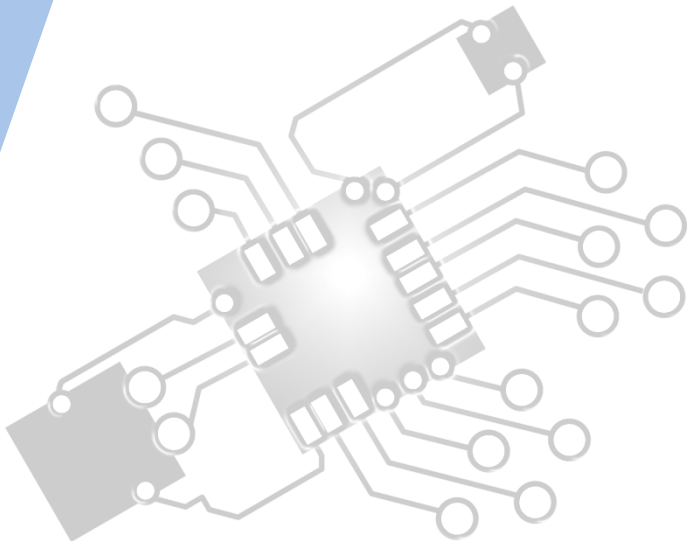




Computational thinking, problem-solving and programming:

Connecting computational thinking and program design

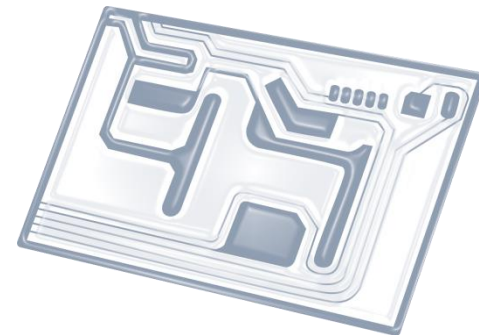
IB Computer Science



*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



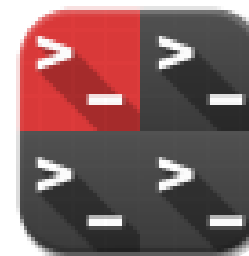
5: Abstract data structures



6: Resource management



7: Control



D: OOP

HL & SL 4.2 Overview

- 4.2.1 Describe the characteristics of standard algorithms on linear arrays
- 4.2.2 Outline the standard operations of collections
- 4.2.3 Discuss an algorithm to solve a specific problem
- 4.2.4 Analyse an algorithm presented as a flow chart
- 4.2.5 Analyse an algorithm presented as pseudocode
- 4.2.6 Construct pseudocode to represent an algorithm
- 4.2.7 Suggest suitable algorithms to solve a specific problem
- 4.2.8 Deduce the efficiency of an algorithm in the context of its use
- 4.2.9 Determine the number of times a step in an algorithm will be performed for given input data



1: System design

2: Computer Organisation



3: Networks

4: Computational thinking



5: Abstract data structures

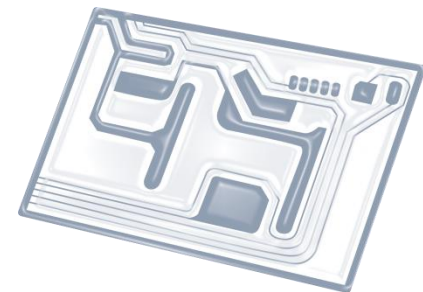
6: Resource management



7: Control

D: OOP

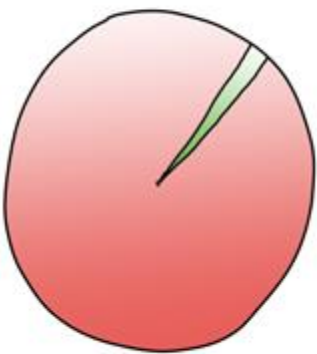




Topic 4.2.5


Analyse an algorithm presented in pseudocode

THE USAGE OF PSEUDOCODE IN REAL LIFE



A pie chart with a red base and a small green slice, representing the usage of pseudocode in real life.

- DESCRIBING AN ALGORITHM
- A TOOL THAT FRESHMAN COMPUTER SCIENCE STUDENTS THAT JUST STARTED TO LEARN PROGRAMMING USES TO EXPRESS THEIR DUMB ACTIONS



A screenshot of a tweet from Derp Johnson (@DerpyJohn). The tweet contains a block of pseudocode for a loop that increments 'alcohol' and 'dance' until it is morning. The tweet has 48 retweets and 213 favorites.

```
while (!morning){  
  alcohol++  
  dance++  
} #mylife #partyhard
```

RETWEETS 48 FAVORITES 213

ctp200.com

Teacher's notes:

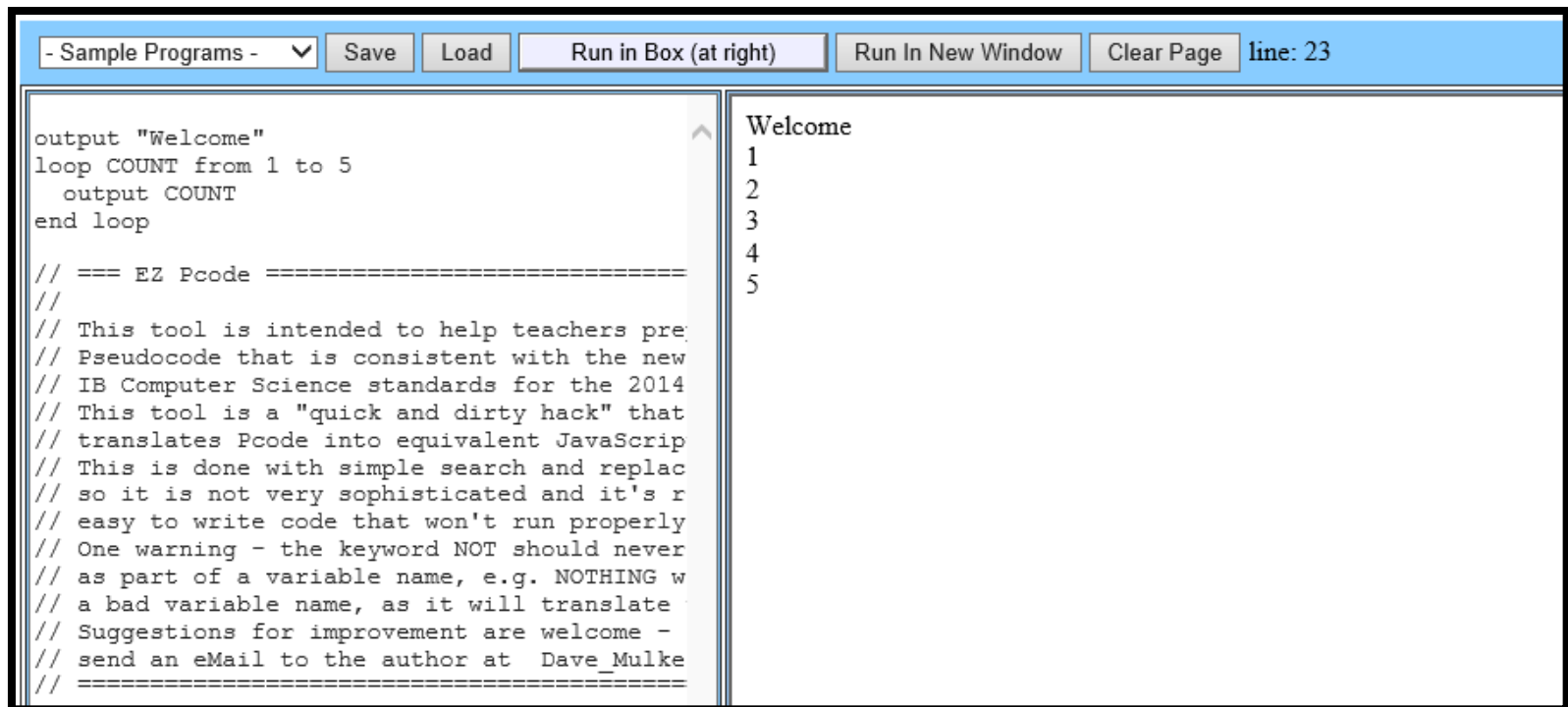
- Examination questions may involve **variables, calculations, simple and nested loops, simple conditionals and multiple or nested conditionals.**
- This would include **tracing an algorithm as well as assessing its correctness.**



Best method: **PRACTICE THIS!**

Use the *D. Mulkey's* **ONLINE PSEUDO CODE GENERATOR:**

<https://dl.dropboxusercontent.com/u/275979/ibcomp/pseudocode/pcode.html>

A screenshot of a web-based pseudo code generator. The interface has a light blue header with a dropdown menu set to '- Sample Programs -', and buttons for 'Save', 'Load', 'Run in Box (at right)', 'Run In New Window', and 'Clear Page'. A 'line: 23' indicator is on the right. The main area is split into two panes. The left pane contains pseudo code: 'output "Welcome"', 'loop COUNT from 1 to 5', ' output COUNT', 'end loop', followed by a large block of comments starting with '// === EZ Pcode ==='. The right pane shows the output: 'Welcome' followed by a vertical list of numbers 1 through 5.

```
output "Welcome"
loop COUNT from 1 to 5
  output COUNT
end loop

// === EZ Pcode ===
//
// This tool is intended to help teachers pre
// Pseudocode that is consistent with the new
// IB Computer Science standards for the 2014
// This tool is a "quick and dirty hack" that
// translates Pcode into equivalent JavaScrip
// This is done with simple search and replac
// so it is not very sophisticated and it's r
// easy to write code that won't run properly
// One warning - the keyword NOT should never
// as part of a variable name, e.g. NOTHING w
// a bad variable name, as it will translate
// Suggestions for improvement are welcome -
// send an eMail to the author at Dave_Mulke
// =====
```

Welcome
1
2
3
4
5