

Cambridge  
International  
AS & A Level

**Cambridge International Examinations**  
Cambridge International Advanced Subsidiary and Advanced Level

---

**COMPUTER SCIENCE**

**9608/21**

Paper 2 Fundamental Problem-solving and Programming Skills

**May/June 2018**

PRE-RELEASE MATERIAL

No Additional Materials are required.

**This material should be given to the relevant teachers and candidates as soon as it has been received at the Centre.**

---

**READ THESE INSTRUCTIONS FIRST**

Candidates should use this material in preparation for the examination. Candidates should attempt the practical programming tasks using their chosen high-level, procedural programming language.

---

This document consists of **6** printed pages and **2** blank pages.

Teachers and candidates should read this material prior to the June 2018 examination for 9608 Paper 2.

## Reminders

The syllabus states:

- there will be questions on the examination paper which do not relate to this pre-release material.
- you must choose a high-level programming language from this list:
  - Visual Basic (console mode)
  - Python
  - Pascal / Delphi (console mode)

**Note:** A mark of **zero** will be awarded if a programming language other than those listed is used.

Questions on the examination paper may ask the candidate to write:

- structured English
- pseudocode
- program code

A program flowchart should be considered as an alternative to pseudocode for documenting an algorithm design.

Candidates should be confident with:

- the presentation of an algorithm using either a program flowchart or pseudocode
- the production of a program flowchart from given pseudocode and vice versa

There is an **Appendix** at the end of this document. Some tasks refer you to this information. There will also be a similar appendix at the end of the question paper.

## Declaration of variables

The syllabus document shows the syntax expected for a declaration statement in pseudocode.

```
DECLARE <identifier> : <data type>
```

If Python is the chosen language, each variable's identifier (name) and its intended data type must be documented using a comment statement.

## Structured English – Variables

An algorithm in pseudocode uses variables, which should be declared. An algorithm in structured English does not always use variables. In this case, the candidate needs to use the information given in the question to complete an identifier table. The table needs to contain an identifier, data type and description for each variable.

## TASK 1 – Structured English, flowcharts and program code

Consider examples of processes in a variety of areas, such as:

- clubs or hobbies
- college or school
- home or factory

### TASK 1.1

Write an algorithm, using structured English, to describe each of these processes.

**Key focus:**

**Problem decomposition**

### TASK 1.2

Split a process into sub-tasks and consider the advantages of this approach.

### TASK 1.3

Convert an algorithm written in structured English into a program flowchart.

You will need to include some or all of the following when constructing a flowchart.

- input
- output
- iteration (conditional or count-controlled)
- selection (using IF or CASE statements)
- sequence
- calls to subroutines

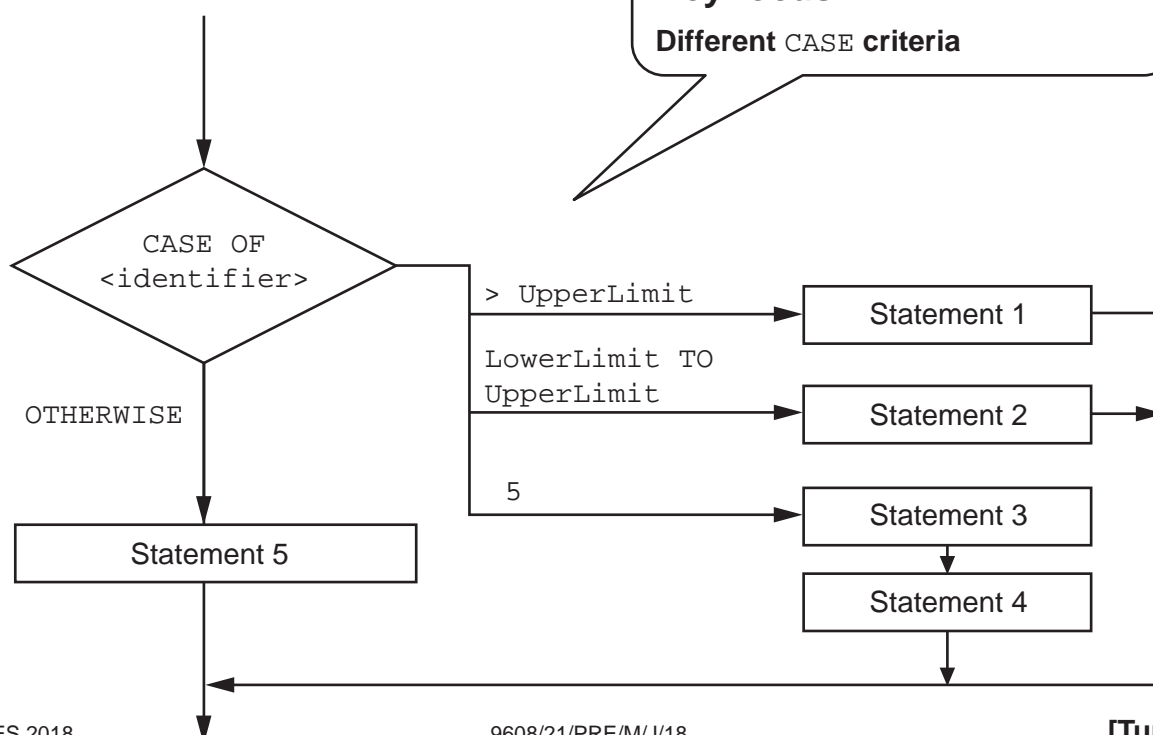
**Key focus:**

**Program flowchart**

Note: Conditions within a CASE statement may contain a range of criteria as illustrated in the following flowchart.

**Key focus:**

**Different CASE criteria**



**TASK 1.4**

Convert the program flowchart into **pseudocode** and then into **program code**. A list of typical pseudocode functions can be found in the **Appendix**.



**Key focus:**

**Pseudocode and program code**

**TASK 1.5**

Consider the different types of error that can occur at different stages of the development cycle.



**Key focus:**

**Errors**

**TASK 1.6**

Consider the advantages of modular programming using both built-in and user-defined functions.



**Key focus:**

**Modular programming**

**TASK 1.7**

Consider ways of testing the complete program, both with and without knowledge of the underlying program code.



**Key focus:**

**Testing**

Consider ways of testing the main program before all of the user-defined functions have been completed.

## TASK 2 – Structure charts and procedure declarations

### TASK 2.1

**Key focus:**

**Producing a structure chart**

Produce a structure chart for a simple modular program. Include one top-level module and at least two lower-level modules.

Add data arrows for both Boolean and non-Boolean variables.

### TASK 2.2

Use the chart to produce **pseudocode** declarations for the modules.

### TASK 2.3

**Key focus:**

**Good programming practice**

Convert the pseudocode into **program code**.

Consider features of the program code that make it easier to read and understand.

### TASK 2.4

Produce pseudocode declarations for a different set of modules, with one top-level module and at least two lower-level modules, using different sets of parameters.

### TASK 2.5

Produce a structure chart from the pseudocode procedures produced in Task 2.4.

**Key focus:**

**Interpret modular code to produce a structure chart**

## Appendix

### Built-in functions (pseudocode)

Each function returns an error if the function call is not properly formed.

MID(ThisString : STRING, x : INTEGER, y : INTEGER) RETURNS STRING  
returns a string of length y starting at position x from ThisString

Example: MID("ABCDEFGH", 2, 3) returns string "BCD"

LENGTH(ThisString : STRING) RETURNS INTEGER  
returns the integer value representing the length of string ThisString

Example: LENGTH("Happy Days") returns 10

LEFT(ThisString : STRING, x : INTEGER) RETURNS STRING  
returns leftmost x characters from ThisString

Example: LEFT("ABCDEFGH", 3) returns string "ABC"

RIGHT(ThisString : STRING, x : INTEGER) RETURNS STRING  
returns rightmost x characters from ThisString

Example: RIGHT("ABCDEFGH", 4) returns string "EFGH"

INT(x : REAL) RETURNS INTEGER  
returns the integer part of x

Example: INT(27.5415) returns 27

ASC(ThisChar : CHAR) RETURNS INTEGER  
returns the ASCII value of character ThisChar

Example: ASC('A') returns 65

MOD(ThisNum : INTEGER, ThisDiv : INTEGER) RETURNS INTEGER  
returns the integer value representing the remainder when ThisNum is divided by ThisDiv

Example: MOD(10, 3) returns 1

### Operators (pseudocode)

| Operator | Description  |
|----------|--|
| &        | Concatenates (joins) two strings.<br>Example: "Summer" & " " & "Pudding" produces "Summer Pudding" |
| AND      | Performs a logical AND on two Boolean values.<br>Example: TRUE AND FALSE produces FALSE            |
| OR       | Performs a logical OR on two Boolean values.<br>Example: TRUE OR FALSE produces TRUE               |

**BLANK PAGE**

**BLANK PAGE**

---

Permission to reproduce items where third-party owned material protected by copyright is included has been sought and cleared where possible. Every reasonable effort has been made by the publisher (UCLES) to trace copyright holders, but if any items requiring clearance have unwittingly been included, the publisher will be pleased to make amends at the earliest possible opportunity.

To avoid the issue of disclosure of answer-related information to candidates, all copyright acknowledgements are reproduced online in the Cambridge International Examinations Copyright Acknowledgements Booklet. This is produced for each series of examinations and is freely available to download at [www.cie.org.uk](http://www.cie.org.uk) after the live examination series.

Cambridge International Examinations is part of the Cambridge Assessment Group. Cambridge Assessment is the brand name of University of Cambridge Local Examinations Syndicate (UCLES), which is itself a department of the University of Cambridge.