Cambridge Assessment International Education

Cambridge International AS & A Level

COMPUTER SCIENCE **9608/42**

Paper 4 Further Problem-solving and Programming Skills **May/June 2020**

PRE-RELEASE MATERIAL

2428180562

No additional materials are needed.

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

INSTRUCTIONS

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

This document has **8** pages. Blank pages are indicated.

DC (PQ) 198991

Teachers and candidates should read this material prior to the June 2020 examination for 9608 Paper 4.

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.

The practical skills for Paper 4 build on the practical skills covered in Paper 2. We therefore recommend that candidates choose the same high-level programming language for this paper as they did for Paper 2. This will give candidates the opportunity for extensive practice and allow them to acquire sufficient expertise.

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 a high-level 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

Candidates will also benefit from using pre-release materials from previous examinations.
These are available on the teacher support site.

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 – Declarative programming

A knowledge base contains information about students in a class and who their friends are. A declarative programming language is used to query the knowledge base.

Some clauses in the knowledge base are shown.

Key focus: Declarative programming

```
01 person(william).
02 person(edith).
03 person(gabrielle).
04 person(raj).
05 person(sasha).
06 person(andrei).
07 friend(william, raj).
08 friend(raj, gabrielle).
```

Clause	Explanation
`01`	William is a person
`07`	William is a friend of Raj

TASK 1.1

Two new students are joining the class. They need to be added to the knowledge base.

Write the clauses to add two new people to the knowledge base.

TASK 1.2

Sasha makes friends with one of the new students.

Add a clause that states Sasha is a friend of one of the new students.

TASK 1.3

A new type of clause `notfriend()` is added to the knowledge base.

Add a clause that states Andrei is not a friend of Gabrielle.

TASK 1.4

Explain why the following two clauses would be needed in the knowledge base:

```
friend(william, raj).

friend(raj, william).
```

TASK 1.5

Write a query to find all the friends of a person.

TASK 2 – Hash tables

Jade stores the details for her customers in a hash table. Each customer has a unique six-character code as a customer ID. Customer IDs start at AA0001 and range up to ZZ9999. A hashing function is used to create an index to store each customer. The hash table is implemented using a 1D array.

Key focus: Hashing functions

TASK 2.1

Discuss why a hashing function would be used to directly access a customer ID.

TASK 2.2

Develop an algorithm to implement the hashing function. Make sure that every index of the array can be directly addressed

Write **pseudocode** for your algorithm.

TASK 2.3

Implement your algorithm in the programming language of your choice.

TASK 2.4

Expand your program to add 10 customer IDs to the hash table, to test your program. Your program should call the hashing function to do this.

TASK 2.5

Develop an algorithm to search the hash table for a customer ID.

Write **pseudocode** for your algorithm.

TASK 2.6

Implement your search algorithm in **TASK 2.5** in the programming language of your choice.

TASK 2.7

Test your program using two of the previous customer IDs you added to the hash table.

Test your program using a customer ID that is not in the hash table.

TASK 2.8

It is possible that collisions could occur when using a hashing function.

Discuss what is meant by a collision, how they can occur and how they can be resolved.

TASK 3 – Recursion

Key focus: Recursion

Recursion can be used when creating programs.

TASK 3.1

Discuss what is meant by the term recursion.

TASK 3.2

Consider the following pseudocode algorithm:

```
PROCEDURE NewYearCountdown(Value : INTEGER)
    IF Value = 0
       THEN
          OUTPUT "Happy New Year"
       ELSE
          Value ← Value - 1
          OUTPUT Value
          NewYearCountdown(Value)
    ENDIF
ENDPROCEDURE
```

In the algorithm, label the base case, the general case and the recursive call.

TASK 3.3

Recursive algorithms can be rewritten as an iterative loop.

Rewrite the recursive algorithm given in **TASK 3.2** as an iterative loop.

TASK 3.4

Using **pseudocode**, write a recursive algorithm that would output each letter in a string, from the beginning, one at a time. The string is passed as a parameter.

TASK 3.5

Implement your algorithm in the programming language of your choice.

TASK 3.6

Discuss why a programmer may choose to create a recursive algorithm rather than using an iterative loop structure.

BLANK PAGE

BLANK PAGE

BLANK PAGE