Pseudo Code

PROGRAMMING ON PAPER FOR IB PAPER 1 EXAMS
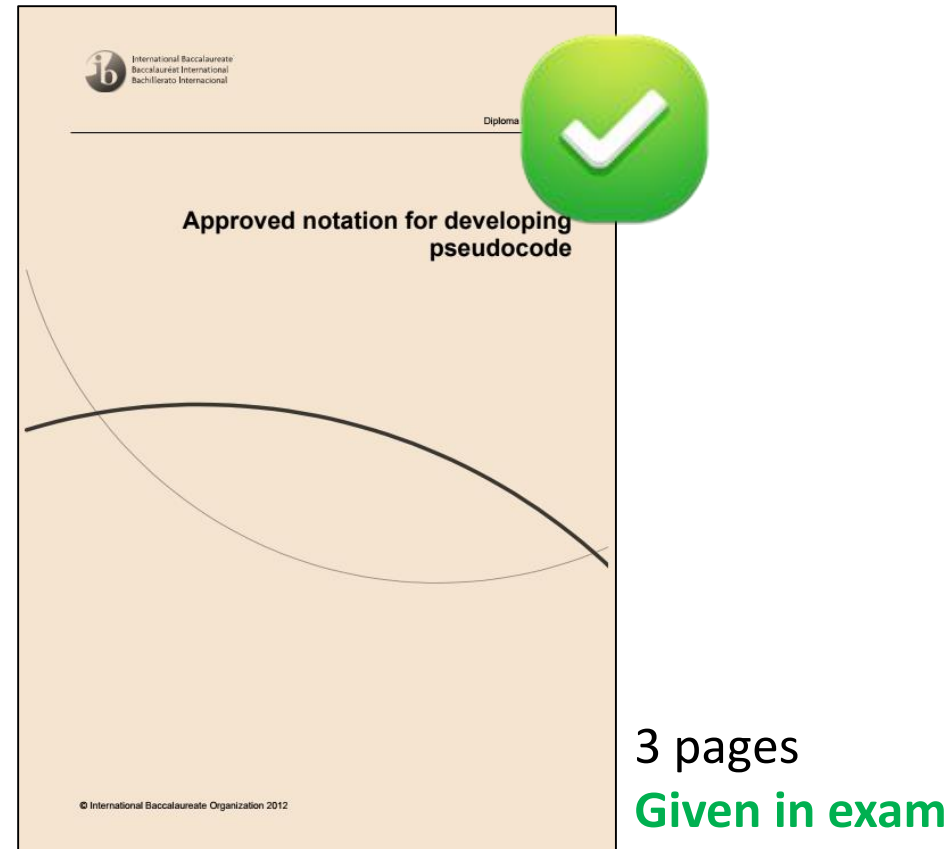
SESSION 2

2 official pseudo code guides

Computer Science
First Exams 2014

Pseudocode in Examinations
- *Standard Data Structures*
- *Examples of Pseudocode*

Candidates are NOT allowed a copy of this document during their examinations.

Approved notation for developing pseudocode
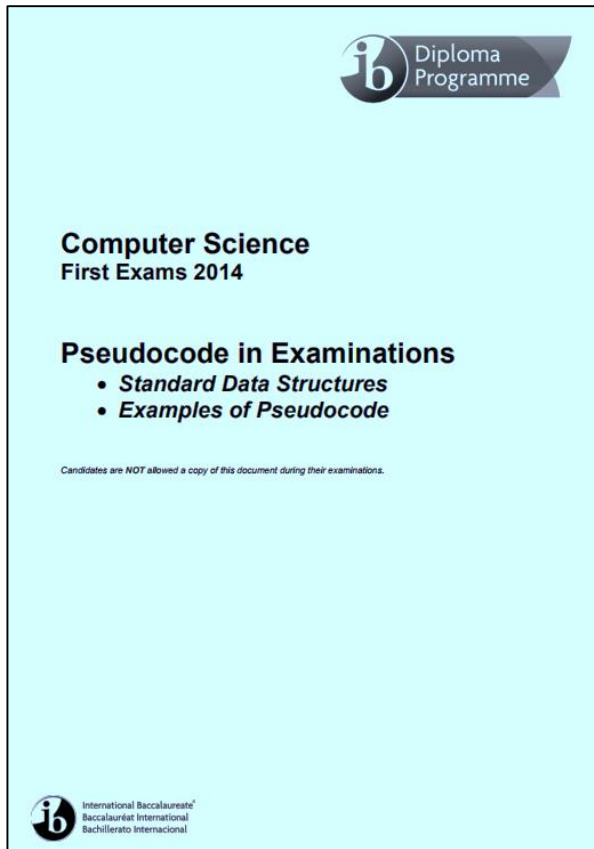
8 pages
NOT given in exam

3 pages
Given in exam

Warning!

Pseudo code questions are *never* as overt or obvious as the examples we discuss in these sessions.

These examples are only there to teach you the *skills* you need to answer more complex problems.

Topics 4, 5 and 7 can include pseudo code…

Top pseudo code tips

- ✓ When possible, start answering a pseudo code question at the top of a page

- ✓ Write pseudo code in pencil first and then copy into pen

- ✓ Think about the data types and associated access methods BEFORE writing anything

- ✓ Think about what control structures (especially loops) are associated with that data type

- ✓ Be sure to return or output something at the end of the problem – even if they don't ask for it!

Basic structure of ALL pseudo code questions

```
Declarations / Initialisations

Control structures / Calculations

Output / Return
```

You get marks for individual sections, not the final output.

This means you could well get 7/8 even if your final output is not right.

T1: Move selected data from collection to queue

You have a collection of numbers called NUMS (containing integer values between 1 and 200).

Transfer all even numbers in NUMS to a newly initialized queue called Q.

```
1  loop while NUMS.hasNext()
2      NUM = NUMS.getNext()
3      if (NUM mod 2 == 0) then
4          Q.enqueue(NUM)
5      end if
6  end loop
7
8  return Q
```

T2: Conditional average of a stack

You have a stack M that contains double values between 0 and 1.

Work out the average number in the stack, but only display it if it is less than 0.5.

```
 1 COUNT = 0
 2 SUM = 0
 3 AVERAGE = 0
 4
 5 loop while ! M.isEmpty()
 6     SUM = SUM + M.pop()
 7     COUNT = COUNT + 1
 8 end loop
 9
10 AVERAGE = SUM / COUNT
11 if AVERAGE < 0.5 then
12     output AVERAGE
13 end if
```

You have a 2D array HAMSTER with 5 columns and 3 rows containing Strings with a value of either "Brown", "White" or "Mottled".

Copy all "Brown" values to a queue called B and return how many elements have been added.

```
 1 BCOUNT = 0
 2
 3 loop I from 0 to 2
 4     loop J from 0 to 4
 5         if HAMSTER[I][J] == "Brown" then
 6             B.enqueue(HAMSTER[I][J])
 7             BCOUNT = BCOUNT + 1
 8         end if
 9     end loop
10 end loop
11
12 return BCOUNT
```

T4: Remove selected values from a queue

You have two queues, Q1 which is full of integer values, and Q2 which is empty.

Move all even numbers from Q1 to Q2 in such a way that by the end you have all odd numbers in Q1 and even numbers in Q2.

```
 1  TEMP = 0
 2  linkedlist L = new linkedlist()
 3
 4  loop while not Q1.isEmpty()
 5      TEMP = Q1.dequeue()
 6      if TEMP mod 2 == 0 then
 7          Q2.enqueue(TEMP)
 8      else
 9          L.addItem(TEMP)
10      end if
11  end loop
12
13  loop while L.hasNext()
14      Q1.enqueue(L.getNext() )
15  end loop
```

T5: Move data from 2D array to collection

	[0]	[1]	[2]	[3]	[4]	[5]	[6]
[0]	5.3	8.0	14.7	37.3	14.7	8.0	12.0
[1]	29.3	10.7	21.3	14.7	22.7	21.3	17.3
[2]	24.0	29.3	18.7	28.0	26.7	34.7	13.3
[3]	10.7	12.0	21.3	14.7	32.0	18.7	30.7
[4]	10.7	4.0	33.3	30.7	30.7	26.7	37.3
[5]	34.7	20.0	37.3	14.7	28.0	29.3	32.0
[6]	21.3	37.3	37.3	4.0	36.0	13.3	37.3
[7]	21.3	9.3	25.3	32.0	26.7	6.7	29.3
[8]	17.3	6.7	33.3	40.0	28.0	26.7	30.7

You have a 2D array called GRID with 7 columns and 9 rows filled with double values.

Move all the data, one column at a time to a new collection called GLUE.

```
1  collection GLUE = new collection()
2
3  loop X from 0 to 6
4      loop Y from 0 to 8
5          GLUE.addItem(GRID[Y][X])
6      end loop
7  end loop
8
9  return GLUE
```

T6: Reversing a stack using a queue

You have a stack S and a queue Q. Reverse the stack using the queue.

No output/return required.

```
1 loop while not S.isEmpty()
2      Q.enqueue(S.pop() )
3 end loop
4
5 loop while not Q.isEmpty()
6      S.push(Q.dequeue() )
7 end loop
```