Cambridge International AS & A Level

Cambridge International Examinations
Cambridge International Advanced Subsidiary and Advanced Level

CANDIDATE NAME	
CENTRE NUMBER	
CANDIDATE NUMBER	

3040246882

COMPUTING **9691/23**

Paper 2 **May/June 2016**

2 hours

Candidates answer on the Question Paper.

No additional materials are required.

READ THESE INSTRUCTIONS FIRST

Write your Centre number, candidate number and name on all the work you hand in.
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.

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.

This document consists of **16** printed pages.

© UCLES 2016

CAMBRIDGE International Examinations

1 Zara wants to write a temperature conversion program. The program is to output a table to show Fahrenheit temperatures and their Celsius equivalents.

She writes the program specification:

- input lower bound and upper bound of Fahrenheit temperatures (whole numbers only)
- input temperature interval (whole number only)
- output conversion table (Celsius temperatures rounded to the nearest whole number)

Example conversion table with lower bound 50, upper bound 90 and interval 5:

```
Conversion Table

Fahrenheit          Celsius
    50                10
    55                13
    :                 :
    :                 :
    90                32
```

The steps to convert a Fahrenheit temperature to a Celsius temperature are:

- subtract 32 from the Fahrenheit temperature
- multiply by 5
- divide by 9

(i) Zara writes pseudocode that uses the variables in the identifier table below.

Complete the identifier table.

Identifier	Data type	Explanation
`LowerTemp`		Lower bound value of Fahrenheit temperatures
`UpperTemp`		Upper bound value of Fahrenheit temperatures
`Interval`		The interval between two successive Fahrenheit temperatures in the table
`Fahrenheit`		The Fahrenheit temperature value to be converted
`Result`		Value of conversion to Celsius before rounding
`Celsius`		The rounded `Result`

[2]

© UCLES 2016

(ii) The built-in function `ROUND(x)` returns `x` rounded to the nearest whole number.

Complete the pseudocode to print the conversion table:

```
INPUT LowerTemp

INPUT UpperTemp

INPUT Interval

OUTPUT "     Conversion Table"

OUTPUT "Fahrenheit         Celsius"

Fahrenheit ← .....

REPEAT

   Result ← .....

   ..... ← ROUND(Result)

   OUTPUT Fahrenheit, "               ", Celsius

   Fahrenheit ← .....

UNTIL Fahrenheit > .....
```

[6]

2 A vending machine dispenses four different types of snack: 'A', 'B', 'C' or 'D'.
The vending machine accepts the following coins: 100 cents, 50 cents, 20 cents, 10 cents.
If the user inserts coins to a greater value than the price of the chosen snack, the machine will give change.

The program controlling the machine is to take as input:

- the type of snack chosen ('A', 'B', 'C' or 'D')
- the value of each coin inserted

The program output is:

- the message "No change", if the money inserted was exactly the price of the snack

  or

- a message summarising the number of each coin value needed to give the correct change

Question 2 continues on page 4.

(a) Write **program code** to implement the flowchart opposite as the procedure `VendingMachine`.

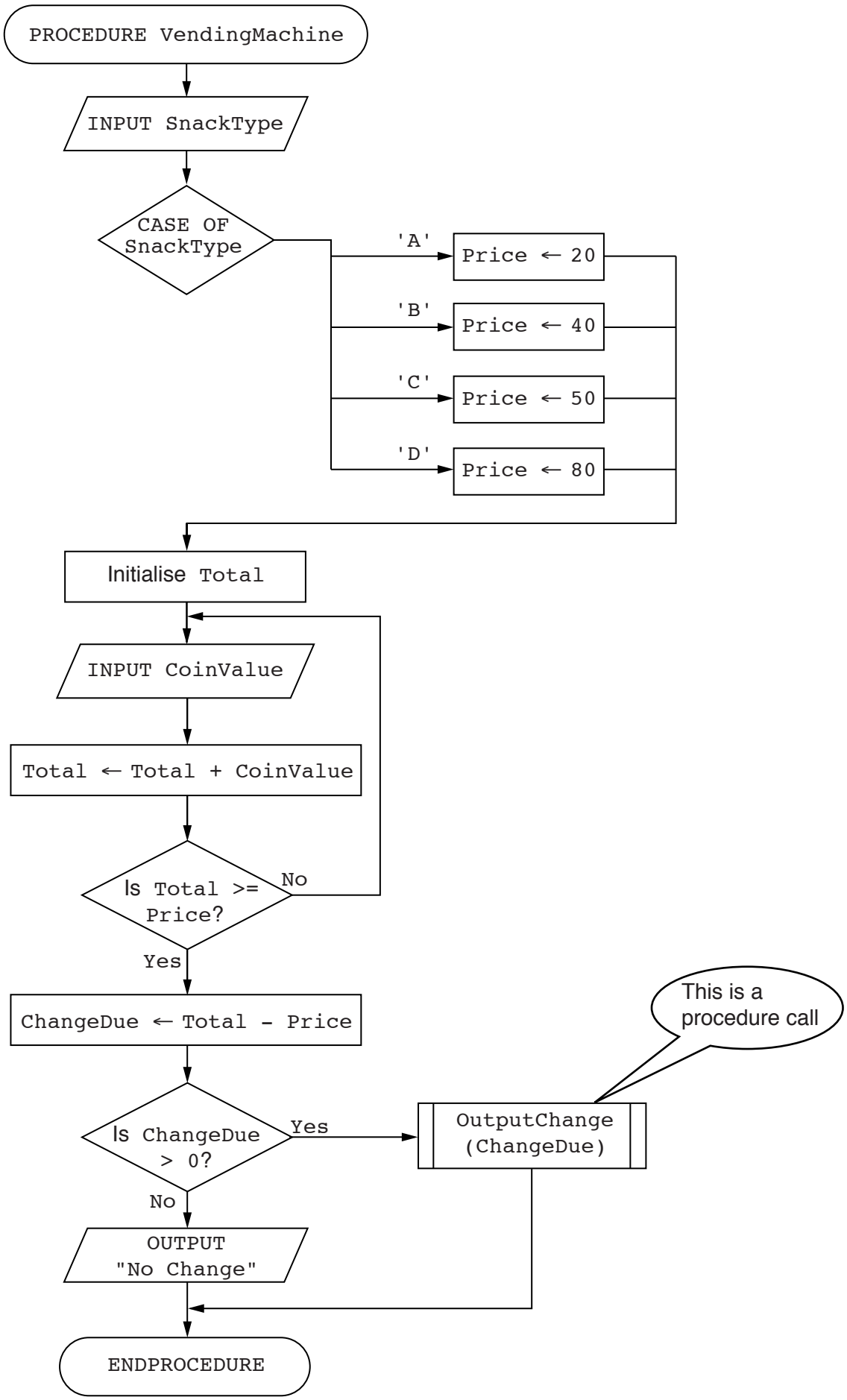
Programming Language .....

Code .....

.....

.....

.


PROCEDURE VendingMachine
INPUT SnackType
CASE OF SnackType
'A'
Price ← 20
'B'
Price ← 40
'C'
Price ← 50
'D'
Price ← 80
Initialise Total
INPUT CoinValue
Total ← Total + CoinValue
Is Total >= Price?
No
Yes
ChangeDue ← Total - Price
Is ChangeDue > 0?
Yes
OutputChange (ChangeDue)
This is a procedure call
No
OUTPUT "No Change"
ENDPROCEDURE

(b) If any change is to be given, it will be a combination of some or all of the coins: 50 cents, 20 cents, 10 cents.

The program will calculate the coin combination that gives as few coins as possible.

This requires calculation of:

- first, the number of 50-cent coins needed
- then, if more change is due, the number of 20-cent coins needed
- finally, if more change is due, the number of 10-cent coins needed

(i) A snack for 20 cents was bought using a 100-cent coin. Therefore `ChangeDue` has the value 100 – 20 = 80.

Complete the following table:

Statement	Value	Explanation
`X ← ChangeDue DIV 50`	`X` = ...	`X` represents ....
`Y ← ChangeDue MOD 50`	`Y` = ...	`Y` represents ....

[4]

(ii) The flowchart in **part (a)** shows the procedure call `OutputChange(ChangeDue)`. This procedure outputs a message summarising the number of each coin value needed to give the correct change.

Write **program code** for the procedure `OutputChange`.

Programming Language .....

Code .....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....[7]

Question 3 begins on Page 9.

3 A social club organises a competition for ten members. The competition consists of eight games. All ten competitors take part in all games.

At the end of each game, each competitor is awarded a score.

For example, after the first game, the following scores were recorded:

Game 1	
Competitor	**Score**
Rowan	23
Hema	14
Kai	10
Ellis	76
Rory	43
Arden	23
Quinn	30
Charles	16
Avery	34
River	31

The average score for each game is calculated.

For example, the average score for Game 1 was 30.

Each individual competitor's score is compared to the average score for that game.

Each competitor is awarded a points value as follows:

- –1 for a score below the average
- +1 for a score above the average
- 0 for a score equal to the average

Therefore, in the given example, competitors receive the following points:

Game 1	
Competitor	**Points**
Rowan	–1
Hema	–1
Kai	–1
Ellis	+1
Rory	+1
Arden	–1
Quinn	0
Charles	–1
Avery	+1
River	+1

© UCLES 2016

A points total is kept for each competitor throughout the competition. At the beginning of the competition each competitor's points total is 0. After each game, each member's points total is updated.

After all 8 games, the highest possible points total is +8 (better than average in all games); the lowest possible points total is –8 (worse than average in all games).

The competitor with the highest points total at the end of the competition is the winner.

The social club asks Alessio to write a program to:

- input each competitor's score after each game
- update the competitors' points totals

Alessio writes top-level pseudocode:

```
PROCEDURE GameResultUpdate
   CALL InputScoresAndCalcAverage(Average)
   CALL UpdatePointsTotals(Average)
ENDPROCEDURE
```

The procedure `Initialisation` sets up arrays `CompetitorName, Score` and `PointsTotal` as follows:

	CompetitorName
1	"Rowan"
2	"Hema"
3	"Kai"
4	"Ellis"
5	"Rory"
6	"Arden"
7	"Quinn"
8	"Charles"
9	"Avery"
10	"River"

	Score
1	0
2	0
3	0
4	0
5	0
6	0
7	0
8	0
9	0
10	0

	PointsTotal
1	0
2	0
3	0
4	0
5	0
6	0
7	0
8	0
9	0
10	0

Alessio now designs the procedures `InputScoresAndCalcAverage` and `UpdatePointsTotals`.

(a) Complete the pseudocode.

(i)
```
PROCEDURE InputScoresAndCalcAverage(BYREF Average : ..... )
   DECLARE ....., ..... : INTEGER
   TotalScore ← .....
   FOR x ← .....
      OUTPUT "Score for ", CompetitorName[x]
      // input scores into Score array
      INPUT .....
      TotalScore ← .....
   ENDFOR
   Average ← .....
ENDPROCEDURE
```
[7]

(ii)
```
PROCEDURE UpdatePointsTotals( ..... )
   DECLARE ..... : INTEGER
   FOR .....
      IF Score[i] > Average
         THEN   // increase PointsTotal
            .....
         ELSE   // below average?
            IF .....
               THEN
                  .....
            ENDIF
         .....
      .....
ENDPROCEDURE
```
[8]

(iii) Explain the use of `BYREF` in the procedure heading:

```
PROCEDURE InputScoresAndCalcAverage(BYREF Average : .....)
```

.....

.....

.....[2]

(b) The social club wants a printout of the competitor names and points totals after each game in rank order (highest points total first).

To do this, Alessio writes a pseudocode procedure:

```
PROCEDURE SortIntoRankOrder
   REPEAT
      NoMoreSwaps ← TRUE
      FOR x ← 1 TO 9
         IF PointsTotal[x] < PointsTotal[x + 1]
            THEN
               NoMoreSwaps ← FALSE
               Temp ← PointsTotal[x]
               PointsTotal[x] ← PointsTotal[x + 1]
               PointsTotal[x + 1] ← Temp
         ENDIF
      ENDFOR
   UNTIL NoMoreSwaps = TRUE
ENDPROCEDURE
```

© UCLES 2016

(i) Dry run the procedure call `SortIntoRankOrder`. Complete the trace table.

x	NoMore Swaps	PointsTotal[x] < PointsTotal[x+1]	Temp	PointsTotal									
				[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]	[10]
				+5	+3	+4	+2	0	+1	-2	-1	-1	-3
	TRUE												
1													
2													
3													
4													
5													
6													
7													
8													
9													

[9]

(ii) Alessio writes a procedure `DisplayRankOrderList`. This procedure is to display the competitors' names and their points totals in rank order.

```
PROCEDURE DisplayRankOrderList
   OUTPUT "          Competitor Table"
   OUTPUT "Rank           Name          Points Total"
   OUTPUT "========================================="
   FOR i ← 1 TO 10
      OUTPUT i, CompetitorName[i], PointsTotal[i]
   ENDFOR
ENDPROCEDURE
```

When the output is published, some competitors complain that their points total and rank are not correct.

Explain the error that Alessio has made.

.....

.....

.....[2]

(c) The social club wants to store the following additional data for each competitor:

- their scores in all 8 games
- date of birth
- telephone number

Alessio decides that a record structure would be more suitable for his program.

(i) Write **program code** to define a competitor record with identifier `CompetitorRecord` and the fields `CompetitorName`, `CompetitorTelNumber`, `DateOfBirth`, `GameScores` and `PointsTotal`.

Programming language .....

Code .....

.....

.....

.....

.....

.....

.....

.....

.....

.....[5]

(ii) There is one record for each competitor. Alessio decides to use an array of records to store the 10 competitor records.

Write **program code** to declare the array `CompetitorData`.

Programming language .....

Code .....

.....

.....[3]

(d) The social club want a facility to search for a competitor to find their current rank. This is done by searching the array `CompetitorData` for the competitor's name.

Complete the pseudocode for function `FindCompetitorRank`.

```
FUNCTION FindCompetitorRank (SearchName : STRING) RETURNS .....
   DECLARE .....
   .....
   REPEAT .....
      .....
      .....
   UNTIL .....
   .....
   RETURN .....
ENDFUNCTION
```

[7]

(e) At the end of the day, the computer is shut down. Alessio wants to save the data from the array `CompetitorData` to a new sequential file `CompetitorFile.DAT`

Write **program code** for a procedure `SaveToFile`.

Programming language .....

Code .....

.....

.....

.....

.....

.....

.....

.....

.....

.....[5]

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 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.