Cambridge International Examinations
Cambridge International Advanced Subsidiary and Advanced Level

CANDIDATE NAME									
CENTRE NUMBER						CANDIDATE NUMBER			

COMPUTER SCIENCE

AS Level - Paper 2

Revision Set 2

Topics covered:

programming constructs and equivalent pseudocode:

4 programming constructs

assignment
sequence
selection
iteration

Count controlled loops (ie FOR LOOPS)

Pre-condition loops (ie WHILE LOOPS)

Post-condition loops (ie REPEAT...UNTIL LOOPS)

pseudocode

data structures

1D Arrays

data types

INTEGER, REAL, BOOLEAN, STRING, CHAR

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]

[Turn over

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

```
01  INPUT InA
02  INPUT InB
03  IF (InA = FALSE AND InB = FALSE) OR (InA = FALSE AND InB = TRUE)
                                        OR (InA = TRUE AND InB = FALSE)
04     THEN
05        OutZ ← TRUE
06     ELSE
07        OutZ ← FALSE
08  ENDIF
09  OUTPUT OutZ
```

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

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

Test case	Input		Output OutZ
	InA	InB	
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.

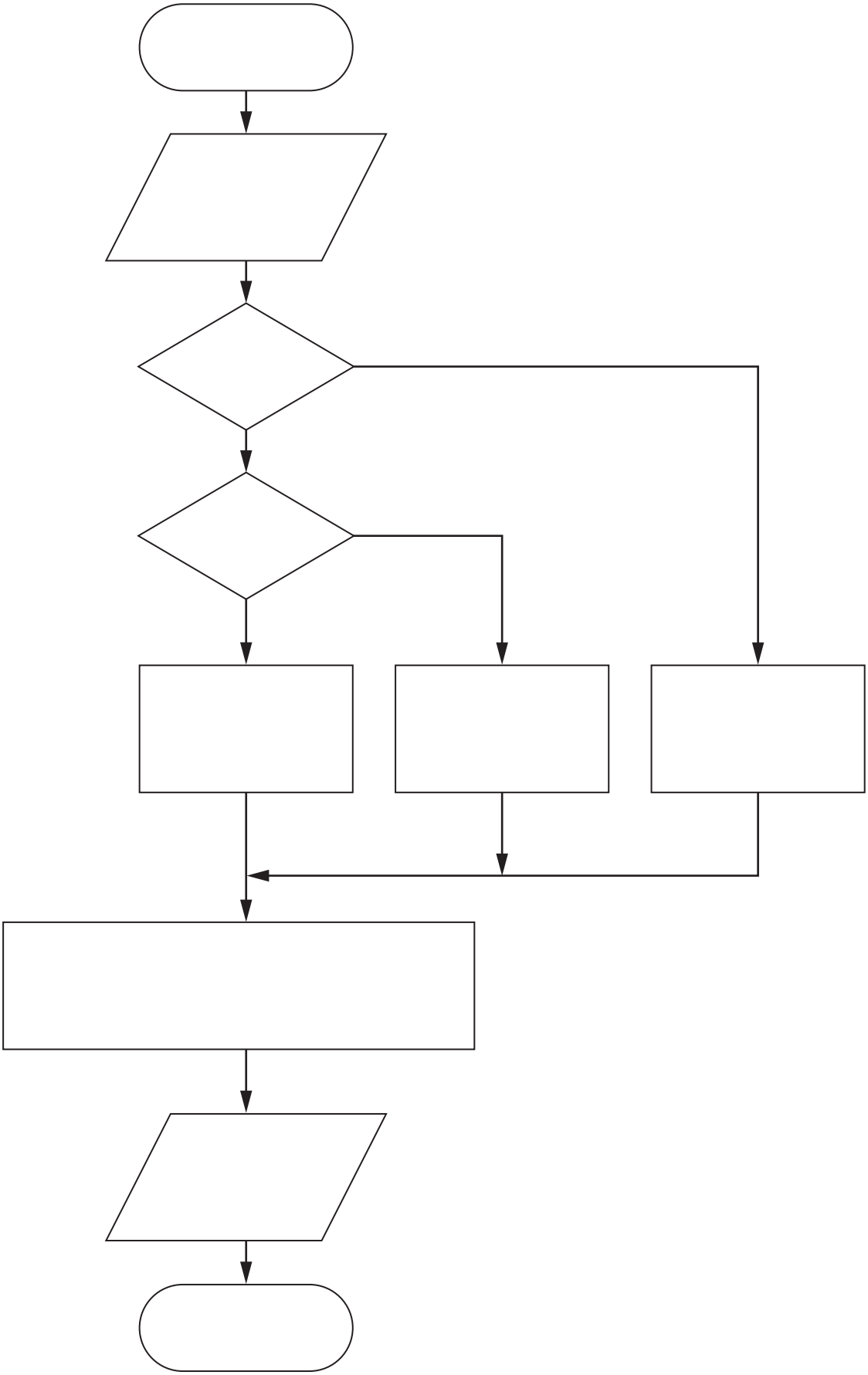
A purchase may qualify for a discount depending on the amount spent. The purchase price (`Purchase`), the discount rate (`DiscountRate`) and amount paid (`Paid`) is calculated as shown in the following pseudocode algorithm.

```
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:

```
01  DECLARE StaffNum          : INTEGER
02  DECLARE TaskNum           : INTEGER
03  DECLARE ……
04  DECLARE NewStaffTask      : BOOLEAN
05
06  CALL InitialiseTaskGrid
07  Completed ← 0
08  WHILE Completed <> 60
09    NewStaffTask ← FALSE
10    WHILE NewStaffTask = FALSE
11       StaffNum ← RANDOM(1,5)       //generates a random number
12       TaskNum ← RANDOM(1,12)       //in the given range
13       IF TaskGrid[StaffNum, TaskNum] = FALSE
14         THEN
15            TaskGrid[StaffNum, TaskNum] ← TRUE
16            NewStaffTask ← TRUE
17            OUTPUT StaffNum, TaskNum
18       ENDIF
19    ENDWHILE
20    Completed ← Completed + 1
21  ENDWHILE
22  OUTPUT "Staff Task Count", Completed
23
24  // end of main program
25
26  PROCEDURE InitialiseTaskGrid()
27    DECLARE i : INTEGER
28    DECLARE j : INTEGER
29    FOR i ← 1 TO 5
30       FOR j ← 1 TO 12
31          TaskGrid[i, j] ← FALSE
32       ENDFOR
33    ENDFOR
34  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.

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

Write a CASE structure using variable **StaffNum** .

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

.....

.....

.....

.....

.....

.....

.....

.....

.....

7 The engine management system of a car includes an energy-saving facility. When certain conditions are met, this facility will automatically stop the engine.

The system is to be software-based. It will include a subroutine, EnergySaver , which repeatedly takes data from sensors in the car. The subroutine decides whether or not to set the EngineStop value.

The table of identifiers used by this subroutine is shown below.

(a) Complete the identifier table below by stating the data types.

Identifier	Data type	Description
Accelerator	.....	Accelerator pedal position Values: 0 to 100 in steps of 1 Meaning: 0: none (not pressed) 100: maximum (fully pressed)
EngineTemp	.....	Engine temperature in °C (-50 to +150 stored to 1 decimal place)
NormalTemp	.....	Normal engine temperature in °C Whole number; typical value 90
Speed	.....	Road speed of car (in km/hr) Values: 0 to 200 in steps of 1
EngineStop	.....	Value used to signal engine must be stopped Possible values: TRUE: stop engine FALSE: run engine

[5]

The condition for stopping the engine is that all three of the following are true:

- Accelerator is not pressed
- Engine temperature is normal or above
- Car speed is zero

The initial design stage will produce a prototype of EnergySaver , with a user interface.

The structured English for this is:

1. INPUT value for accelerator pedal position
2. INPUT value for engine temperature
3. INPUT value for normal engine temperature
4. INPUT value for car speed
5. EVALUATE engine stopping condition
6. IF stopping condition satisfied SET engine stop value to TRUE
7. IF stopping condition not satisfied SET engine stop value to FALSE
8. OUTPUT message indicating engine stop value

(b) Write the pseudocode equivalent of the structured English. Use the identifiers from the table in part (a) .

For the built-in functions list, refer to the Appendix on the last page.

.....[6]

Appendix

Built-in functions (Pseudocode)

In each function below, if the function call is not properly formed, the function returns an error.

MID(ThisString : STRING, x : INTEGER, y : INTEGER) RETURNS STRING

returns the string of length y starting at position x from ThisString
Example: MID("ABCDEFGH", 2, 3) will return string "BCD"

LEFT(ThisString : STRING, x : INTEGER) RETURNS STRING

returns the leftmost x characters from ThisString
Example: LEFT("ABCDEFGH", 3) will return string "ABC"

RIGHT(ThisString: STRING, x : INTEGER) RETURNS STRING

returns the rightmost x characters from ThisString
Example: RIGHT("ABCDEFGH", 3) will return string "FGH"

CHR(x : INTEGER) RETURNS CHAR

returns the character whose ASCII value is x
Example: CHR(87) will return 'W'

ASC(x : CHAR) RETURNS INTEGER

returns the ASCII value of character x
Example: ASC('W') will return 87

LCASE(x : CHAR) RETURNS CHAR

returns the lower case equivalent character of x
Example: LCASE('W') will return 'w'

UCASE(x : CHAR) RETURNS CHAR

returns the upper case equivalent character of x
Example: UCASE('h') will return 'H'

INT(x : REAL) RETURNS INTEGER

returns the integer part of x
Example: INT(27.5415) will return 27