UNIVERSITY OF CAMBRIDGE INTERNATIONAL EXAMINATIONS
Cambridge International Diploma in Computing
Advanced Level

DIPLOMA IN COMPUTING **5217**

Module 2: Practical Tasks

May/June 2005

READ THESE INSTRUCTIONS FIRST

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

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 **6** printed pages and **2** blank pages.

IB05 06_5217_01/FP

UNIVERSITY *of* CAMBRIDGE
International Examinations

[Turn over

Answer **all** questions

Task 1 [22 marks]

This is a design and implementation task

At a university a lecturer may teach many modules and a module may be taught by many lecturers.

Lecturers have an office and a telephone number. Several lecturers, who may have to share a telephone, may use an office. An office is identified by a 5-character code consisting of two uppercase letters followed by three digits such as AB123. The telephone number consists of four digits such as 0362. The lecturer's name is the only personal data to be stored.

Modules have a module code consisting of two letters and four digits such as IT1002. Each module has a title such as Information Technology.

(a) Design a database, consisting of three tables, to store the data. You should include evidence showing the attributes of the tables together with their data types and any validation rules you have used. Screen dumps are acceptable for this purpose. [11]

Create sufficient data to complete the rest of this task. Provide copies of your completed tables. Screen dumps are acceptable for this purpose.

(b) **(i)** Create a form that will allow the user to add a lecturer to the appropriate table.

(ii) Create a form that will allow the user to add a module to the appropriate table.

(iii) Create a form that will allow a user to link a lecturer to a module. The form should allow the user to choose a lecturer from a list and to choose a module from a list. [6]

(c) Create a query that, when run, requests a user to select a lecturer and then lists all the modules taught by that lecturer. Include evidence that your query works correctly. [2]

(d) Create a report that lists the lecturers for each module. The report should group the information on module code and have appropriate headings. Include evidence that your report is correct. [3]

Task 2 [18 marks]

This is a white box test task. No implementation is required.

Read the following algorithm for a procedure, called Validate, in which each step has been numbered.

```
Validate(aString)
1  Length = Len(aString)
2     If Length = 0 Then
3        Output "Empty string is not allowed"
4     Else
5        OK = TRUE
6        DP = FALSE
7        Count = 1
8        Ch = 1st character of aString
9        IF Ch < "0" OR Ch > "9" THEN
10          OK = FALSE
11       ELSE
12          WHILE Count < Length AND OK DO
13             Count = Count + 1
14             Ch = next character in aString
15             IF Ch = "." THEN
16                IF DP THEN
17                   OK = FALSE
18                ELSE
19                   DP = TRUE
20                ENDIF
21             ELSE
22                IF Ch < "0" OR Ch > "9" THEN
23                   OK = FALSE
24                ENDIF
25             ENDIF
26          ENDWHILE
27       ENDIF
28       IF OK THEN
29          Output "Valid string"
30       ELSE
31          Output "Invalid string"
32       ENDIF
33    ENDIF
34 END PROCEDURE Validate
```

The function Len(aString) returns the number of characters in aString. For example, if aString = "Computer", Len(aString) = 8.

When the procedure is called with

    Validate("58")

the lines

    1, 2, 4, 5, 6, 7, 8, 9, 11, 12, 13, 14, 15, 21, 22, 24, 25, 26,12, 26, 27,
    28, 29, 30, 32, 33, 34

are executed and the output is

    Valid string.

The example test above can be described as shown in Table 2.1.

Path	**Test Data**	**Expected Output**
1, 2, 4, 5, 6, 7, 8, 9, 11, 12, 13, 14, 15, 21, 22, 24, 25, 26,12, 26, 27, 28, 29, 30, 32, 33, 34	58	Valid string.

Table 2.1

Using a table similar to Table 2.1, write down the lines that are executed and the output when the procedure is called with

(i) Validate(".376"); [4]

(ii) Validate("4.9"); [7]

(iii) Validate("6.2.7"). [7]

Task 3 [20 marks]

This is a design and implementation task

You are to design and create a very simple binary calculator using a high-level language of your choice. A binary calculator can only use the digits 0 and 1. This calculator will allow the user to enter two unsigned binary numbers and will then allow the user to add, subtract, multiply or divide these numbers. In the case of division, the result should be truncated to a whole number. This is equivalent to integer division. The result is to be displayed in binary.

To do this you are advised to follow these steps.

(a) Design an interface that will allow the user to input two unsigned binary numbers and will display them in two boxes. The user should then be able to choose addition, subtraction, multiplication or division. There should be a box to show the result of the calculation. The user should also be able to clear all the boxes. (No processing is required at this stage.) [4]

(b) Create a function, called BinaryToDecimal, that will accept a binary number as a parameter and will return its decimal equivalent. For example,

BinaryToDecimal(101101) will return the decimal value 45.

You may use the following algorithm to do this.

```
Input the binary number as a string
Total = 0
For each binary digit in the string, starting on the left,
    Total = Total * 2 + value of digit
Return Total
```

You should clearly annotate your code and use meaningful names for variables and other objects such as buttons and text boxes (if you use them). You should include a copy of your code as evidence. [4]

(c) Create a function, called DecimalToBinary, that will accept a decimal number as a parameter and will return its binary equivalent. For example,

DecimalToBinary (47) will return the binary value 101111.

You may use the following algorithm to do this.

```
Input the decimal number called Number
Binary = ""                    'Empty string
Repeat
    Remainder = remainder after Number is divided by 2
    Number = Whole part of Number divided by 2
    Binary = String value of Remainder & Binary
Until Number = 0
```

& means concatenation. For example

"1010" & "1" = "10101"

You should clearly annotate your code and use meaningful names for variables and other objects such as buttons. You should include a copy of your code as evidence. [4]

(d) Create code for each of the operations of addition, subtraction, multiplication and division. You are advised to change the binary numbers input by the user to decimal, do the operation and then convert the result back to binary before displaying it. You should include a copy of your code as evidence. [3]

(e) Create a set of test data that shows that the operations of addition, subtraction, multiplication and division work and provide evidence that you have used this data. Screen dumps are acceptable but must show the data entered and the result of the operation. [5]

BLANK PAGE

BLANK PAGE