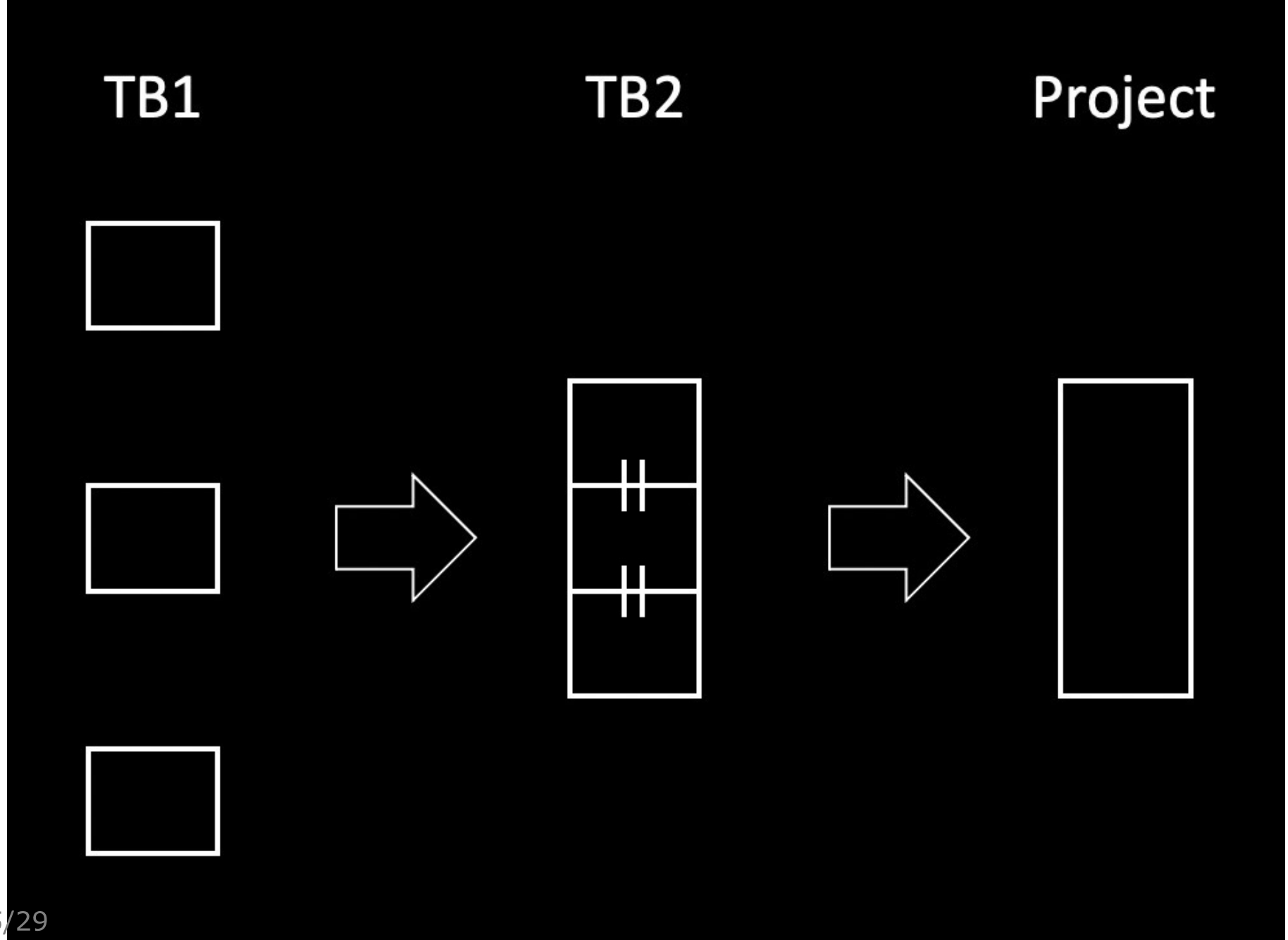
WARNING

This unit is NOT JUST about "Coding"
(i.e. implementing a specification you've been given)

It's NOT JUST a re-run of TB1 programming units
(just using a different programming 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 centers 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 are provided as needed
Lecture video fragments are integrated inline
Content where you need it, when you need it

Sandwich ?

This unit is designed to have a "sandwich" structure

Self study: gain initial understanding of workbook

Practical: get one-on-one help to resolve problems

Self study: complete practical activities of workbook

Don't expect...

To get everything done

during the practical session !

Weekly Practical Briefing

Each week there is a practical briefing session

In this session we will:

- Explore the intricacies of the workbook
- Explain any essential concepts
- Introduce any key skills and tools needed
- Provide an opportunity for Q&A

Briefing takes place Thursday at 13:00 in MBV 2.11

Duration will vary (depending on complexity of topic)

But that is fine, since briefing flows straight into...

Weekly Practical Lab

2 hour (approx) practical, starting around 14:00

(starts after the weekly briefing has finished !)

You'll need to do some pre-processing of materials

Practical shouldn't be first time you open workbook !

(the top slice of bread in the sandwich)

Opportunity to get one-on-one help and advice

With a team of skilled teaching assistants to help...

Meet the Team !

Will Jerry Casper Alex Harry

Weekly Theory Lectures

Theory Lectures: Friday 13:00 in Queens Building 1.40
In these sessions we will go "deep" and "broad"...

Deep

Delving into complex low-level features of Java & OOP

Broad

Situating unit within context of Software Development
(Analysis, Spec., Design, Testing, Maintenance etc.)

You must all like vim by now ?

Recommended IDE

Use of modern dev. environments is a key skill
NOW is right time to introduce this kind of tool !
On this unit we will be using the IntelliJ IDE

IntelliJ is already installed on the lab machines
Check out the unit "Welcome" page on Blackboard...
Video guides for installing on your own computer

For this unit, you will need _at least_ Java 17

Assessments

There will be 2 assessed exercises for this unit

Each spans approximately 2-3 weeks

The first deadline is shortly before easter

The second shortly before the end on term

Weightings: 40% / 60% (reflecting complexity)

Questions ?

Java

Java is a very popular programming language
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)

Object Orientation is core - hence our interest

What is Java like ?

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!");
    }
}
```

Some of it looks very similar to C

But also lots of new stuff in there !

We'll go into more details in the workbook !

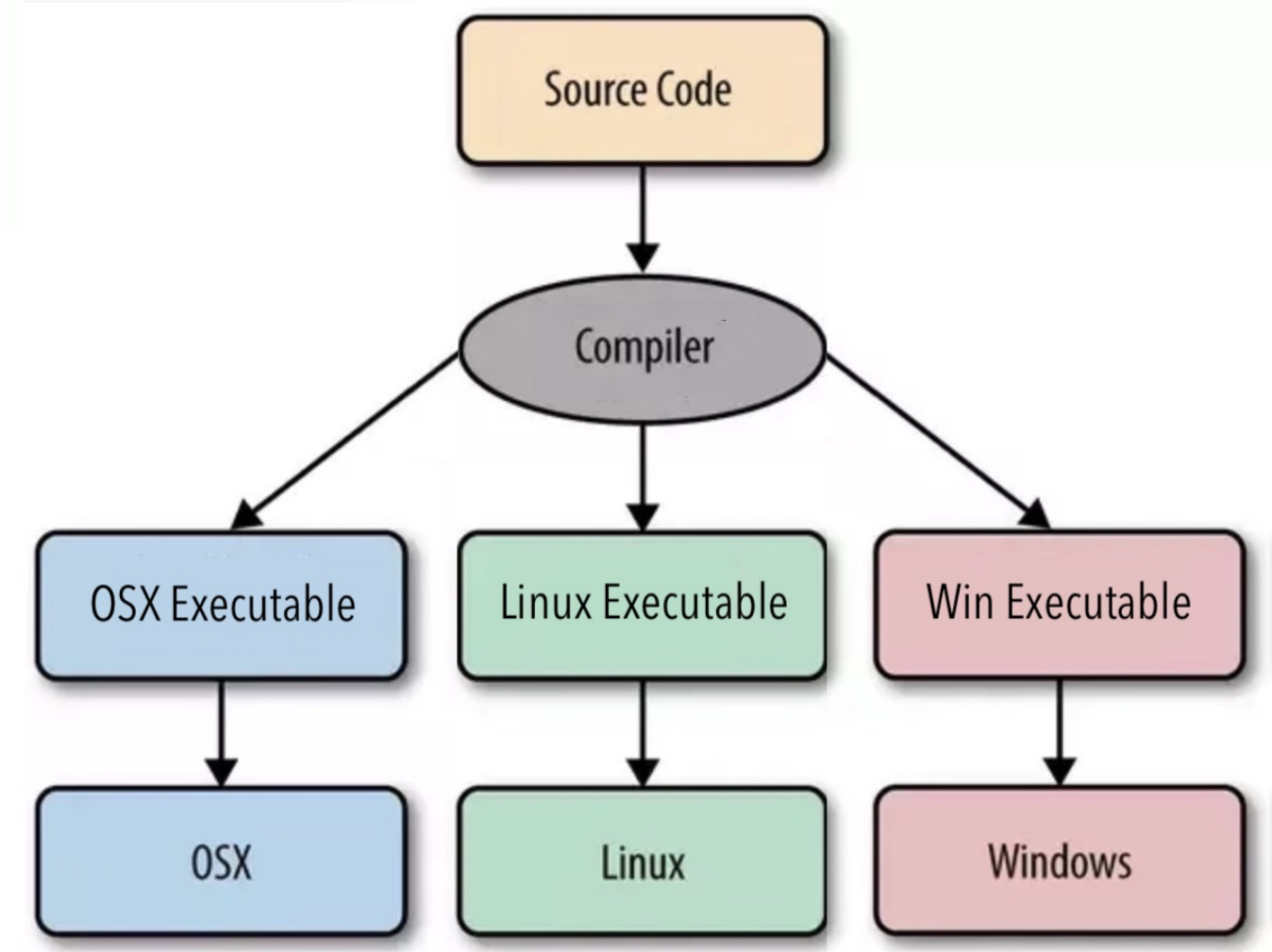
Write Once, Run Anywhere

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

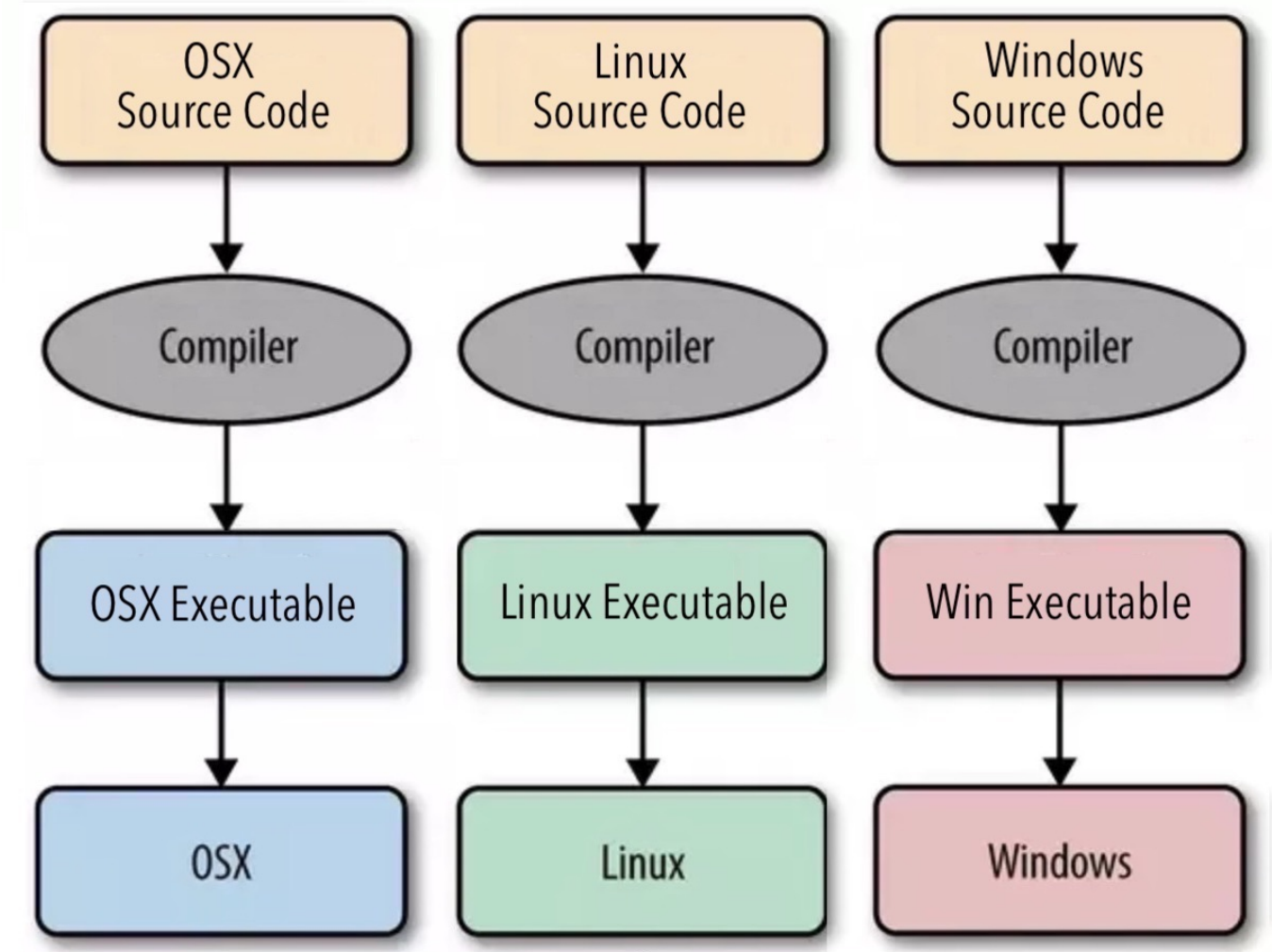
Source code compiled to cross-platform 'bytecode'
(midway between source and binary executable)

Bytecode is *interpreted* at runtime
By a standardised 'Virtual Machine'
(This abstracts over the low-level detail of host OS)

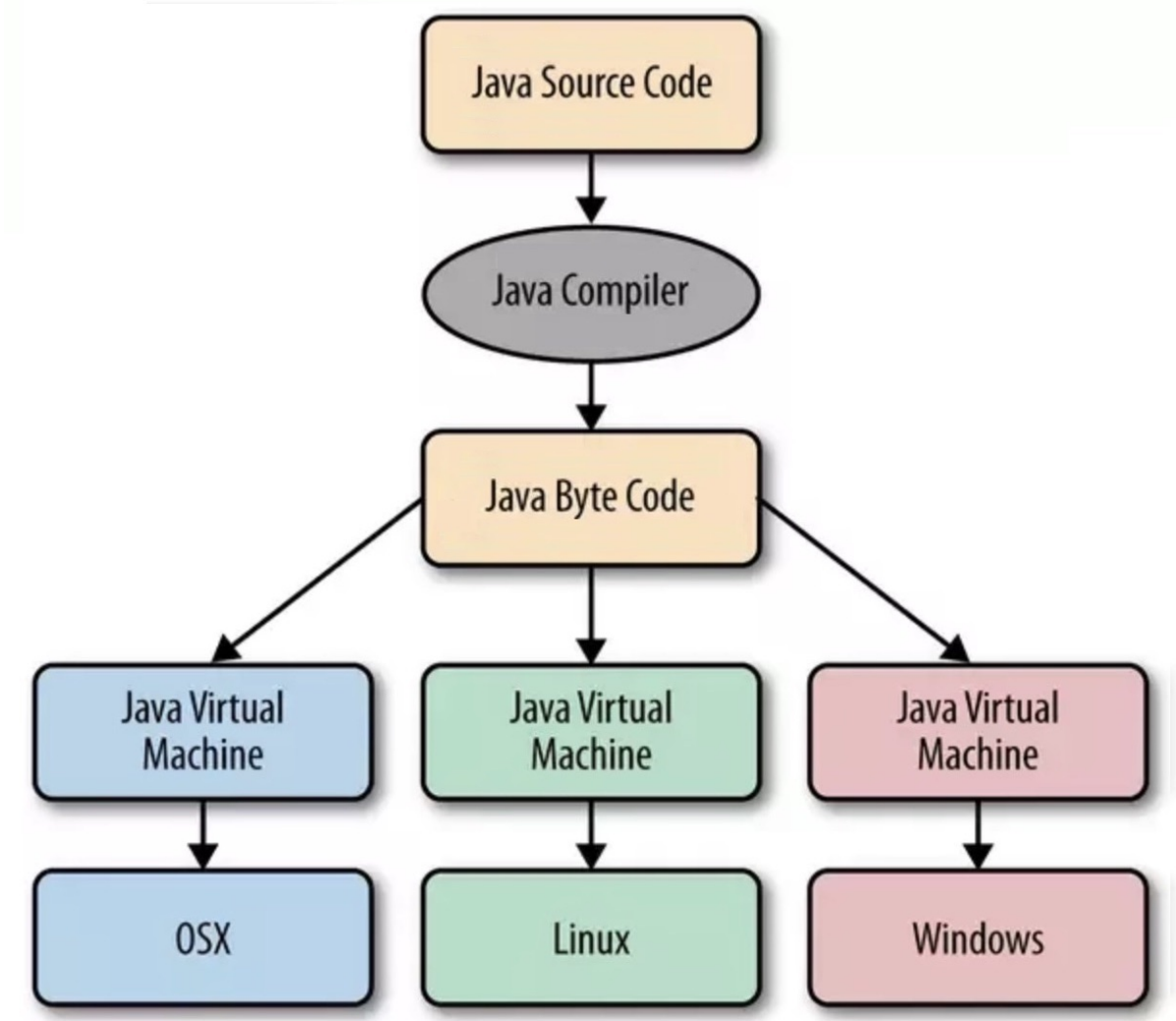
C Programming - in Theory



C Programming - in Practice



Java Programming - in Theory & Practice !



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 (hence the name)

So performance difference is actually not that big

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)

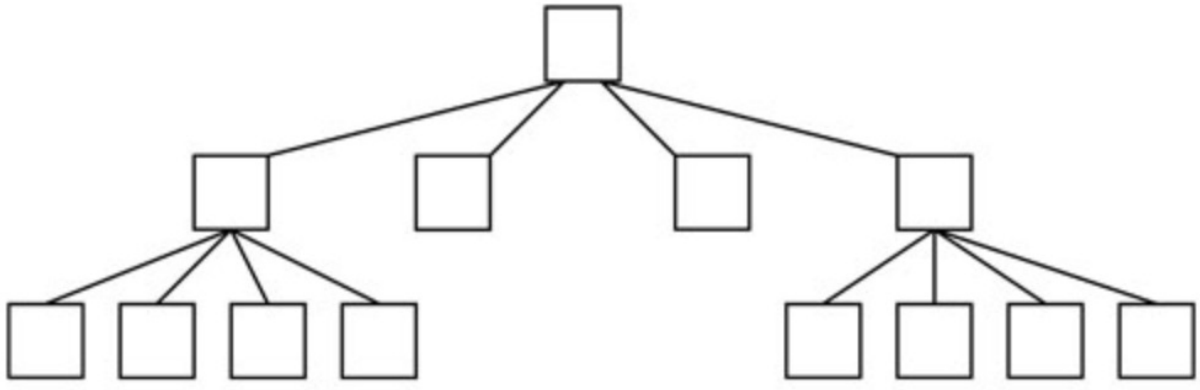
Main function is broken down into smaller functions

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

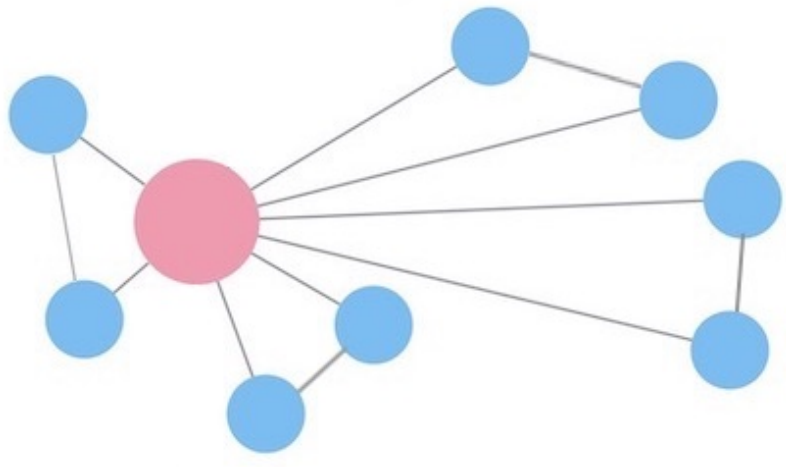
Program written as decentralised cooperating pieces

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 large teams (if done properly !)



What are Classes and Objects?

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

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

OBJECTS: structures that divide up running program
(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 !

Enough talk, let's take a look at first workbook:

https://github.com/drslock/JAVA2023