
COMPUTER SCIENCE

9608/21

Paper 2 Written Paper

October/November 2016

MARK SCHEME

Maximum Mark: 75

Published

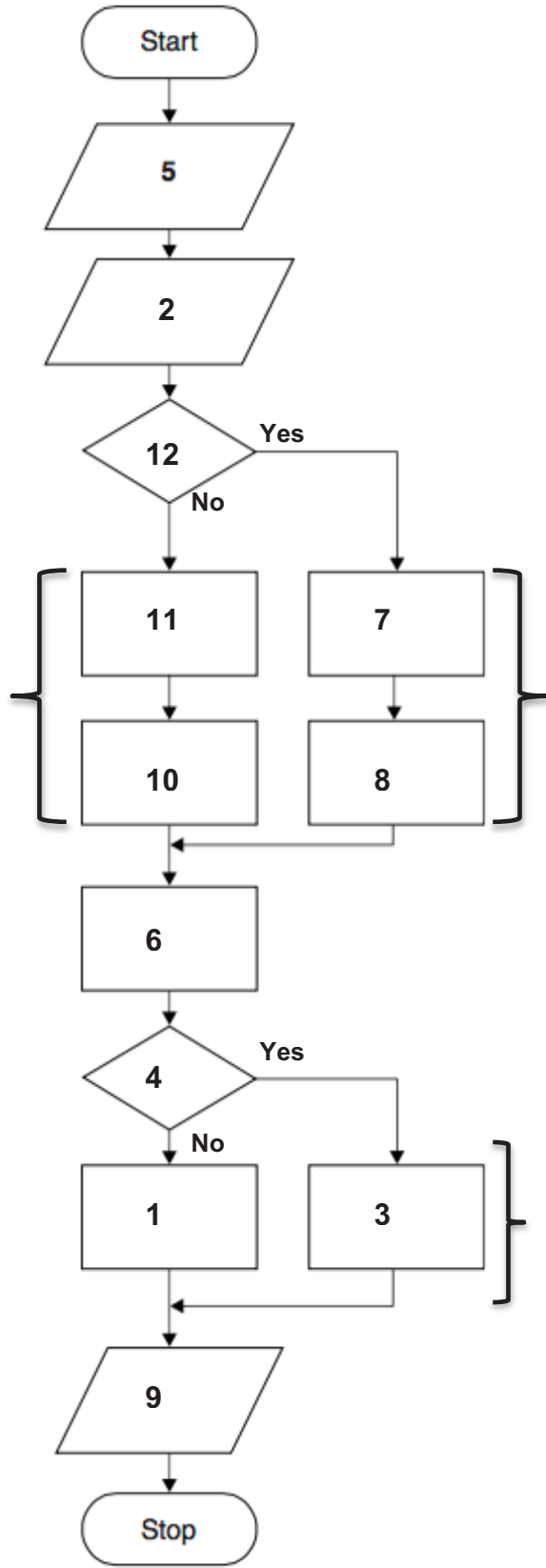
This mark scheme is published as an aid to teachers and candidates, to indicate the requirements of the examination. It shows the basis on which Examiners were instructed to award marks. It does not indicate the details of the discussions that took place at an Examiners' meeting before marking began, 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 2016 series for most Cambridge IGCSE[®], Cambridge International A and AS Level components and some Cambridge O Level components.

1 (a)



Note: Order of 11, 10 and 7,8 may be reversed.

Page 3	Mark Scheme	Syllabus	Paper
	Cambridge International AS/A Level – October/November 2016	9608	21

One mark for each of the following symbols / symbol combinations:

- 2
- 7 and 8 from YES
- 10 and 11
- 6
- 1 and 3 (1 from NO, 3 from YES)
- 9
- 12 and 4

Max [6]

(b) Rows 2 to 7 are examples only

TicketType	BaggageWeight	Explanation	Expected output
E	15	Under the allowance	0
E	> 16	Under the allowance	<i>Charge</i>
S	<= 20	Under the allowance	0
S	> 20	Under the allowance	<i>Charge</i>
E	16	Boundary weight for a type E ticket	0
S	20	Boundary weight for a type S ticket	0
E or S	negative or non-numeric	Invalid weight	Error message

Ticket type	Baggage allowance (kg)	Charge rate per additional kg (\$)
'E'	16	3.50
'S'	20	5.75

One mark for each different test (examples above)

Max [5]

```
(c) INPUT TicketType
    WHILE NOT (TicketType = 'E') OR (TicketType = 'S')
        INPUT TicketType
    ENDWHILE
```

One mark for each of:

- WHILE ... ENDWHILE
 - Correct condition in a loop
 - INPUT within loop plus one before loop // alternative arrangement leading to correct exit from loop
- [3]**

2 (a)

Status2	ReadingCount	ThisBit	BitCount	OUTPUT
			0	
1	1	1	1	
	2	0	1	
	3	1	2	
	4	1	3	
	5	1	4	
	6	0	4	
1				
	1	1	5	Error – Investigate
			0	
	2	1	1	
	3	0	1	
	4	0	1	
	5	1	2	
	6	1	3	
0				

1 must 'follow' 6 as shown by arrow. Can have only 1 or nothing above.

1

Error – Investigate

0 must 'follow' 6 as shown by arrow. Can have only 1 or nothing above.

0

One mark per area outlined

[7]

Page 5	Mark Scheme	Syllabus	Paper
	Cambridge International AS/A Level – October/November 2016	9608	21

(b) One mark for each of:

- Assignment: 01 // 02 // 06 // 09 // 14 // 18
- Selection: 07 // 11
- Iteration: 03 // 05

[3]

3 (a) (i) 7

[1]

(ii) 103

[1]

(iii) 'K'

[1]

(iv) "come"

[1]

(b) (i) PROCEDURE CalculateCustomerID
 OUTPUT "Key in surname"
 INPUT Surname
 Length ← **CHARACTERCOUNT (Surname)**
 CustomerID ← 0
 FOR i ← 1 TO Length
 //NextChar is a single character from Surname
 Nextchar ← 1 **SUBSTR(Surname, i, 1)** // **ONECHAR(Surname, i)**
 NextCodeNumber ← ASC(NextChar)
 CustomerID ← CustomerID + **NextCodeNumber**
 ENDFOR
 OUTPUT "Customer ID is ", CustomerID
 ENDPROCEDURE

One mark per phrase in **bold**

[3]

Page 6	Mark Scheme	Syllabus	Paper
	Cambridge International AS/A Level – October/November 2016	9608	21

- (ii) 'Pseudocode' solution included here for development and clarification of mark scheme. Programming language example solutions appear in the Appendix.

```

PROCEDURE CalculateCustomerID
  DECLARE Surname : STRING
  DECLARE NextChar : CHAR
  DECLARE NextCodeNumber, i, CustomerID, SLength : INTEGER
  OUTPUT "Key in surname"
  INPUT Surname
  SLength ← LEN(Surname)
  CustomerID ← 0
  FOR i ← 1 TO SLength
    //NextChar is a single character from Surname
    Nextchar ← MID(Surname, i, 1)
    NextCodeNumber ← ASC(NextChar)
    CustomerID ← CustomerID + NextCodeNumber
  ENDFOR
  OUTPUT "Customer ID is ", CustomerID
ENDPROCEDURE

```

Mark as follows:

- Declaration of Surname as STRING and NextChar as CHAR and any three INTEGERS
- Prompt and Input
- Calculation of string length
- FOR Loop to process all characters in the string
- Assignment to NextChar in a loop
- Assignment to NextCodeNumber in a loop
- Totalling CustomerID in a loop
- Output following a loop

[6]

- (c) (i) Visual Basic

Function CalculateCustomerID(ByVal AnyName AS STRING) As Integer

Pascal

FUNCTION CalculateCustomerID(AnyName : STRING) : INTEGER

Python

def CalculateCustomerID(AnyName) :

Mark as follows:

- Correct keyword + Function name
- Single input parameter of correct type
- Return parameter type

[3]

- (ii) Visual Basic

Return customerID // CalculateCustomerID = CustomerID

Pascal

Result := CustomerID // CalculateCustomerID := CustomerID

Python

Return CustomerID

[1]

Page 7	Mark Scheme	Syllabus	Paper
	Cambridge International AS/A Level – October/November 2016	9608	21

(iii) Visual Basic

ThisID = CalculateCustomerID ("Wilkes")

Pascal

ThisID := CalculateCustomerID ('Wilkes')

Python

ThisID = CalculateCustomerID ("Wilkes")

One mark per underlined element

[3]

- (d) (i)
- Built-in functions are made available by the programming language / already in the system
 - Built-in functions are ready made and tested
 - User-defined functions can be modified // built-in cannot be modified
 - User defined functions can be designed to meet the user's requirements
 - User-defined functions can only be used in that program / module

[Max 2]

- (ii)
- They have an identifier name
 - They return a value
 - They have none, one or more arguments
 - Both perform a specific task
 - Both represent re-usable code
 - Both are 'called'

[Max 2]

- 4 (a)
- Create / modify the source code using the text editor
 - Compiler translates the source code
 - Compiler produces the object code

[Max 3]

- (b) (i)
- Errors in keywords are highlighted // before the compilation process
 - Provides line-by-line syntax checking as code is typed in
 - Provides line number of the error
 - Display of known identifier names
 - Auto-complete
 - Colour-coding
 - Auto-indent
 - type checking
 - Subroutine parameter checking

[Max 1]

- (ii)
- Set break-points
 - Single step / step into/over subroutine
 - Window to watch the changing value of variables

[Max 1]

Page 8	Mark Scheme	Syllabus	Paper
	Cambridge International AS/A Level – October/November 2016	9608	21

(c) (i) OPEN "PRODUCTS" FOR READ
i ← 1
WHILE NOT EOF("PRODUCTS")

```

    READFILE ("PRODUCTS", PCode[i])
    READFILE ("PRODUCTS", PDescription[i])
    READFILE ("PRODUCTS", Temp // PRetailPrice[i])
}

    PRetailPrice[i] ← TONUM(Temp)

```

```

    i ← i + 1
ENDWHILE

```

```

CLOSE "PRODUCTS"
OUTPUT "Product file contents written to arrays"

```

One mark per bold phrase (three READFILE() counts as a single mark)

[5]

(ii) Benefit:

- The number of file read operations is reduced (by 2/3rds)
- It may use less storage / space in the file if strings are NOT fixed length
- All the data related to a single product is read at once / in one file operation / grouped together

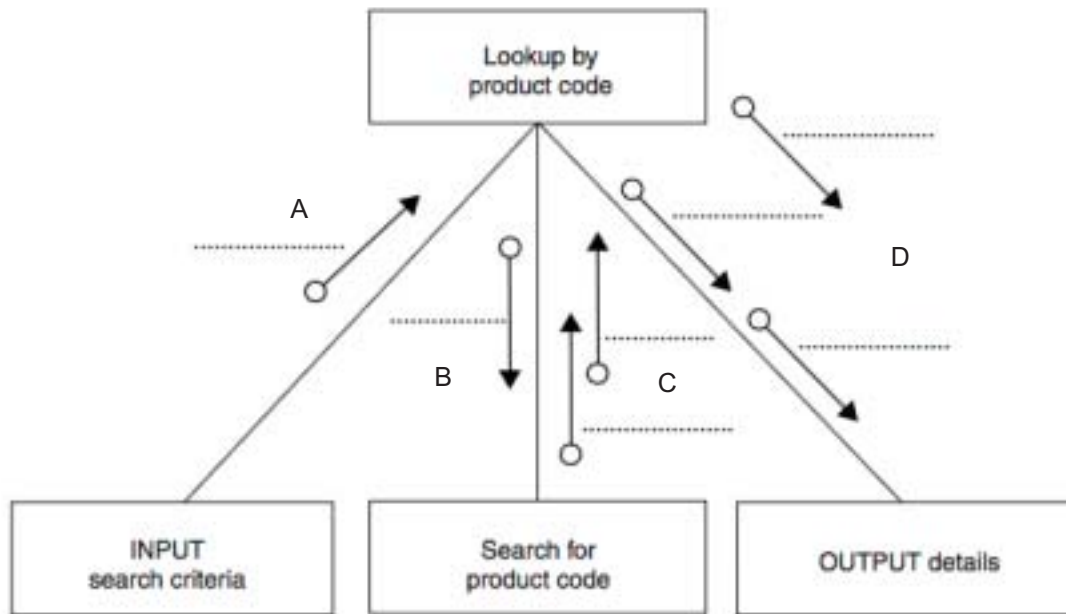
Drawback:

- The program will need to use the string handling functions to isolate each of the three items of data
- Difficult to isolate data items if the format is not consistent
- More difficult to search

Max one benefit and one drawback

[2]

(d)



One mark per group (one or more names) as follows:

- A: SearchCode
- B: SearchCode // ThisIndex
- C: ThisRetailPrice, ThisDescription
- D: SearchCode, ThisDescription, ThisRetailPrice

[4]

Page 10	Mark Scheme	Syllabus	Paper
	Cambridge International AS/A Level – October/November 2016	9608	21

- (e) 'Pseudocode' solution included here for development and clarification of mark scheme. Programming language example solutions appear in the Appendix.

```

FUNCTION ProductCodeSearch(AnyName : String) RETURNS : Integer
  DECLARE FoundPos : Integer
  DECLARE i : Integer

  i ← 1
  FoundPos ← -1

  REPEAT
    IF AnyName = PCode[i]
      THEN
        FoundPos ← i
      ELSE
        i ← i + 1
      ENDIF
  UNTIL (i = 1001) OR (FoundPos <> -1)

  RETURN FoundPos

ENDFUNCTION

```

Mark as follows:

- Function header returns INTEGER
- Initialisation of index variable
- Loop through array PCode (including exit when found)
- Comparison of AnyName with PCode[i] in a loop
- Increment index variable in a loop
- Return index if AnyName found AND return -1 if AnyName not found **[Max 6]**

- | | | | |
|----------|--------------|-----------|------------|
| 5 | (i) | 13 / 13.0 | [1] |
| | (ii) | 18.6 | [1] |
| | (iii) | TRUE | [1] |
| | (iv) | 32 | [1] |
| | (v) | 22 | [1] |

*** End of Mark Scheme – Example program code solutions follow ***

Page 11	Mark Scheme	Syllabus	Paper
	Cambridge International AS/A Level – October/November 2016	9608	21

Appendix – Example program code solutions

3(b)(ii): Visual Basic

```

Dim Surname As String
Dim NextChar As Char
Dim NextCodeNumber As Integer
Dim i As Integer
Dim CustomerID As Integer
Dim SLength As Integer

Console.Write("Key in surname ")
Surname = Console.ReadLine
SLength = Len(Surname)
CustomerID = 0
    For i = 1 To SLength
        \\ NextChar is a single character from surname
        NextChar = Mid(Surname, i, 1)
        NextCodeNumber = Asc(NextChar)
        CustomerID = CustomerID + NextCodeNumber
    Next

    Console.WriteLine("Customer ID is " & CustomerID)

```

3(b)(ii): Pascal

```

Var Surname : string;
    SLength, i, CustomerID, NextCodeNumber : integer;
    NextChar : char;
begin
    Writeln ('Enter the surname: ');
    Readln (Surname);
    SLength := Length(Surname);
    CustomerID := 0;
    For i := 1 to SLength do
        begin
            NextChar := SurName[i];
            NextCodeNumber := Ord(NextChar);
            CustomerID := CustomerID + NextCodeNumber;
        end;
    Writeln ('Customer ID is ', CustomerID);
    Readln;
end.

```

Page 12	Mark Scheme	Syllabus	Paper
	Cambridge International AS/A Level – October/November 2016	9608	21

3(b)(ii): Python

```
# Surname String
# NextChar Char
# NextCodeNumber, I, CustomerID, SLength Integer

Surname = input("Key in Surname ")
SLength = len(Surname)
CustomerID = 0

for i in range(SLength):
    # NextChar is a single character from surname
    NextChar = Surname[i]
    NextCodeNumber = ord(NextChar)
    CustomerID = CustomerID + NextCodeNumber

print("Customer ID is " + str(CustomerID))
```

4(e): Visual Basic

```
Function ProductCodeSearch(ByVal SearchCode As String) As Integer
    Dim FoundCode As Integer
    Dim i As Integer

    i = 1
    FoundCode = -1

    Do
        If SearchCode = PCode(i) Then
            FoundCode = i
        Else
            i = i + 1
        End If
    Loop Until i = 1001 Or FoundCode <> -1
    Return FoundCode
End Function
```

Page 13	Mark Scheme	Syllabus	Paper
	Cambridge International AS/A Level – October/November 2016	9608	21

4(e): Pascal

```

Function ProductCodeSearch (SearchCode : String): integer;

    var FoundCode, ThisIndex : integer;
        Found : Boolean;

Begin
    Found := false;
    ThisIndex := 1;
    Repeat
        If SearchCode = PCode[ThisIndex] then
            Begin
                FoundCode := ThisIndex;
                Found := true;
            Else
                ThisIndex := ThisIndex + 1;
            end;
        Until (ThisIndex = 1001) OR (Found);
        If Found = false then
            FoundCode := -1
        ProductCodeSearch := FoundCode;
    end.

```

4(e): Python

```

def ProductCodeSearch(SearchCode):
    # list indexes start at zero
    i = 0
    Found = "no"
    while not(i == 1001 or Found == "yes"):
        if SearchCode == PCode[i]:
            Found = "yes"
            FoundIndex = i
        else:
            i = i + 1

    if Found == "no":
        FoundIndex = -1

    return FoundIndex

```