Cambridge Assessment
International Education

Cambridge International AS & A Level

CANDIDATE NAME	

CENTRE NUMBER						CANDIDATE NUMBER					

COMPUTER SCIENCE **9608/22**

Paper 2 Fundamental Problem-solving and Programming Skills **May/June 2020**

2 hours

You must answer on the question paper.

No additional materials are needed.

INSTRUCTIONS

- Answer **all** questions.
- Use a black or dark blue pen.
- 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.
- You may use an HB pencil for any diagrams, graphs or rough working.
- 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 **20** pages. Blank pages are indicated.

BLANK PAGE

1 **(a)** Selection and repetition are basic constructs of an algorithm.

Name **and** describe **one other** construct.

Name ....

Description ....

....

.... [3]

(b) Program coding is a transferable skill.

Explain the term **transferable skill**.

....

....

....

.... [2]

(c) Count-controlled and post-condition are two types of loop.

Describe the characteristics of each of these types of loop.

Count-controlled ....

....

....

Post-condition ....

....

.... [2]

(d) Name **three** features provided by an Integrated Development Environment (IDE) that assist in the coding and **initial** error detection stages of the program development cycle.

1 ....

2 ....

3 .... [3]

2 **(a)** A structure chart is often produced as part of a modular program design. The chart shows the hierarchy of modules and the sequence of execution.

Give **two other** features the structure chart can show.

Feature 1 ....

....

Feature 2 ....

....
[2]

(b) Six program modules implement part of an online shopping program. The following table gives the modules and a brief description of each module:

Module	**Description**
`Shop()`	Allows the user to choose a delivery slot, select items to be added to the basket and finally check out
`ChooseSlot()`	Allows the user to select a delivery time. Returns a delivery slot number
`FillBasket()`	Allows the user to select items and add them to the basket
`Checkout()`	Completes the order by allowing the user to pay for the items. Returns a Boolean value to indicate whether or not payment was successful
`Search()`	Allows the user to search for a specific item. Returns an item reference
`Add()`	Adds an item to the basket. Takes an item reference and a quantity as parameters

(i) The online shopping program has been split into sub-tasks as part of the design process.

Explain the advantages of decomposing the program into modules. Your explanation should refer to the scenario and modules described in **part (b)**.

....

....

....

....

....

.... [3]

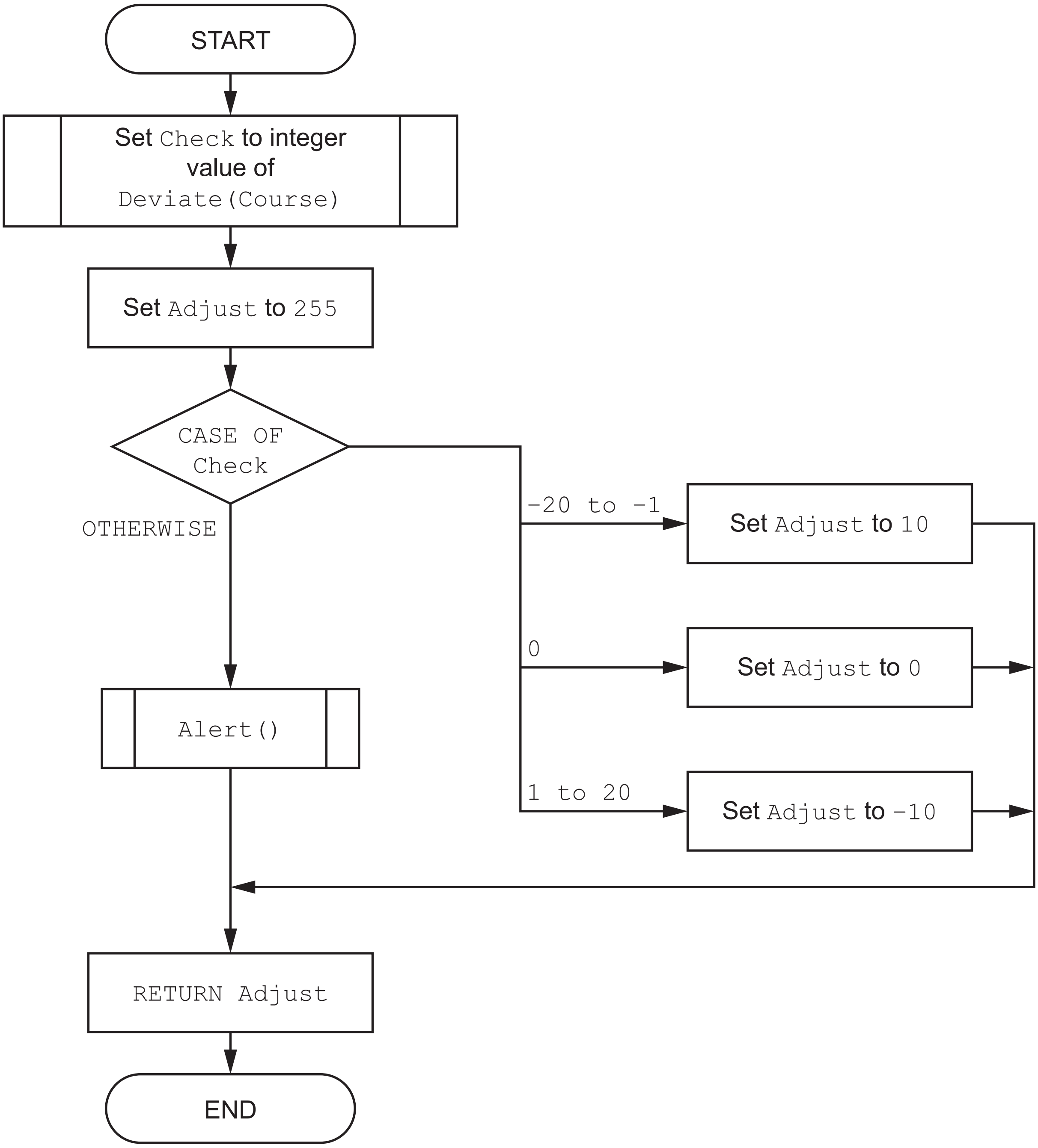
(ii) Complete the structure chart for the six modules described in **part (b)**.

[6]

3 A navigation program includes a function, `CheckCourse()`. This function is called with a real value, `Course`, and returns an integer value.

The identifier table and the program flowchart for the function are shown as follows:

Identifier	Type	Description
`Course`	`REAL`	The value passed to `CheckCourse()`
`Adjust`	`INTEGER`	The value returned by `CheckCourse()`
`Check`	`INTEGER`	A local variable
`Deviate()`	`FUNCTION`	A function that is passed a `REAL` value representing the course and returns a `REAL` value representing the current deviation
`Alert()`	`PROCEDURE`	A procedure that generates a warning



Write **pseudocode** to implement the function `CheckCourse()`. The pseudocode must follow the algorithm represented by the flowchart. Declare any local variables used.

Refer to the **Appendix** on page 19 for a list of built-in pseudocode functions and operators.

.... [7]

4 (a) Mustafa has developed the following pseudocode to generate ten random integers in the range 1 to 100.

```
DECLARE Random : ARRAY [1:10] OF INTEGER
DECLARE Count, RNum : INTEGER

FOR Count ← 1 TO 10
   Rnum ← INT(RAND(100)) + 1
   Random[Count] ← RNum
ENDFOR
```

Refer to the **Appendix** on page 19 for a list of built-in pseudocode functions and operators.

Rewrite the pseudocode so that there are no duplicated numbers in the list of random numbers.

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

.... [6]

(b) The changes made to the pseudocode in **part (a)** were as a result of changes to the program requirement.

Give the term used to describe changes made for this reason.

.... [1]

5 A global 1D array, `Contact`, of type `STRING` is used to store a list of names and email addresses.

There are 1000 elements in the array. Each element stores one data item. The format of each data item is as follows:

```
<Name>':'<EmailAddress>
```

`Name` and `EmailAddress` are both variable-length strings.

For example:

```
"Sharma Himal:hsharma99@stlmail.com"
```

A function, `GetName()`, is part of the program that processes the array. A data item string will be passed to the function as a parameter. The function will return the `Name` part. Validation is **not** necessary.

(a) Use **structured English** to describe the algorithm for the function `GetName()`.

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

.... [7]

(b) **(i)** The array is to be sorted using an efficient bubble sort algorithm. An efficient bubble sort reduces the number of unnecessary comparisons between elements.

Describe how this could be achieved.

....

....

....

....

....

....

....

.... [4]

(ii) A procedure, `BubbleSort()`, is needed to sort the 1D array `Contact` into ascending order of `Name` using an efficient bubble sort algorithm.

Write **program code** for the procedure `BubbleSort()`.

Visual Basic and Pascal: You should include the declaration statements for variables.
Python: You should show a comment statement for each variable used with its data type.

Programming language ....

Program code ....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

.... [8]

Question 6 begins on the next page.

6 A company hires out rowing boats on a lake. The company has 17 boats numbered from 1 to 17.

Boats may be hired between 9:00 and 18:00, with a maximum hire duration of 90 minutes.

The company is developing a program to help manage and record the boat hire process.

The programmer has decided to store all values relating to hire time as strings. The program will use a 24-hour clock format. For example:

Time (in words)	**String value**
Nine o'clock in the morning	`"09:00"`
Five minutes past ten o'clock in the morning	`"10:05"`
Ten minutes before three o'clock in the afternoon	`"14:50"`

The programmer has defined the first module as follows:

Module	**Description**
`AddTime()`	• Takes two parameters: ◦ `StartTime`: a `STRING` value representing a time as described ◦ `Duration`: an `INTEGER` value representing a duration in minutes • Adds the duration to the time to give a new time • Returns the new time as a `STRING`

(a) (i) Write **pseudocode** for the module `AddTime()`. Assume both input parameters are valid.

Refer to the **Appendix** on page 19 for a built-in list of pseudocode functions and operators.

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

.... [8]

(ii) `AddTime()` will be tested using white-box testing.

State the reason for using white-box testing.

....

.... [1]

(iii) A run-time error is one type of error that black-box testing can reveal.

Describe **one other** type of error that black-box testing can reveal.

....

.... [2]

(b) The user will input the desired start time of a hire. A new module will be written to validate the input string as a valid time in 24-hour clock format.

The string is already confirmed as being in the format `"NN:NN"`, where `N` is a numeric character.

Give an example of suitable test data that is in this format but which is **invalid**. Explain your answer.

Test data ....

Explanation ....

....

....

....

[2]

(c) Each time a boat is hired out, details of the hire are added to a text file, `Hirelog.txt`. Each line of the text file corresponds to information about one hire session.

The format of each line is as follows:

```
<BoatNumber><Date><AmountPaid>
```

- `BoatNumber` is a two-digit numeric string
- `Date` is a six-digit numeric string in the format DDMMYY
- `AmountPaid` is a variable-length string representing a numeric value, for example `"12.75"`

The total hire amount from each boat is to be stored in a global array, `Total`. This array is declared in pseudocode as follows:

```
DECLARE Total : ARRAY [1:17] OF REAL
```

The programmer has defined module `GetTotals()` as follows:

Module	Description
`GetTotals()`	• Search through the file `Hirelog.txt` • Extract the `AmountPaid` each time a boat is hired • Store the total of `AmountPaid` for each boat in the array

Write **program code** for the module `GetTotals()`.

Visual Basic and Pascal: You should include the declaration statements for variables.
Python: You should show a comment statement for each variable used with its data type.

Programming language ....

Program code

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

.... [8]

BLANK PAGE

Appendix

Built-in functions (pseudocode)

Each function returns an error if the function call is not properly formed.

LENGTH(ThisString : STRING) RETURNS INTEGER
returns the integer value representing the length of ThisString

Example: LENGTH("Happy Days") returns 10

LEFT(ThisString : STRING, x : INTEGER) RETURNS STRING
returns leftmost x characters from ThisString

Example: LEFT("ABCDEFGH", 3) returns "ABC"

RIGHT(ThisString: STRING, x : INTEGER) RETURNS STRING
returns rightmost x characters from ThisString

Example: RIGHT("ABCDEFGH", 3) returns "FGH"

INT(x : REAL) RETURNS INTEGER
returns the integer part of x

Example: INT(27.5415) returns 27

RAND(x : INTEGER) RETURNS REAL
returns a real number in the range 0 to x (not inclusive of x)

Example: RAND(87) could return 35.43

MOD(ThisNum : INTEGER, ThisDiv : INTEGER) RETURNS INTEGER
returns the integer value representing the remainder when ThisNum is divided by ThisDiv

Example: MOD(10,3) returns 1

DIV(ThisNum : INTEGER, ThisDiv : INTEGER) RETURNS INTEGER
returns the integer value representing the whole number part of the result when ThisNum is divided by ThisDiv

Example: DIV(10,3) returns 3

NUM_TO_STRING(x : REAL) RETURNS STRING
returns a string representation of a numeric value.

Example: If x has the value 87.5 then NUM_TO_STRING(x) returns "87.5"
Note: This function will also work if x is of type INTEGER

STRING_TO_NUM(x : STRING) RETURNS REAL
returns a numeric representation of a string.

Example: If x has the value "23.45" then STRING_TO_NUM(x) returns 23.45
Note: This function will also work if x is of type CHAR

Operators (pseudocode)

Operator	Description
&	Concatenates (joins) two strings Example: "Summer" & " " & "Pudding" produces "Summer Pudding"
AND	Performs a logical AND on two Boolean values Example: TRUE AND FALSE produces FALSE
OR	Performs a logical OR on two Boolean values Example: TRUE OR FALSE produces TRUE

BLANK PAGE