Cambridge International AS & A Level

Cambridge International Examinations
Cambridge International Advanced Subsidiary and Advanced Level

CANDIDATE NAME	

CENTRE NUMBER						CANDIDATE NUMBER					

* 4 5 3 1 9 9 0 1 8 1 *

COMPUTER SCIENCE **9608/23**

Paper 2 Fundamental Problem-solving and Programming Skills **October/November 2015**

2 hours

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.
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 75.

This document consists of **19** printed pages and **1** blank page.

DC (GB) 116769

CAMBRIDGE International Examinations

[Turn over

Throughout the paper you will be asked to write either **pseudocode** or **program code**.

Complete the statement to indicate which high-level programming language you will use.

Programming language .....

1 Computer programs have to evaluate expressions.

Study the sequence of pseudocode statements.

Write down the value assigned to each variable.

```
DECLARE h, w, r, Perimeter, Area : REAL
DECLARE A, B, C, D, E            : BOOLEAN
```

```
h ← 13.6
w ← 6.4
Perimeter ← (h + w) * 2
```

(i) Perimeter ..... [1]

```
r ← 10
Area 3.142 * r^2
```

(ii) Area ..... [1]

```
Z ← 11 + r / 5 + 3
```

(iii) Z ..... [1]

```
A ← NOT(r > 10)
```

(iv) A ..... [1]

2 A programmer uses an Integrated Development Environment (IDE) for all program development.

(i) Describe what is meant by an IDE.

.....

.....

.....

.....[2]

(ii) Name **three** features you would expect to be available in an IDE to help initial error detection or debugging.

1 .....

.....

2 .....

.....

3 .....

.....[3]

3 A program is to simulate the operation of a particular type of logic gate.

- The gate has two inputs (TRUE or FALSE) which are entered by the user.
- The program will display the output (TRUE or FALSE) from the gate.

The program uses the following identifiers in the pseudocode below:

Identifier	Data type	Description
`InA`	`BOOLEAN`	Input signal
`InB`	`BOOLEAN`	Input signal
`OutZ`	`BOOLEAN`	Output signal

```
INPUT InA
INPUT InB
IF (InA = FALSE AND InB = FALSE) OR (InA = FALSE AND InB = TRUE)
                                    OR (InA = TRUE AND InB = FALSE)
   THEN
      OutZ ← TRUE
   ELSE
      OutZ ← FALSE
ENDIF
OUTPUT OutZ
```

(a) The programmer chooses the following four test cases.

Show the output (`OutZ`) expected for each test case.

Test case	Input		Output
	`InA`	`InB`	`OutZ`
1	`TRUE`	`TRUE`	
2	`TRUE`	`FALSE`	
3	`FALSE`	`TRUE`	
4	`FALSE`	`FALSE`	

[4]

(b) The selection statement (lines `03` – `08`) could have been written with more simplified logic.

Rewrite this section of the algorithm in **pseudocode**.

.....

.....

.....

.....

.....

.....[3]

4 A program is to be written to calculate the discount given on purchases.

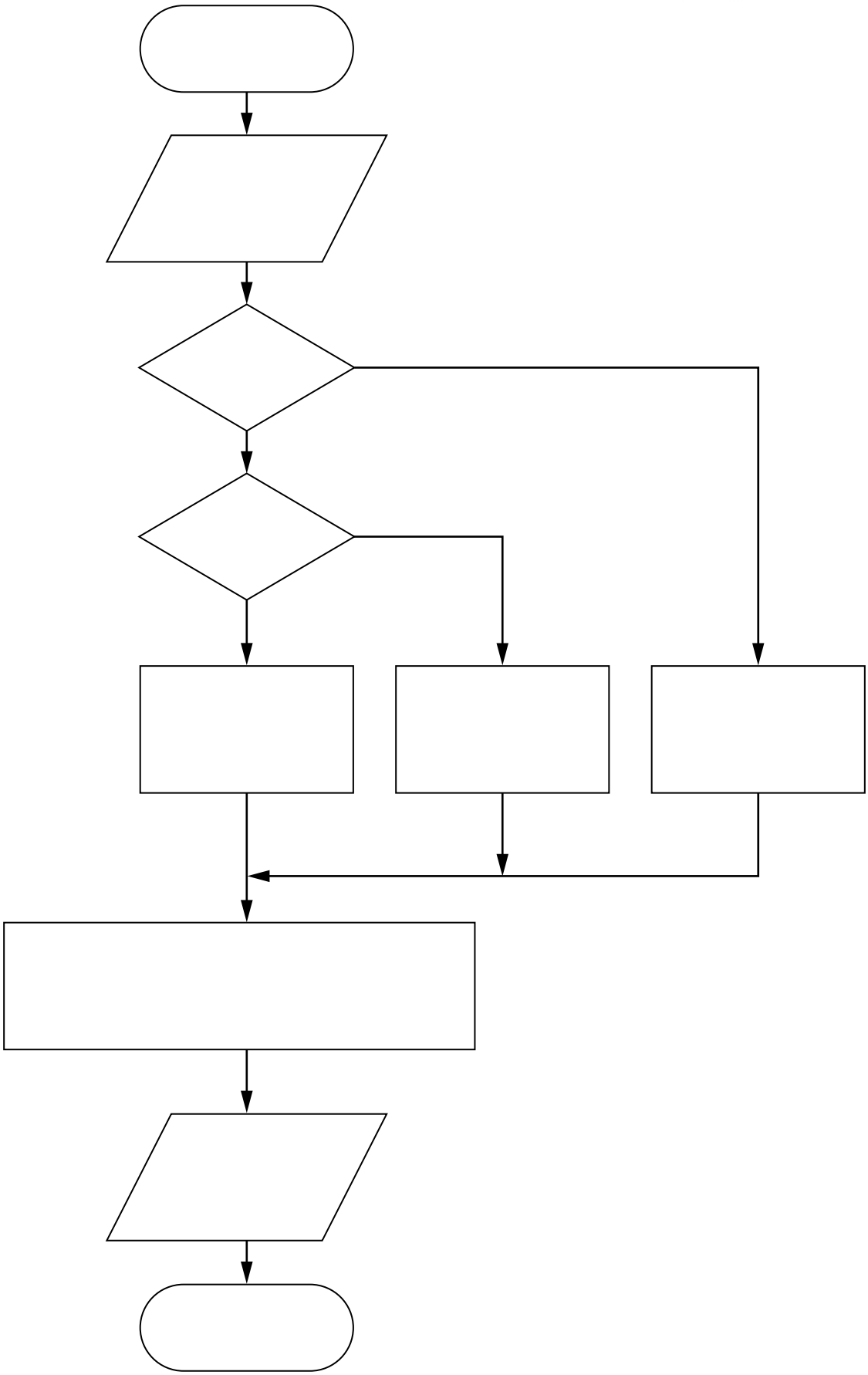
A purchase may qualify for a discount depending on the amount spent. The purchase price (`Purchase`), the discount rate (`DiscountRate`) and amount paid (`Paid`) is calculated as shown in the following pseudocode algorithm.

```
INPUT Purchase

IF Purchase > 1000
   THEN
      DiscountRate ← 0.10
   ELSE
      IF Purchase > 500
         THEN
            DiscountRate ← 0.05
         ELSE
            DiscountRate ← 0
      ENDIF
ENDIF

Paid ← Purchase * (1 - DiscountRate)
OUTPUT Paid
```

The algorithm is also to be documented with a program flowchart.

Complete the flowchart by:

- filling in the flowchart boxes
- labelling, where appropriate, lines of the flowchart



[6]

5 A driver buys a new car.

The value of the car reduces each year by a percentage of its current value.

The percentage reduction is:

- in the first year, 40%
- in each following year, 20%

The driver writes a program to predict the value of the car in future years.

The program requirements are:

- enter the cost of the new car (to nearest $)
- calculate and output the value of the car at the end of each year
- the program will end when either the car is nine years old, or when the value is less than $1000

(a) Study the incomplete pseudocode which follows in **part (b)** and fill in the identifier table.

Identifier	Data type	Description

[3]

(b) Complete the pseudocode for this design.

```
OUTPUT "Enter purchase price"
INPUT PurchasePrice

CurrentValue ← ..............................................................
YearCount ← 1

WHILE ..................................... AND ..............................

   IF ..........................................................................
      THEN
         CurrentValue ← CurrentValue * (1 - 40 / 100)
      ELSE
         CurrentValue ← ...........................................................
   ENDIF

   OUTPUT YearCount, CurrentValue

   ..............................................................................
ENDWHILE
```

[6]

6 A firm employs five staff who take part in a training programme. Each member of staff must complete a set of twelve tasks which can be taken in any order. When a member of staff successfully completes a task, this is recorded.

A program is to be produced to record the completion of tasks for the five members of staff.

To test the code, the programmer makes the program generate test data.

The program generates pairs of random numbers:

- the first, in the range, 1 to 5 to represent the member of staff
- the second, in the range, 1 to 12 to represent the task

Each pair of numbers simulates the completion of one task by one member of staff.

(a) Explain why the generation of 60 (5 staff x 12 tasks) pairs of random numbers will not simulate all tasks completed by all staff.

.....

.....

.....[2]

(b) Data is currently recorded manually as shown.

Staff number	Task number											
	1	2	3	4	5	6	7	8	9	10	11	12
1												
2												
3				✓								
4												
5								✓				

The table shows that two members of staff have each successfully completed one task.

The program must use a suitable data structure to store, for all staff:

- tasks successfully completed
- tasks not yet successfully completed

The program will output the staff number and task number in the order in which tasks are completed.

The program design in pseudocode is produced as follows:

```
DECLARE StaffNum          : INTEGER
DECLARE TaskNum           : INTEGER
DECLARE ..................................................................................
DECLARE NewStaffTask      : BOOLEAN

CALL InitialiseTaskGrid
Completed ← 0
WHILE Completed <> 60
   NewStaffTask ← FALSE
   WHILE NewStaffTask = FALSE
      StaffNum ← RANDOM(1,5)       //generates a random number
      TaskNum ← RANDOM(1,12)       //in the given range
      IF TaskGrid[StaffNum, TaskNum] = FALSE
         THEN
            TaskGrid[StaffNum, TaskNum] ← TRUE
            NewStaffTask ← TRUE
            OUTPUT StaffNum, TaskNum
      ENDIF
   ENDWHILE
   Completed ← Completed + 1
ENDWHILE
OUTPUT "Staff Task Count", Completed

// end of main program

PROCEDURE InitialiseTaskGrid()
   DECLARE i : INTEGER
   DECLARE j : INTEGER
   FOR i ← 1 TO 5
      FOR j ← 1 TO 12
         TaskGrid[i, j] ← FALSE
      ENDFOR
   ENDFOR
ENDPROCEDURE
```

Study the pseudocode and answer the questions below.

Give the line number for:

(i) The declaration of a `BOOLEAN` global variable. ..... [1]

(ii) The declaration of a local variable. ..... [1]

(iii) The incrementing of a variable used as a counter, but not to control a 'count controlled' loop.

..... [1]

(iv) A statement which uses a built-in function of the programming language.

..... [1]

(c) **(i)** State the number of parameters of the `InitialiseTaskGrid` procedure.

..... [1]

(ii) Copy the condition which is used to control a 'pre-condition' loop.

.....[1]

(iii) Explain the purpose of lines `13` – `18`.

.....

.....

.....

.....

.....

.....[3]

(iv) Give the global variable that needs to be declared at line `03`.

.....[2]

(d) Line `17` in the pseudocode outputs the staff number and the task number.

A new requirement is to display the name of the member of staff given in the table.

Write a CASE structure using variable `StaffNum`.

Assign to a new variable `StaffName` the appropriate staff name.

Staff number	Staff name
1	Sadiq
2	Smith
3	Ho
4	Azmah
5	Papadopoulos

.....

.....

.....

.....

.....

.....

.....

.....

.....[4]

Question 7 begins on page 14.

7 ASCII character codes are used to represent a single character.

Part of the code table is shown below.

ASCII code table (part)

Character	Decimal	Character	Decimal	Character	Decimal
<Space>	32	I	73	R	82
A	65	J	74	S	83
B	66	K	75	T	84
C	67	L	76	U	85
D	68	M	77	V	86
E	69	N	78	W	87
F	70	O	79	X	88
G	71	P	80	Y	89
H	72	Q	81	Z	90

Some pseudocode statements follow which use the built-in functions below:

```
ONECHAR(ThisString : STRING, Position : INTEGER) RETURNS CHAR
```
returns the single character at position `Position` (counting from the start of the string with value 1) from the string `ThisString`.
For example: `ONECHAR("Barcelona", 3)` returns 'r'.

```
CHARACTERCOUNT(ThisString : STRING) RETURNS INTEGER
```
returns the number of characters in the string `ThisString`.
For example: `CHARACTERCOUNT("BRAZIL")` returns 6.

```
CHR(ThisInteger : INTEGER) RETURNS CHAR
```
returns the character with ASCII code `ThisInteger`.
For example: `CHR(65)` returns character 'A'.

```
ASC(ThisCharacter : CHAR) RETURNS INTEGER
```
returns the ASCII value for character `ThisCharacter`.
For example: `ASC('A')` returns 65.

(a) Show the values stored by variables `A`, `B`, `C` and `D`.

The & operator is used to concatenate two strings.

```
Num1 ← 15
A ← CHR(67) & CHR(65) & CHR(84)
B ← ASC('P') - ASC('F') + 3
C ← ASC(ONECHAR("BISCUITS", 3))
D ← CHARACTERCOUNT("New York City") + 2
```

(i) `A` ..... [1]

(ii) `B` ..... [1]

(iii) `C` ..... [1]

(iv) `D` ..... [1]

(b) A program is to be written which accepts a string and then calculates a numeric value from this string. The input string and the calculated value are then to be sent to a remote computer over a communications link.

Study the following pseudocode:

```
OUTPUT "Enter string"
INPUT MyString
StringTotal ← 0

FOR i ← 1 TO CHARACTERCOUNT(MyString)
   NextNum ← ASC(ONECHAR(MyString, i))
   StringTotal ← StringTotal + NextNum
ENDFOR

OUTPUT MyString, StringTotal
```

Write the above pseudocode algorithm as **program code**.

There is no need to show the declaration of variables or comment statements.

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....[6]

(c) Explain the purpose of sending the value of `StringTotal` to the remote computer, in addition to `MyString`.

.....

.....

.....

.....[2]

8 In this question you will need to use the given pseudocode built-in function:

```
ONECHAR(ThisString : STRING, Position : INTEGER) RETURNS CHAR
```
returns the single character at position `Position` (counting from the start of the string with value 1) from the string `ThisString`.
For example: `ONECHAR("Barcelona", 3)` returns 'r'.

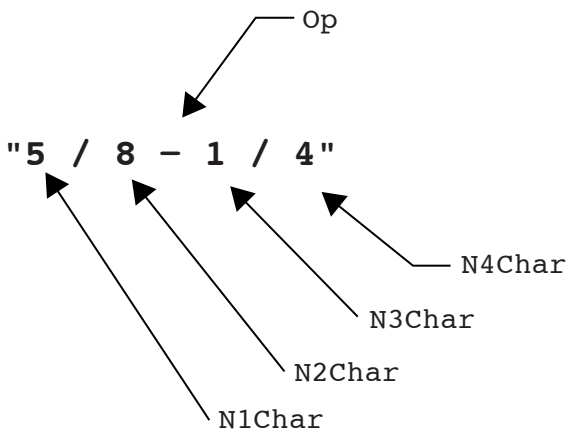
(a) Give the value assigned to variable `y` by the following statement:

`y ← ONECHAR("San Francisco", 6)` `y` ..... [1]

A program reads a string entered by the user. The string represents the addition or subtraction of two fractions. Each part of the fraction within the string is always a single digit only and the top digit is always less than the bottom digit.

Example strings are: `"3/8+3/5"` and `"5/8-1/4"`

The program steps are:
- the user enters the string
- the program isolates each digit and the operator
- the program computes the answer as either:
  - a fraction
  - a whole number followed by a fraction
  - a whole number
- the program displays the answer to the user



The identifier table shows the variables to be used to store the characters in the string as shown in the diagram.

Identifier	Data type	Description
`FractionString`	`STRING`	String input by user. For example: `"5/8-1/4"`
`N1Char`	`CHAR`	See diagram
`N2Char`	`CHAR`	See diagram
`N3Char`	`CHAR`	See diagram
`N4Char`	`CHAR`	See diagram
`Op`	`CHAR`	See diagram

(b) Study the sequence of pseudocode statements.

Show the values assigned to each variable.

```
FractionString ← "3/7+2/9"
N3Char ← ONECHAR(FractionString, 5)
Op ← ONECHAR(FractionString, 4)
```

(i) `N3Char` ..... [1]

(ii) `Op` ..... [1]

(iii) Complete the function call to isolate the character `'9'` from `FractionString`.

```
FractionString ← "3/7+2/9"
ONECHAR(FractionString, ..............)
```

[1]

The following additional variables are to be used by the program:

Identifier	Data type	Description
N1	INTEGER	The number value of N1Char
N2	INTEGER	The number value of N2Char
N3	INTEGER	The number value of N3Char
N4	INTEGER	The number value of N4Char
TopAnswer	INTEGER	The numerator of the fraction answer
BottomAnswer	INTEGER	The denominator of the fraction answer

(c) The following pseudocode uses these additional built-in functions:

`TONUM(ThisDigit : CHAR) RETURNS INTEGER` returns the integer value of character `ThisDigit` For example: `TONUM('8')` returns digit `8`.

`TOSTR(ThisNumber : INTEGER) RETURNS STRING` returns the string value of integer `ThisNumber` For example: `TOSTR(27)` returns `"27"`.

Study the pseudocode.

Complete the **three** dry runs for the three given values of `FractionString`.

```
OUTPUT "Enter the expression"
INPUT FractionString

// isolate each number digit and assign its number value
N1Char ← ONECHAR(FractionString, 1)
N1 ← TONUM(N1Char)
N2Char ← ONECHAR(FractionString, 3)
N2 ← TONUM(N2Char)
N3Char ← ONECHAR(FractionString, 5)
N3 ← TONUM(N3Char)
N4Char ← ONECHAR(FractionString, 7)
N4 ← TONUM(N4Char)

BottomAnswer ← N2 * N4

Op ← ONECHAR(FractionString, 4)
IF Op = '+'
   THEN
      // add fractions
      TopAnswer ← (BottomAnswer/N2) * N1 + (BottomAnswer/N4) * N3
   ELSE
      // subtract fractions
      TopAnswer ← (BottomAnswer/N2) * N1 - (BottomAnswer/N4) * N3
ENDIF

IF TopAnswer = BottomAnswer
   THEN
      OUTPUT '1'
   ELSE
      IF TopAnswer > BottomAnswer
         THEN
            TopAnswer ← TopAnswer MOD BottomAnswer
            // the & operator joins strings or character values
            OUTPUT "1 " & TOSTR(TopAnswer) & "/" & TOSTR(BottomAnswer)
         ELSE
            OUTPUT TOSTR(TopAnswer) & "/" & TOSTR(BottomAnswer)
      ENDIF
ENDIF
```

(i) `FractionString ← "2/5-3/8"`

N1	N2	N3	N4	BottomAnswer	Op	TopAnswer	OUTPUT

[2]

(ii) `FractionString ← "3/4+1/4"`

N1	N2	N3	N4	BottomAnswer	Op	TopAnswer	OUTPUT

[2]

(iii) `FractionString ← "7/9+2/3"`

N1	N2	N3	N4	BottomAnswer	Op	TopAnswer	OUTPUT

[3]

(d) The programmer writes code from the given pseudocode design. The program works, but the design is limited.

The programmer is to make amendments to the design following suggested specification changes.

(i) State the name for this type of maintenance.

.....[1]

(ii) Describe **three** specification changes which will make the program more useful.

1 .....

.....

2 .....

.....

3 .....

.....[3]

BLANK PAGE