

CANDIDATE
NAME

--

CENTRE
NUMBER

--	--	--	--	--

CANDIDATE
NUMBER

--	--	--	--

COMPUTER SCIENCE

2210/21

Paper 2 Problem-solving and Programming

May/June 2018

1 hour 45 minutes

Candidates answer on the Question Paper.

No Additional Materials are required.

No calculators allowed.

READ THESE INSTRUCTIONS FIRST

Write your Centre number, candidate number and name in the spaces at the top of this page.

Write in dark blue or black pen.

You may use an HB pencil for any diagrams, graphs or rough working.

Do not use staples, paper clips, glue or correction fluid.

DO NOT WRITE IN ANY BARCODES.

Answer **all** questions.

DO NOT ATTEMPT TASKS 1, 2 AND 3 in the pre-release material; these are for information only.

You are advised to spend no more than **40 minutes** on **Section A** (Question 1).

No marks will be awarded for using brand names of software packages or hardware.

At the end of the examination, **fasten** all your work securely together.

The number of marks is given in brackets [] at the end of each question or part question.

The maximum number of marks is 50.

This document consists of 11 printed pages and 1 blank page.

Section A

You are advised to spend no longer than 40 minutes answering this section.

Here is a copy of the pre-release material.

DO NOT attempt Tasks 1, 2 and 3 now.

Use the pre-release material and your experience from attempting the tasks before the examination to answer Question 1.

Pre-release material

A computer shop will build a computer from components to meet a customer's requirements. For each request for a computer to be built, an estimate of the cost is produced. The component stock level is checked; if all the components are in stock, a firm order to build the computer can be placed. A program is required to work out the cost of the computer, update the stock levels and provide a daily summary of orders for the shop owner.

Write and test a program or programs for the computer shop owner.

- Your program or programs must include appropriate prompts for the entry of data.
- Error messages and other output need to be set out clearly and understandably.
- All variables, constants and other identifiers must have meaningful names.

You will need to complete these **three** tasks. Each task must be fully tested.

TASK 1 – Produce an estimate.

Write a program for TASK 1 to calculate the cost of building a computer using these components.

Component	Choices	Prices in \$
Processor	p3 / p5 / p7	100 / 120 / 200
RAM	16GB / 32 GB	75 / 150
Storage	1TB / 2TB	50 / 100
Screen	19" / 23"	65 / 120
Case	Mini Tower / Midi Tower	40 / 70
USB ports	2 ports / 4 ports	10 / 20

The customer makes a choice for each component and an estimate is produced. The estimate must show a unique estimate number, the components chosen and the price of each component. The estimate must also show the total cost of the computer, which is calculated as the sum of the cost of the components chosen plus 20%.

TASK 2 – Place an order.

Using your estimate from TASK 1, check if the components required are in stock. If all the components are in stock then update the stock levels. Add the unique estimate number to the list of order numbers. Add the customer's details and today's date to the estimate details to finalise the order. Print two copies of the order, one for the customer and one for the shop.

TASK 3 – Summarise the day's orders.

Extend TASK 2 to provide an end of day summary showing the number of orders made, the total number of each component sold and the value of the orders.

1 (a) All variables, constants and other identifiers should have meaningful names.

- (i) You recorded information for the estimate of the cost of building a computer in **Task 1**. Give a data structure that you created for **Task 1**, its name, data type and use.

Data structure Array

Name Processors

Data type String

Use To store/hold the names of the types of processors available. [4]

- (ii) Describe the data structures that you have used in **Task 2** to record the customer details. Include sample data in the description.

Processor Array - to store the type of processors
type string

Ram Array - to store the type of RAMs available
type string

Example Processor[Index] ← 'P3'

Ram[Index] ← '16 GB'

Variable - Number of Components ← 3 [4]

To hold the number of components selected

- (b) Explain how your program for **Task 1** produces a unique estimate number.

Ask the user to enter a unique number. The program checks in the already existing numbers if that number has been taken. If it hasn't then it is used as the new number. [2]

- (c) Write an algorithm for part of **Task 2** to check that the chosen **processor** and chosen **RAM** are in stock, using **either** pseudocode, programming statements **or** a flowchart. Assume that **Task 1** has been completed. Do **not** check the other components or produce the order.

FoundProc $\leftarrow$ False

Count $\leftarrow$ 1

WHILE Not FoundProc AND Count $\leq$ 3 DO

IF Processor[TestNo] = proc[Count] AND StProc(Count) > 0 Then

FoundProc $\leftarrow$ True

ENDIF

Count $\leftarrow$ Count + 1

ENDWHILE

IF FoundProc $\neq$ True Then

FoundRAM $\leftarrow$ False

IF RAM[TestNo] = RAM1 AND STAM1 > 0 Then

FoundRAM $\leftarrow$ True

STAM1 $\leftarrow$ STAM1 - 1

ENDIF

IF RAM[TestNo] = RAM2 AND STAM2 > 0 Then

FoundRAM $\leftarrow$ True

STAM2 $\leftarrow$ STAM2 - 1

ENDIF

ENDIF

IF Not FoundProc Then

Output "Processor out of stock"

Else

stproc(Count) $\leftarrow$ Stproc(Count) - 1

[5]

ENDIF

- (d) Explain how your program completes **Task 3**. Any programming statements used in your answer must be fully explained.

Add a variable NumOfOrders that is incremented each time an order is made.

Add variables that increment by 1 for each component that is selected

Each time an order is completed you increment a value of the Total Cost of orders made.

[5]

Section B

- 2 (a) Write an algorithm to input 1000 numbers. Count how many numbers are positive and how many numbers are zero. Then output the results. Use **either** pseudocode **or** a flowchart.

Number $\leftarrow 0$, PosCount $\leftarrow 0$, ZeroCount $\leftarrow 0$

For Count $\leftarrow 1$ To 1000

OUTPUT "Enter A number"

INPUT Number

IF Number > 0 THEN

PosCount $\leftarrow$ PosCount + 1

END IF

IF Number $= 0$ THEN

ZeroCount $\leftarrow$ ZeroCount + 1

END IF

END FOR

OUTPUT "Number of Positive Numbers are",
PosCount

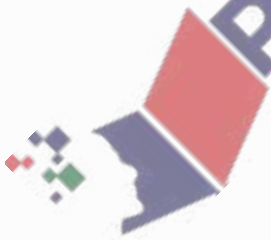
OUTPUT "Number of Zero Numbers are",
ZeroCount

- (b) Give one change you could make to your algorithm to ensure initial testing is more manageable.

Reduce the number of iterations to be more manageable. [1]

You don't need to enter 1000 values just to test if your program works

Question 3 starts on page 8.



- 3 The global trade item number (GTIN-8) barcode has seven digits and a check digit. This pseudocode algorithm inputs seven digits and calculates the eighth digit, then outputs the GTIN-8.

DIV(*X*, *Y*), finds the number of divides in division for example **DIV**(23, 10) is 2.

MOD(*X*, *Y*), finds the remainder in division for example **MOD**(23, 10) is 3.

```

FOR Count ← 1 TO 7
    INPUT Number
    Digit(Count) ← Number
NEXT
Sum ← (Digit(1)+Digit(3)+Digit(5)+Digit(7))*3+Digit(2)+Digit(4)+Digit(6)
IF MOD(Sum,10) <> 0
    THEN Digit(8) ← DIV(Sum,10)*10 + 10 - Sum
    ELSE Digit(8) ← 0
ENDIF
OUTPUT "GTIN-8"
FOR Count ← 1 TO 8
    OUTPUT Digit(Count)
NEXT

```

- (a) Complete the trace table for the input data: 5, 7, 0, 1, 2, 3, 4

Digit(1)	Digit(2)	Digit(3)	Digit(4)	Digit(5)	Digit(6)	Digit(7)	Digit(8)	Sum	OUTPUT
5	7	0	1	2	3	4	6	44	GTIN-8
									57012346

Complete the trace table for the input data: 4, 3, 1, 0, 2, 3, 1

Digit(1)	Digit(2)	Digit(3)	Digit(4)	Digit(5)	Digit(6)	Digit(7)	Digit(8)	Sum	OUTPUT
4	3	1	0	2	3	1	0	30	GTIN-8
									43102310

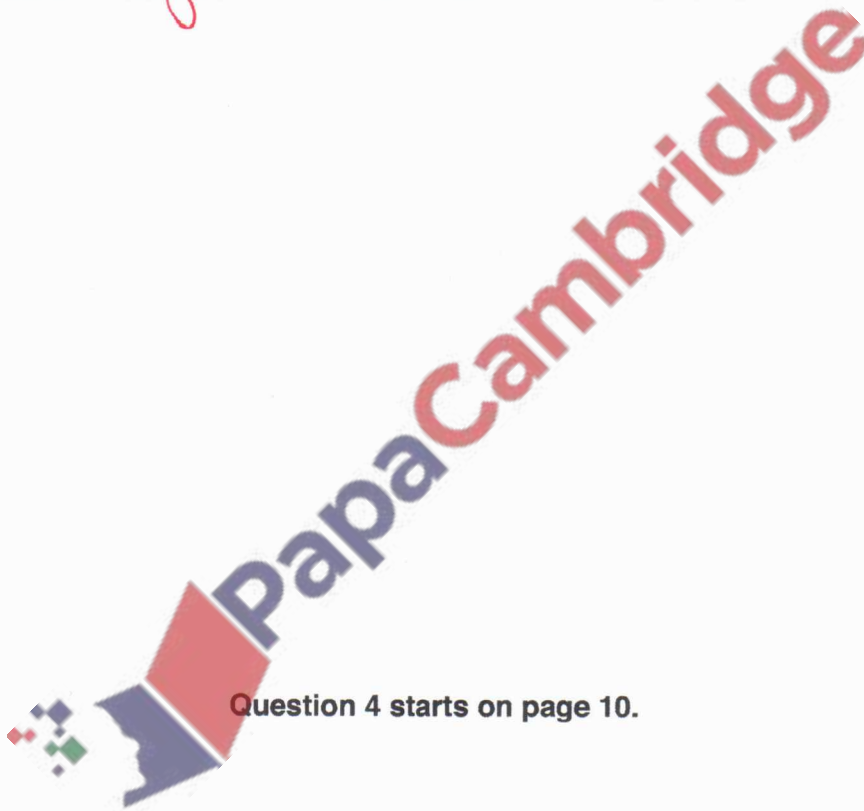
[5]

- (b) Explain how you would change the algorithm to input eight digits (seven digits and the check digit) and output if the check digit entered is correct or not.

change the first loop to 8 iterations to allow for eight digits to be entered
check that the eighth digit entered is equal to the calculated 8th digit.

If they are equal then output that the check digit is ok otherwise output "incorrect check digit"

[3]



Question 4 starts on page 10.

- 4 A programmer has written a routine to check that prices are below \$10.00. These values are used as test data.

10.00

9.99

ten

Explain why each value was chosen.

10.00 Boundary data to test if this is the edge of the acceptable data. to test if this data will be rejected

9.99 Normal data. To test if this data will be accepted

ten Abnormal data
To test if this data will be rejected.

[3]

- 5 Explain the difference between the programming concepts of **counting** and **totalling**. Include an example of a programming statement for each concept in your explanation.

Counting allows one to keep a running count of data as they are being entered. or as a process is executed. it adds by 1 (increments by 1)

$\text{Count} \leftarrow \text{Count} + 1$

Totalling allows one to keep a running total of values as they are being entered. it adds the value to a cumulative total
example

$\text{Total} \leftarrow \text{Total} + \text{Number.}$

[4]

- 6 A database table, PERFORMANCE, is used to keep a record of the performances at a local theatre.

Show Number	Type	Title	Date	Sold Out
SN091	Comedy	An Evening at Home	01 Sept	Yes
SN102	Drama	Old Places	02 Oct	No
SN113	Jazz	Acoustic Evening	03 Nov	No
SN124	Classical	Mozart Evening	04 Dec	Yes
SN021	Classical	Bach Favourites	01 Feb	Yes
SN032	Jazz	30 Years of Jazz	02 Mar	Yes
SN043	Comedy	Street Night	03 Apr	No
SN054	Comedy	Hoot	04 May	No

- (a) State the number of fields and records in the table.

Fields 5

Records 8

[2]

- (b) Give **two** validation checks that could be performed on the **Show Number** field.

Validation check 1 length check

Validation check 2 presence check

[2]

- (c) Using the query-by-example grid, write a query to identify jazz performances that are not sold out. Only display the date and the title.

Field:	Show Number	Type	Title	Date	Sold Out
Table:	Performance	Performance	Performance	Performance	Performance
Sort:					
Show:	☐	☐	☒	☒	☐
Criteria:		= Jazz			= No
or:					

[4]