Cambridge Assessment
International Education

Cambridge IGCSE™

CANDIDATE NAME	MyCSTutor.co.uk CS Worked Solutions		
CENTRE NUMBER	□□□□□	CANDIDATE NUMBER	□□□□

0123456789

COMPUTER SCIENCE **0478/02**

Paper 2 Algorithms, Programming and Logic **For examination from 2023**

SPECIMEN PAPER **1 hour 45 minutes**

You must answer on the question paper.

No additional materials are needed.

INSTRUCTIONS

- Answer **all** questions.
- Use a black or dark blue pen. You may use an HB pencil for any diagrams or graphs.
- Write your name, centre number and candidate number in the boxes at the top of the page.
- Write your answer to each question in the space provided.
- Do **not** use an erasable pen or correction fluid.
- Do **not** write on any bar codes.
- Calculators must **not** be used in this paper.

INFORMATION

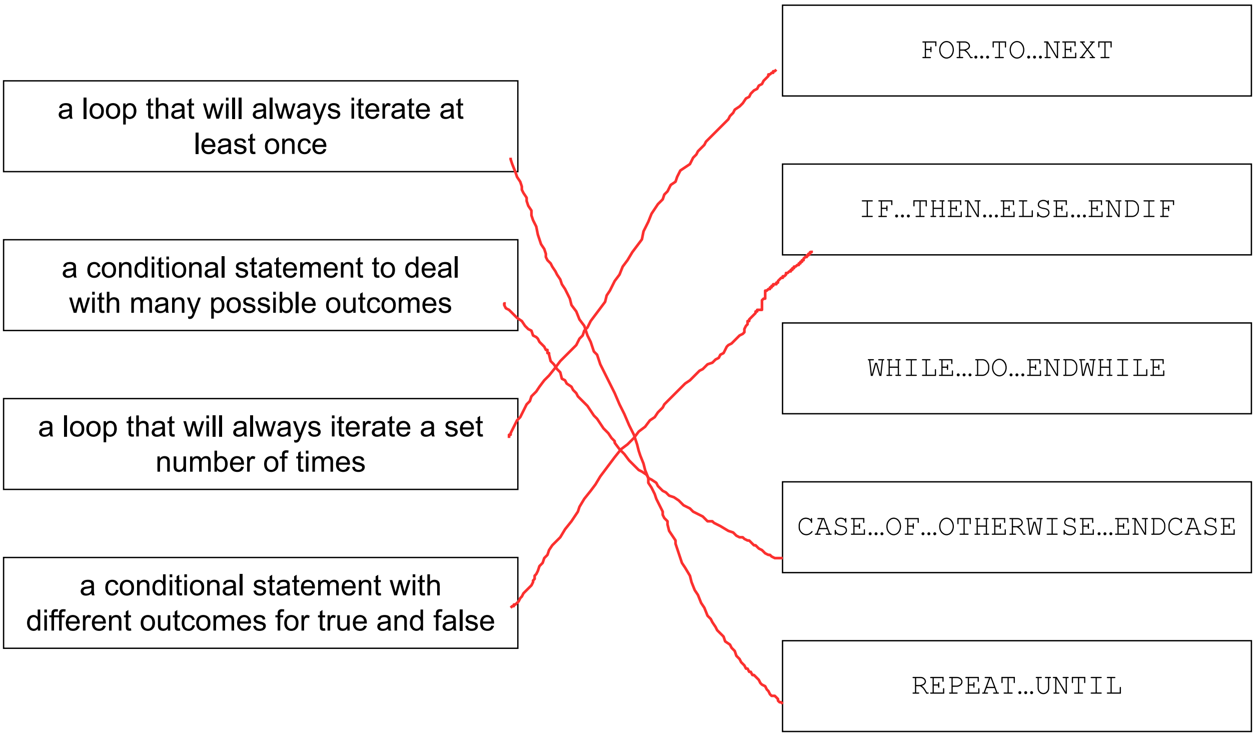
- The total mark for this paper is 75.
- The number of marks for each question or part question is shown in brackets [ ].
- No marks will be awarded for using brand names of software packages or hardware.

This document has **16** pages. Any blank pages are indicated.

1 Four pseudocode descriptions and five pseudocode statements are shown.

(a) Draw a line to link each pseudocode description to the most appropriate pseudocode statement.

Some pseudocode statements will **not** be used.



Pseudocode description
a loop that will always iterate at least once
a conditional statement to deal with many possible outcomes
a loop that will always iterate a set number of times
a conditional statement with different outcomes for true and false
Pseudocode statement
FOR…TO…NEXT
IF…THEN…ELSE…ENDIF
WHILE…DO…ENDWHILE
CASE…OF…OTHERWISE…ENDCASE
REPEAT…UNTIL

[4]

(b) Using a single loop, write an algorithm in pseudocode to output 50 names that have been stored in the array, `Name[]`

```
count <-- 0
WHILE count < 50 DO
    OUTPUT Name[Count]
    Count <-- Count + 1
ENDWHILE
```

[3]

2 Describe the purpose of validation and verification checks during data entry.

Include an example for each.

Validation check ....

Test if data entered is reasonable

EG Range check

....

Verification check ....

Test if data entered is as intended

EG Double entry

....

....

[4]

3 Tick (✓) **one** box to show the named section of a program that performs a specific task.

A file □ ✗

B function ☑ ✓

C parameter □ ✗

D process □ ✗

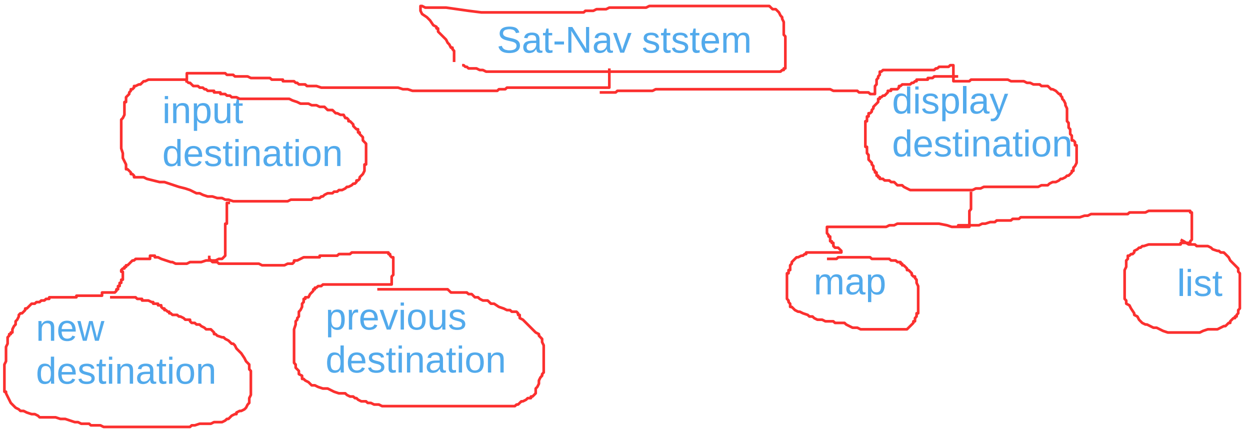
[1]

4 A satellite navigation system is an example of a computer system that is made up of sub-systems.

Part of a satellite navigation system:

- allows the user to enter details for a new destination or select a previously saved destination
- displays directions in the form of a visual map or as a list.

Draw a structure diagram for this part of the satellite navigation system.



[4]

5 An algorithm has been written in pseudocode to input some numbers. It only outputs any numbers that are greater than or equal to `100`. The number `999` is not output and stops the algorithm.

```
INPUT Number
WHILE Numbers <> 999 DO
   IF Number > 100
      THEN
         OUTPUT Number
   ENDIF
ENDWHILE
OUTPUT Number
```

(a) Identify the **four** errors in the pseudocode and suggest corrections.

Error 1 Numbers

Correction WHILE Number <> 999 DO

Error 2 not greater than or equal to 100

Correction IF Number >= 100

Error 3 number can only be input once as outside of loop

Correction put inside loop, after the endif statement

Error 4 we don't need the last output number

Correction remove it

[4]

(b) Write a pseudocode statement to change the corrected algorithm to output all numbers between `100` and `200` inclusive.

You do **not** need to rewrite the whole algorithm.

IF Number >= 100 AND Number <= 200

[2]

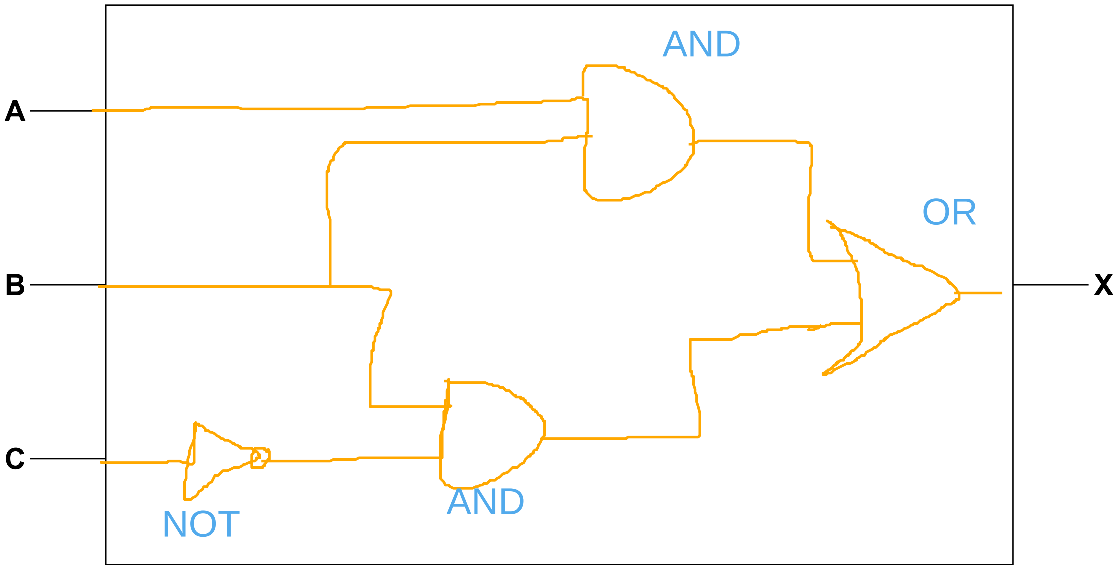
6 Consider this logic expression.

X = (**A** AND **B**) OR (**B** AND NOT **C**)

(a) Draw a logic circuit for this logic expression.

Each logic gate must have a maximum of **two** inputs.

Do **not** simplify this logic expression.



[4]

(b) Complete the truth table from the given logic expression.

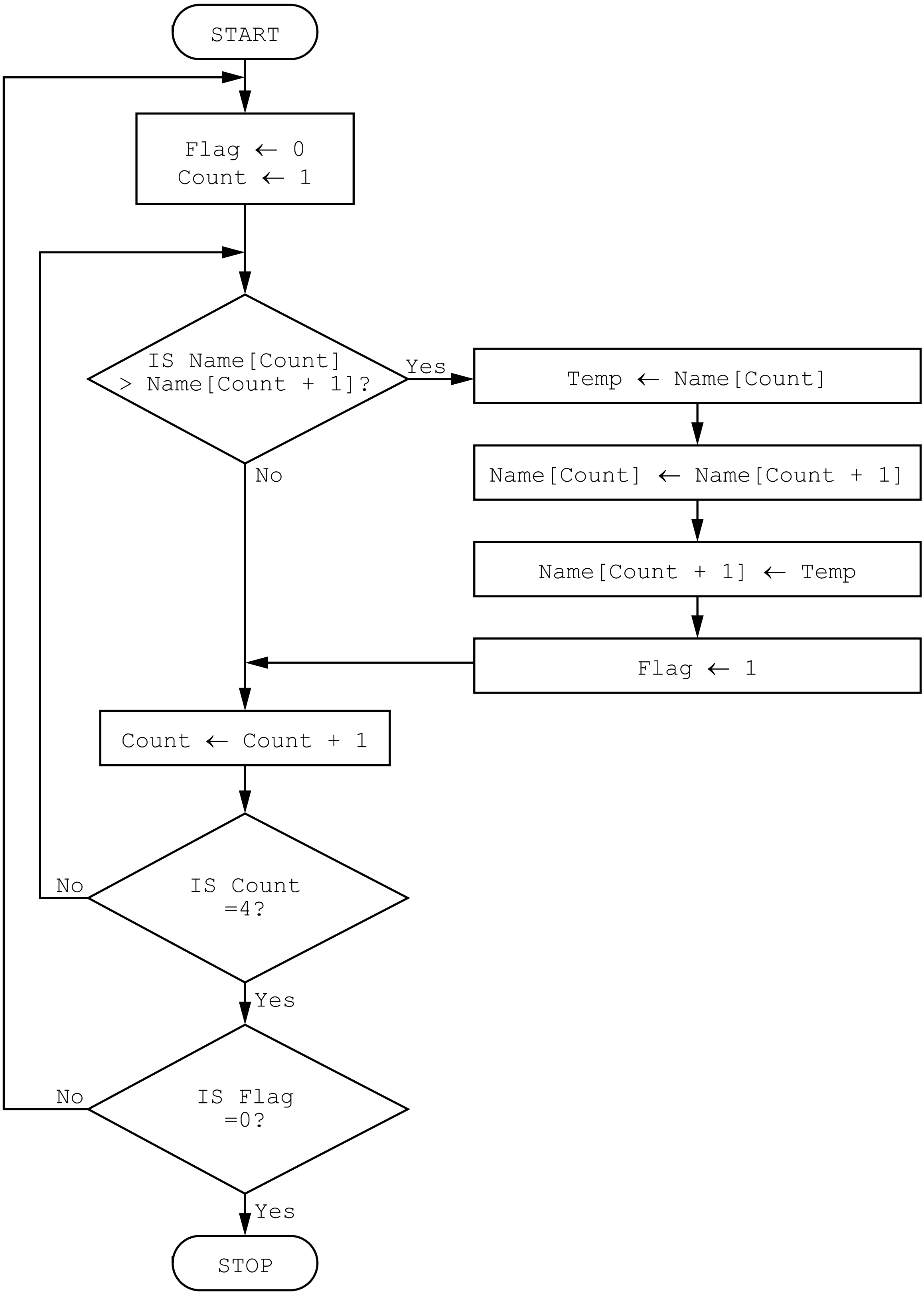
A	**B**	**C**	Working space			**X**
			NOT C	A AND B	B and NOT C	
0	0	0	1	0	0	0
0	0	1	0	0	0	0
0	1	0	1	0	1	1
0	1	1	0	0	0	0
1	0	0	1	0	0	0
1	0	1	0	0	0	0
1	1	0	1	1	1	1
1	1	1	0	1	0	1

[4]

BLANK PAGE

[Turn over

7 This flowchart represents an algorithm.



(a) The array `Name[1:4]` used in the flowchart contains the following data:

Name[1]	**Name[2]**	**Name[3]**	**Name[4]**
Jamal	Amir	Eve	Tara

Complete the trace table using the data given in the array.

Flag	**Count**	**Name[1]**	**Name[2]**	**Name[3]**	**Name[4]**	**Temp**
		Jamal	Amir	Eve	Tara	
0	1		Jamal			Jamal
1	2		Eve	Jamal		
	3					
	4					
0	1					
	2					
	3					
	4					

[5]

(b) Describe what the algorithm represented by the flowchart is doing.

Bubble Sort - soritng names into ascending order

[2]

8 A programmer has written an algorithm to check that prices are less than $10.00

These values are used as test data:

10.00 9.99 ten

State why each value was chosen as test data.

10.00 Boundary - should be rejected

9.99 Normal - should be accepted

ten Abnormal - data type is wrong

[3]

9 Explain why a program might need to store data in a file.

Can be transported, sent to another person

Can be saved when computer is switched off

Can be backed up

[3]

10 A function is declared using pseudocode.

```
FUNCTION ConvertToCm(Inches: REAL) RETURNS REAL
  RETURN Inches * 2.4
ENDFUNCTION
```

Tick (✓) **one** box which accurately describes the use of the variable `Inches`

A answer □

B call □

C parameter ☑

D response □

[1]

11 A database table, `2018MOV`, is used to keep a record of movie details.

`CatNo`	`Title`	`Genre1`	`Genre2`	`Blu-ray`	`DVD`	`Streaming`
18m01	Power Rangers	Adventure	Fantasy	Yes	No	Yes
18m02	Baywatch	Comedy	Drama	Yes	No	Yes
18m03	Table 19	Comedy	Drama	Yes	Yes	No
18m04	Wonder Woman	Action	Fantasy	Yes	No	Yes
18m05	Justice League	Action	Fantasy	Yes	Yes	Yes
18m06	Twilight	Thriller	Action	Yes	Yes	No
18m07	Ant Man	Action	Fantasy	No	Yes	No
18m08	Venice Beach	Action	History	No	Yes	No
18m12	Fast Five	Action	Thriller	No	Yes	No
18m15	King Kong	Adventure	Fantasy	No	Yes	No
18m16	Transformers: The Last Knight	Action	Sci-Fi	Yes	Yes	Yes
18m17	The Dark Tower	Fantasy	Sci-Fi	Yes	Yes	No
18m19	Beauty and the Beast	Fantasy	Romance	Yes	Yes	Yes
18m21	The Mummy	Action	Fantasy	No	No	Yes
18m22	Star Wars: Episode VIII	Sci-Fi	Action	Yes	No	Yes
18m23	Guardians of the Galaxy	Action	Sci-Fi	Yes	Yes	Yes
18m26	Thor	Action	Sci-Fi	No	Yes	Yes
18m27	Twilight	Fantasy	Sci-Fi	No	No	Yes
18m30	Beneath	Action	Fantasy	Yes	No	No
18m31	Despicable Me	Animation	Action	Yes	Yes	No

(a) State the number of records in the database table.

20 .... [1]

(b) **(i)** Give the name of the field that would be used for the primary key.

CatNo .... [1]

(ii) State the reason for choosing this field for the primary key.

it is a unique identifier ....

.... [1]

(c) Complete the table to identify the most appropriate data type for each field based on the data shown in the database table, `2018MOV`.

Field	**Data type**
`CatNo`	Text
`Title`	Text
`Genre1`	Text
`Streaming`	Boolean

[2]

(d) Complete the structured query language (SQL) to return the category number and title for all Comedy movies.

```
SELECT CatNo, Title
FROM .............................................. 2018MOV
WHERE Genre1 = ......"Comedy"...........................;
```

[2]

12 The variables X, Y and Z are used to store data in a program:

- X stores a string
- Y stores a position in the string (e.g. 2)
- Z stores the number of characters in the string.

(a) Write pseudocode statements to declare the variables X, Y and Z.

....

DECLARE X : STRING

....

DECLARE Y : INTEGER

....

DECLARE Z : INTEGER

....

....

.... [3]

(b) The function LENGTH(X) finds the length of a string X.

The function SUBSTRING(X,Y,Z) finds a substring of X starting at position Y and Z characters long. The first character in X is in position 1.

Write pseudocode statements to:
- store the string "Programming is fun" in X
- find the length of the string and output it
- extract the word fun from the string and output it.

....

X <-- "Programming is fun"

....

OUTPUT LENGTH(X)

....

Y <- 16

....

Z <- 3

....

OUTPUT SUBSTRING(X,Y,Z)

....

....

....

....

....

....

.... [6]

[Turn over

13 The one-dimensional (1D) array `StudentName[]` contains the names of students in a class. The two-dimensional (2D) array `StudentMark[]` contains the mark for each subject, for each student. The position of each student's data in the two arrays is the same, for example, the student in position 10 in `StudentName[]` and `StudentMark[]` is the same.

The variable `ClassSize` contains the number of students in the class. The variable `SubjectNo` contains the number of subjects studied. All students study the same number of subjects.

The arrays and variables have already been set up and the data stored.

Students are awarded a grade based on their average mark.

Average mark	**Grade awarded**
greater than or equal to 70	distinction
greater than or equal to 55 and less than 70	merit
greater than or equal to 40 and less than 55	pass
less than 40	fail

Write a program that meets the following requirements:

- 1 calculates the combined total mark for each student for all their subjects
- 2 calculates the average mark for each student for all their subjects, rounded to the nearest whole number
- 3 outputs for each student:
  - name
  - combined total mark
  - average mark
  - grade awarded
- 4 calculates, stores and outputs the number of distinctions, merits, passes and fails for the whole class.

You must use pseudocode or program code **and** add comments to explain how your code works.

You do **not** need to initialise the data in the array.

```
DECLARE TotalMark, AverageMark : ARRAY[1:50] OF INTEGER
DECLARE SubjectCounter,StudentCounter,DistinctionNo,MeritNo,PassNo,FailNo : INTEGER
CONTANT Distinction = 70
CONSTANT Merit = 55
CONSTANT Pass = 40
DestinctionNo <-- 0
MeritNo <-- 0
PassNo <-- 0
FailNo <-- 0
```

```
FOR StudentCounter <-- 1 to ClassSize
     TotalMark[StudentCounter] <-- 0
NEXT StudentCounter
 FOR StudentCounter <- 1 to ClassSize
      FOR SubjectCounter <- 1 to SubjectNo
         TotalMark[StudentCounter] <-- TotalMark[StudentCounter] + StudentMark [StudentCounter,SubjectCounter]   1
      NEXT SubjectCounter
       AverageMark[StudentCounter] <-- INT((Total[StudentCounter] / SubjectNo) + 0.5)   2
       OUTPUT "Name ", StudentName[StudentCounter]
       OUTPUT "Combined total mark ", TotalMark[StudentCounter]
       OUTPUT "Average mark ", AverageMark[StudentCounter]
       IF AverageMark[StudentCounter] >= Distinction THEN
           DistinctionNo <-- DistinctionNo +1
           OUTPUT " Grade Distinction"   3
       ELSE
           IF AverageMark[StudentCounter] >= Merit THEN
               MeritNo <-- MeritNo + 1
               OUTPUT "Grade Merit"
           ELSE
               IF AverageMark[StudentCounter] >= Pass THEN
                   PassNo <-- PassNo + 1
                   OUTPUT "Grade Pass"
               ELSE
                   FailNo <-- FailNo + 1
                   OUTPUT "Grade Fail"
               ENDIF
           ENDIF
       ENDIF
     Next StudentCounter
```

```
OUTPUT "Number of Distinctions",DistinctionNo
OUTPUT "Number of Merits ",MeritNo
OUTPUT "Number of Passes ", PassNo
OUTPUT "Number of Fails ", FailNo
```

4

[15]