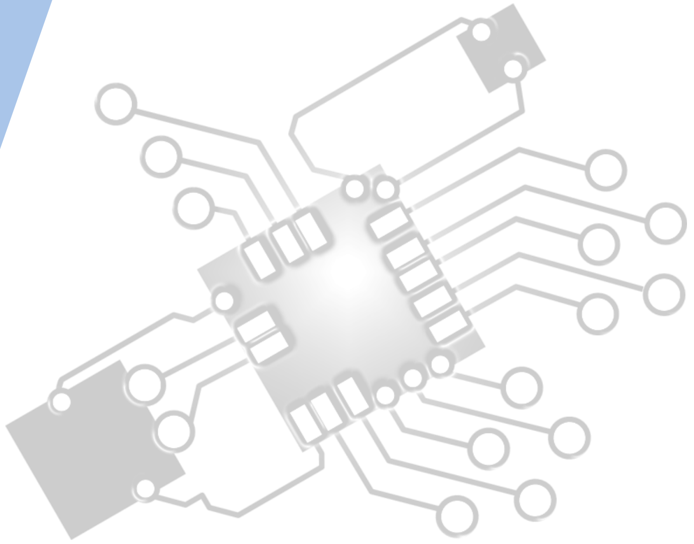
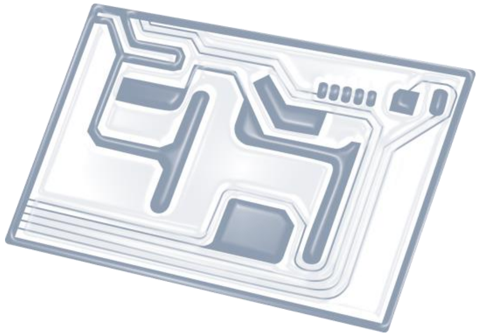
Features of OOP

IB Computer Science



Content developed by
Dartford Grammar School
Computer Science Department

HL Topics 1-7, D1-4



1: System design

2: Computer Organisation

3: Networks

4: Computational thinking

5: Abstract data structures

6: Resource management

7: Control

D: OOP

HL & SL D.2 Overview

D.2 Features of OOP

D.2.1 Define the term encapsulation

D.2.2 Define the term inheritance

D.2.3 Define the term polymorphism

D.2.4 Explain the advantages of encapsulation

D.2.5 Explain the advantages of inheritance

D.2.6 Explain the advantages of polymorphism

D.2.7 Describe the advantages of libraries of objects

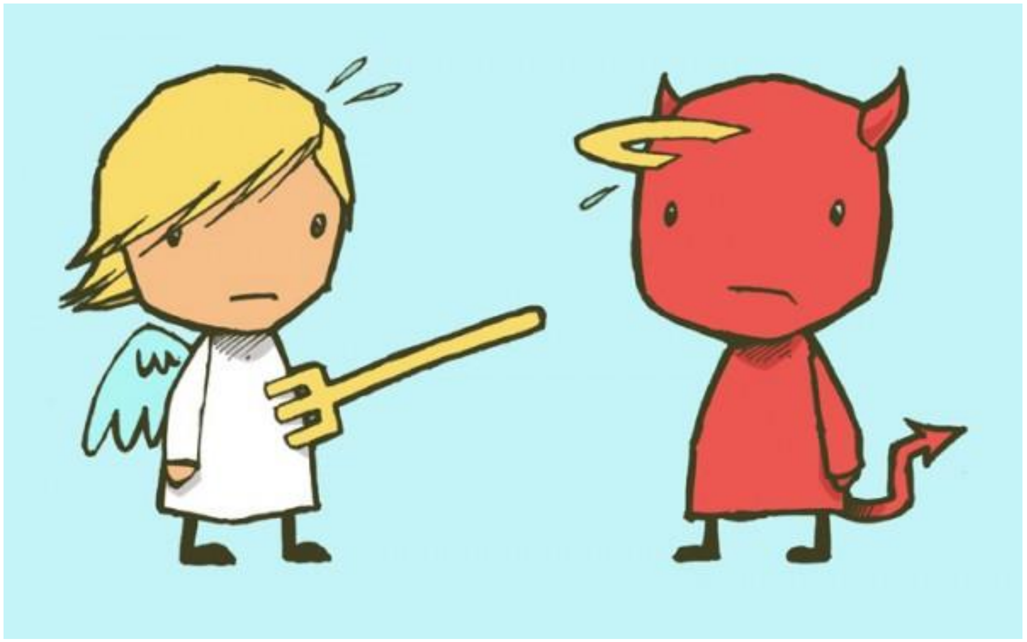
D.2.8 Describe the disadvantages of OOP

D.2.9 Discuss the use of programming teams

D.2.10 Explain the advantages of modularity in program development

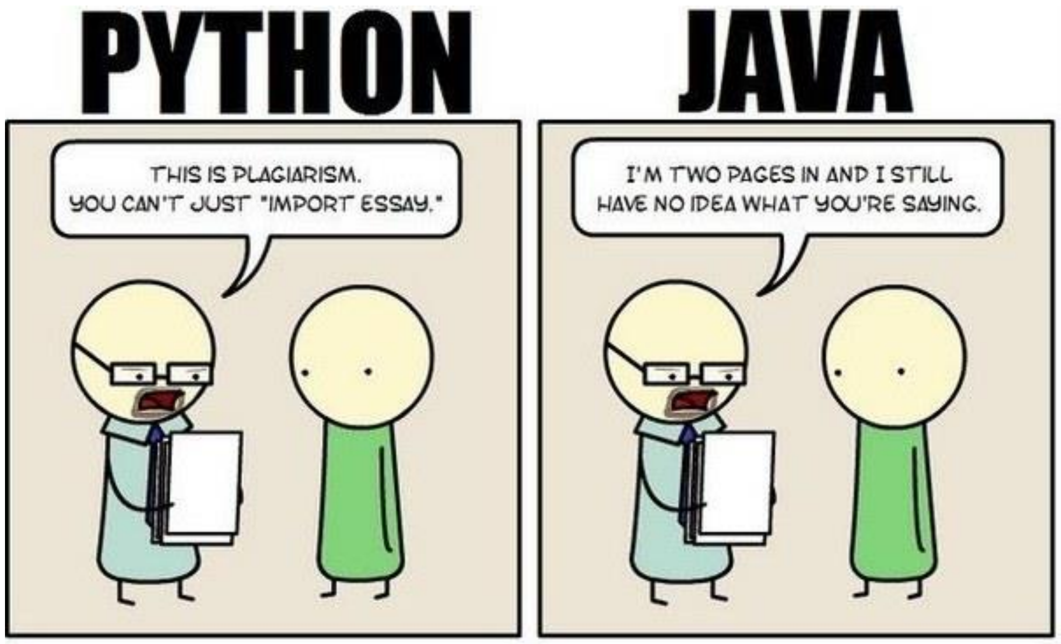
Topic D.2.8

Describe the **disadvantages** of OOP



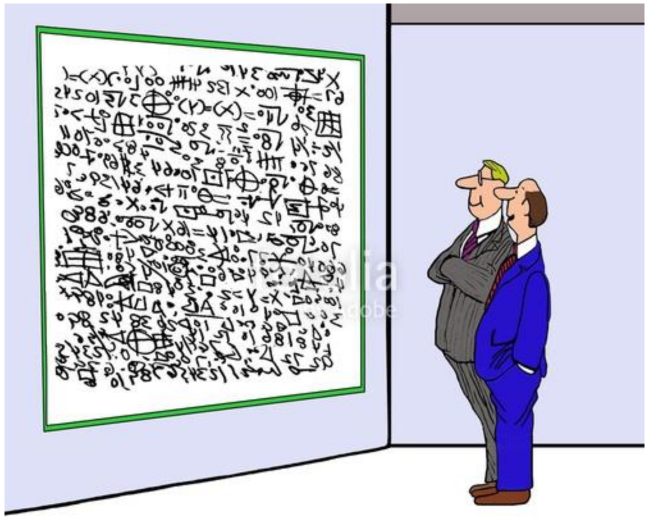
Disadvantages of OOP

A. **Increased complexity** for small problems
B. **Unsuited** to particular classes of problems


PYTHON
THIS IS PLAGIARISM. YOU CAN'T JUST "IMPORT ESSAY."
JAVA
I'M TWO PAGES IN AND I STILL HAVE NO IDEA WHAT YOU'RE SAYING.

A. Increased complexity

- OOP typically involve more lines of code than procedural programs.
- OOP are typically slower than procedure-based programs, as they typically require more instructions to be executed.



"When you put it like that, it makes complete sense."

B. Unsuited to particular problems

- There are problems that lend themselves well to functional-programming style, logic-programming style, or procedure-based programming style, and applying object-oriented programming in those situations will not result in efficient programs.
- These problems tend to small and involving only one data source.

  Example: *Why make an object when using a String will do?*

Non-OOP vs OOP

```
class Student{
    private String name;

    Student(){
        name = "none";
    }

    Student(){
        //default constructor
    }

    public String getName(){
        return name;
    }
    public void setName(String n){
        name = n;
    }
}
```

All of that...

```
class RunName{
    public static void main(String args[]){
        Student s = new Student("Alex");
        System.out.println( s.getName() );
    }
}
```

Does the same as this...

```
class NonOOP{
    public static void main(String args[]){
        String name = "Alex";
        System.out.println(name);
    }
}
```