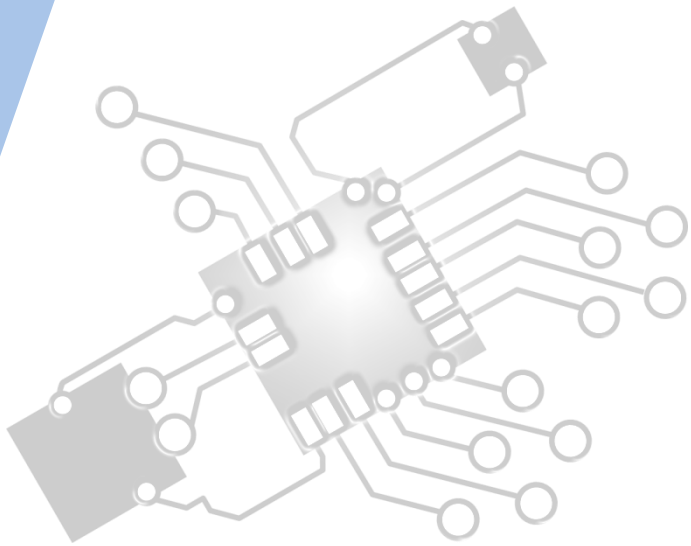
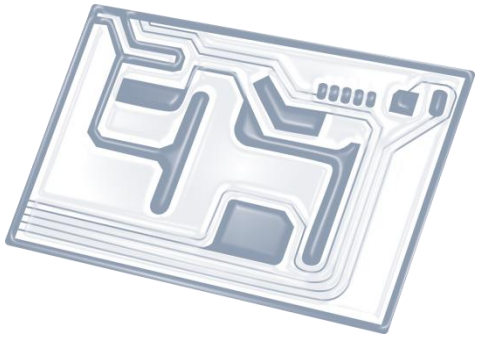
Computational thinking, problem-solving and programming: General Principals

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 4.1 Overview

Thinking procedurally

4.1.1 Identify the procedure appropriate to solving a problem

4.1.2 Evaluate whether the order in which activities are undertaken will result in the required outcome

4.1.3 Explain the role of sub-procedures in solving a problem

Thinking logically

4.1.4 Identify when decision-making is required in a specified situation

4.1.5 Identify the decisions required for the solution to a specified problem

4.1.6 Identify the condition associated with a given decision in a specified problem

4.1.7 Explain the relationship between the decisions and conditions of a system

4.1.8 Deduce logical rules for real-world situations

Thinking ahead

4.1.9 Identify the inputs and outputs required in a solution

4.1.10 Identify pre-planning in a suggested problem and solution

4.1.11 Explain the need for pre-conditions when executing an algorithm

4.1.12 Outline the pre- and post-conditions to a specified problem

4.1.13 Identify exceptions that need to be considered in a specified problem solution

Thinking concurrently

4.1.14 Identify the parts of a solution that could be implemented concurrently

4.1.15 Describe how concurrent processing can be used to solve a problem

4.1.16 Evaluate the decision to use concurrent processing in solving a problem

Thinking abstractly

4.1.17 Identify examples of abstraction

4.1.18 Explain why abstraction is required in the derivation of computational solutions for a specified situation

4.1.19 Construct an abstraction from a specified situation

4.1.20 Distinguish between a real-world entity and its abstraction

1: System design

2: Computer Organisation

3: Networks

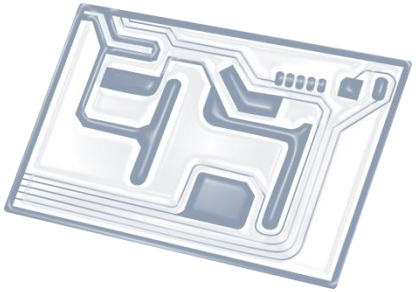
4: Computational thinking

5: Abstract data structures

6: Resource management

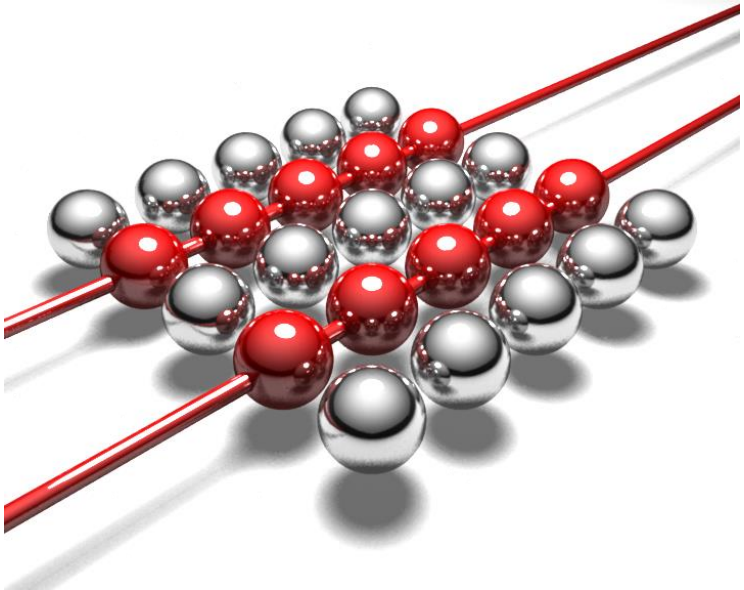
7: Control

D: OOP



Topic 4.1.15

Describe how **concurrent processing** can be used to solve a problem



Teacher notes:

- *Examples: building a house, production lines, division of labour*

- *Students will not be expected to construct a flowchart or pseudocode relating to concurrent processing.*

Concurrent Programming

- **Concurrent processing system** -- one job uses several processors to execute sets of instructions in parallel.
  - Requires a programming language and a computer system that can support this type of construct.
- Increases computation speed.
- Increases complexity of programming language and hardware (machinery & communication among machines).
- Reduces complexity of working with array operations within loops, of performing matrix multiplication, of conducting parallel searches in databases, and of sorting or merging files.