

**CAMBRIDGE INTERNATIONAL EXAMINATIONS**

Cambridge International Advanced Subsidiary and Advanced Level

## **MARK SCHEME for the October/November 2015 series**

### **9608 COMPUTER SCIENCE**

**9608/22**

Paper 2 (Written Paper), maximum raw mark 75

This mark scheme is published as an aid to teachers and candidates, to indicate the requirements of the examination. It shows the basis on which Examiners were instructed to award marks. It does not indicate the details of the discussions that took place at an Examiners' meeting before marking began, which would have considered the acceptability of alternative answers.

Mark schemes should be read in conjunction with the question paper and the Principal Examiner Report for Teachers.

Cambridge will not enter into discussions about these mark schemes.

Cambridge is publishing the mark schemes for the October/November 2015 series for most Cambridge IGCSE<sup>®</sup>, Cambridge International A and AS Level components and some Cambridge O Level components.

® IGCSE is the registered trademark of Cambridge International Examinations.

- 1 (i) 2 [1]
- (ii) 7.5 [1]  
Accept: 7 ½
- (iii) FALSE [1]
- (iv) TRUE [1]
- (v) ERROR [1]

2 (a)

Test Case	Inputs		Output	
	P	Q	X	
1	1	1	<b>1</b>	[1]
2	1	0	<b>0</b>	[1]
3	0	1	<b>0</b>	[1]
4	0	0	<b>0</b>	[1]

(b)

```
IF P = 1 AND Q = 1
  THEN
    X ← 1
  ELSE
    X ← 0
ENDIF
```

```
IF P = 0 OR Q = 0
  THEN
    X ← 0
  ELSE
    X ← 1
ENDIF
```

**Mark as follows:**

**Structure:** IF – THEN – ELSE – ENDIF [1]

**Condition:** P = 1 AND Q = 1 [1]

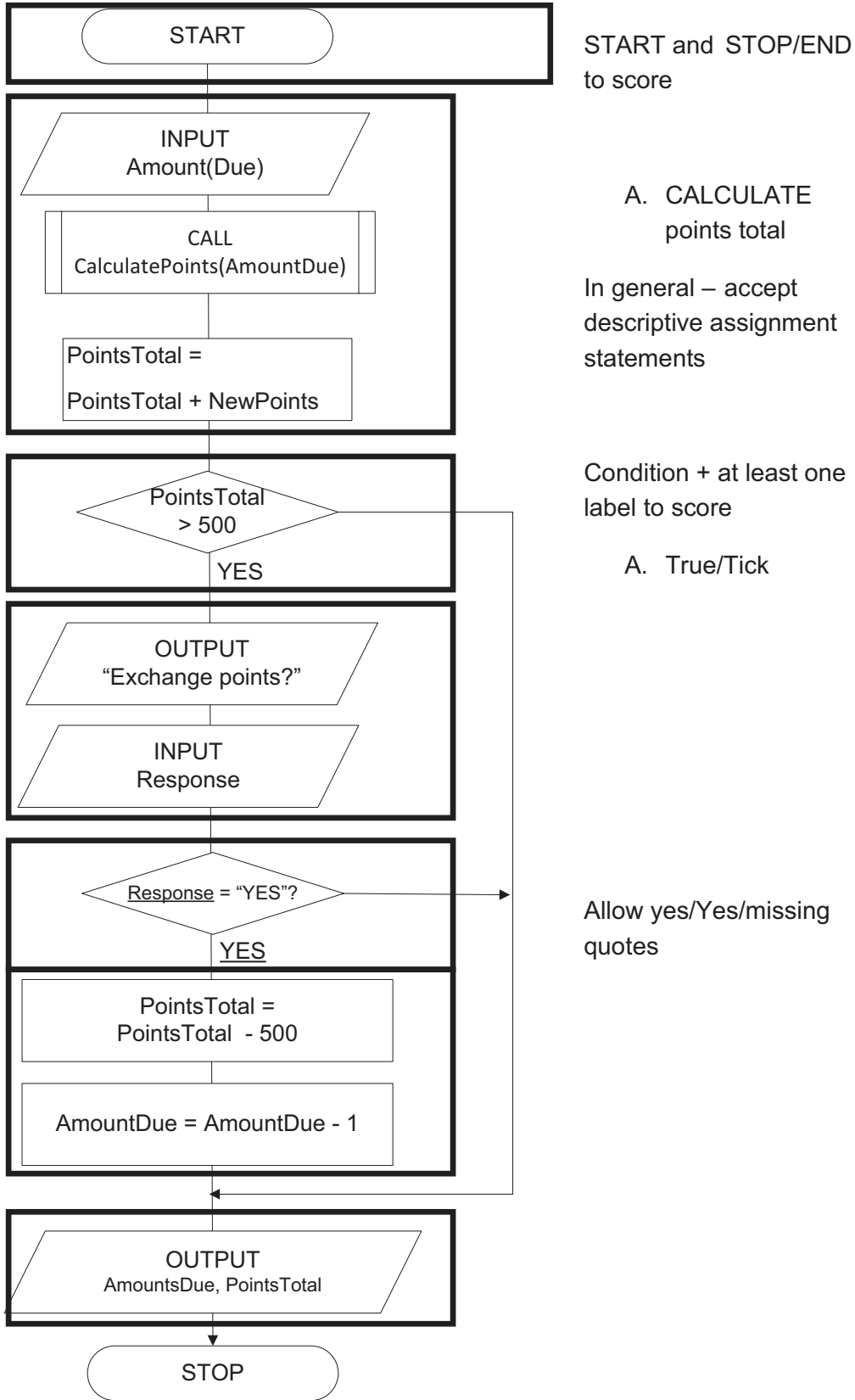
Allow &/&& for the operator

**Logic:** X ← 1 (for TRUE) }  
X ← 0 (for FALSE) } [1]

Check carefully for:

- other alternative correct algorithm
- a 'mirror copy' of the question paper algorithm – score 0

3



[Max 6]

Page 4	Mark Scheme	Syllabus	Paper
	Cambridge International AS/A Level – October/November 2015	9608	22

- 4 (a) The combination of suit and card number // the 'pair' of numbers // the pair of random numbers .... [1]  
 There will be duplicates/repeats//not all cards will be drawn [1]
- (b) (i) 32 // 33 [1]  
 (ii) 27 // 28 [1]  
 (iii) 08 [1]  
 (iv) 12 // 13 [1]
- (c) 1 [1]
- (d) DealCount <> 52 // NewCard = FALSE  
 Allow: Inclusion of the WHILE [1]
- (e) Test has the card has already been drawn? [1]  
 Set value TRUE for this card entry (in the array)/this card [1]  
 Flags that this is the first time this card has been drawn // decides if another card must be generated [1]  
 Outputs the new card value [1]
- [Max 2]
- (f) CardPack ARRAY[1:4 , 1:13] OF/:/AS BOOLEAN [1]  
 Allow: parentheses
- (g) **Pseudocode ...**  
 (SELECT) CASE (OF) CardValue + ENDCASE [1]  
 (CASE) 1: CardName ← "Ace" 1 mark for any one correct [1]  
 (CASE) 11: CardName ← "Jack"  
 (CASE) 12: CardName ← "Queen"  
 (CASE) 13: CardName ← "King" (final three cases ...) [1]  
 OTHERWISE (/ELSE) CardName ← CardValue //  
 (CASE) 2 TO 10: CardName ← CardValue [1]  
 ENDCASE // ENDSELECT

Note: Must be double quotes present and correct case

<b>Page 5</b>	<b>Mark Scheme</b>	<b>Syllabus</b>	<b>Paper</b>
	<b>Cambridge International AS/A Level – October/November 2015</b>	<b>9608</b>	<b>22</b>

**Visual Basic**

```
Select Case CardValue
```

```
Case 1
```

```
CardName = "Ace"
```

```
Case 11
```

```
CardName = "Jack"
```

```
Case 12
```

```
CardName = "Queen"
```

```
Case 13
```

```
CardName = "King"
```

```
Case Else // Case 2 to 10
```

```
CardName = Str(CardValue) [4]
```

```
End Select
```

Allow: omission of Str

5 (a) (i)

N	i	j	Temp	1	2	3	4	5
				11	16	13	7	8
5								
	1	1						
		2			13	16		
		3				7	16	
		4					8	16
		5						
	2	1						
		2			7	13		
		3				8	13	
	3	1		7	11			
		2			8	11		
		3						
	4	1		<i>final values are:</i>				
		2		7	8	11	13	16

Terminates at 5 and 4 respectively

Must be:

- Preceded by other entries ...
- nothing after the 1, 2 sequence

Note these final values will not be shown on the same row ...

[8]

(ii) To sort / to order/put in ascending order the items (in the array) [1]

(iii) There were no swaps on the last pass / on pass 4 [1]

(b)

Identifier	Data Type	Description
Num		
N	INTEGER	The number of numbers in the list
i	INTEGER	Loop counter // The number of 'passes' up through the list
j	INTEGER	The <u>index</u> // position in the array
Temp	INTEGER	Description must imply/states the 'swapping' operation

Mark as follows:

INTEGER × 4

One mark per description

[1]

[4]

6 (a) (i) 12

[1]

(ii) 'L'

Note: quotes are optional – must be upper case L

[1]

(b) (i)

Identifier	Data Type	Description
InputString	STRING	The string value input by the user
i	INTEGER	<u>Loop</u> counter // (index) position of an individual character
j	INTEGER	Number of characters in / length of InputString
NextChar	CHAR//CHARACTER	(Single) character within InputString / from string input by the user
NewString	STRING	The string formed/made/created//output Allow: if "by the user" added

[1]

[1]

[1]

[1]

Note: Correct (identifier + the data type + description) needed to score

Page 8	Mark Scheme	Syllabus	Paper
	Cambridge International AS/A Level – October/November 2015	9608	22

```

(ii) // main program
INPUT MyString
ChangedString ← RemoveSpaces(MyString) (1)
OUTPUT ChangedString

// function definition (1)
FUNCTION RemoveSpaces(InputString : STRING)
RETURNS STRING (1)
  DECLARE i :/AS INTEGER } (1)
  DECLARE j :/AS INTEGER }
  DECLARE NextChar :/AS CHAR } (1)
  DECLARE NewString :/AS STRING }

  NewString = "" (1)

  j ← CharacterCount(InputString)
  FOR i ← 1 TO j
    NextChar ← OneChar(InputString, i)
    IF NextChar <> " "
      THEN
        // the & character joins together two strings
        NewString ← NewString & NextChar
      ENDIF
  ENDFOR (1) (1) only awarded if follows the previous mark

RETURN NewString //
RemoveSpaces ← NewString
ENDFUNCTION

```

[Max 7]

- 7 (a) (i) 165 [1]
- (ii) "YES" Quotes optional [1]
- (iii) 9 [1]
- (iv) 83 [1]



Page 9	Mark Scheme	Syllabus	Paper
	Cambridge International AS/A Level – October/November 2015	9608	22

(b) (i) Use of correct identifiers only to score

*Declaration/Commenting of variables*

```
MyMessage As String
EncryptString As String
i As Integer
NextNum As Integer
```

At least two variables correctly documented [1]

*Input of string ...*

Correct syntax (for both prompt and assignment) and ...  
Uses the MyMessage identifier [1]

EncryptString set to 'empty string' [1]

Note: Must suggest 'empty' string

*For loop ...*

FOR – NEXT keywords // (Python) correct indentation [1]

Correct start/end boundaries [1]

Note: the end boundary must use the language length  
function/method //alternative Python syntax

Isolate single character [1]

Use of language functions to calculate new number and ....  
Assigned to NextNum [1]

Conversion of NextNum to a character and concatenated  
to EncryptString [1]

Correct syntax for output of EncryptString [1]

[MAX 8]

**SAMPLE CODE**

**PYTHON**

```
MyMessage = input("Enter message : ")
EncryptString = ""
for i in range(0, len(MyMessage)) :
    NextNum = ord(MyMessage[i]) + 3
    EncryptString = EncryptString + chr(NextNum)
print(EncryptString)
```

*Alternative solution:*

```
MyMessage = input("Enter message : ")
EncryptString = ""
for NextChar in MyMessage :
    NextNum = ord(NextChar) + 3
    EncryptString = EncryptString + chr(NextNum)
print(EncryptString)
```

<b>Page 10</b>	<b>Mark Scheme</b>	<b>Syllabus</b>	<b>Paper</b>
	<b>Cambridge International AS/A Level – October/November 2015</b>	<b>9608</b>	<b>22</b>

**VB**

```
Dim MyMessage, EncryptString As String
Dim NextNum, i As Integer
Console.WriteLine("Enter message : ")
MyMessage = Console.ReadLine()
EncryptString = ""
For i = 1 To Len(MyMessage)
    NextNum = Asc(Mid(MyMessage, i, 1)) + 3
    EncryptString = EncryptString + //& Chr(NextNum)
Next
Console.WriteLine(EncryptString)
```

**Alternatives:**

```
GetChar(MyMessage, i)
MyMessage.Substring(i, 1)
```

**Allow: Use of InputBox and MsgBox**

**Alternative solution :**

```
Dim MyMessage, EncryptString As String
Dim NextNum, i As Integer
Console.WriteLine("Enter message : ")
MyMessage = Console.ReadLine()
EncryptString = ""
For i = 0 To Len(MyMessage) - 1
    NextNum = Asc(MyMessage.Chars(i)) + 3
    EncryptString = EncryptString + Chr(NextNum)
Next
Console.WriteLine(EncryptString)
```

**PASCAL**

```
var
    MyMessage, EncryptString : string;
    NextNum, i : integer;
begin
    write('Enter message : ');
    readln(MyMessage);
    EncryptString := '';
    for i := 1 to length(MyMessage) do
        begin
            NextNum := ord(MyMessage[i]) + 3;
            EncryptString := EncryptString + chr(NextNum);
        end;
    writeln(EncryptString);
end.
```

- (ii) For each/every character .... [1]  
 A replacement character is 'calculated' from its ASCII value // or by example ... [1]