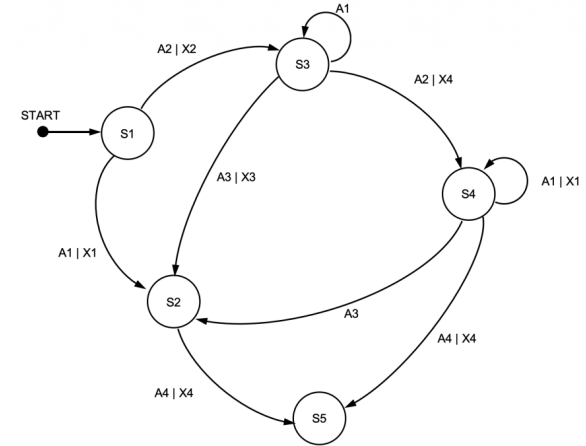
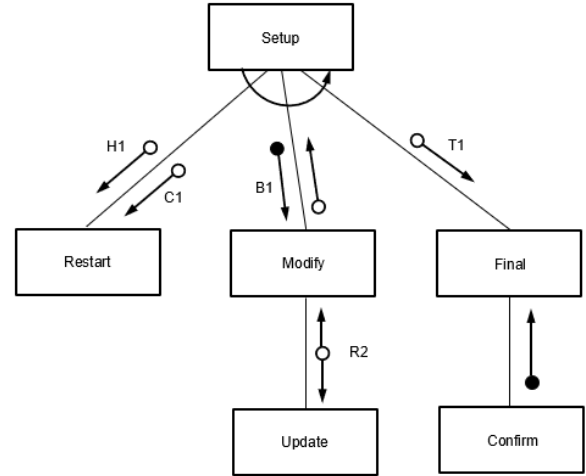


1(a)	Expression	
	<code>MyString ← 'X' & MyString</code>	
	<code>MyChar ← MID ("ABCD", 1 / 2 / 3 / 4 , 1)</code>	
	<code>MyString ← NUM_TO_STR (DAY / MONTH / YEAR / DAYINDEX (MyDOB))</code>	
1(b)	Test description	
	carried out as soon as a program module has been coded	alpha
	carried out as program modules are being combined	integration
	completed by the developers without referring to the code	black-box
1(c)	Test method	
	completed by the customer	acceptance / beta
	One mark per row	
	One mark per row (2 to 4)	
1(c)	One mark per point:	
	1 reference to a breakpoint	
	2 reference to single stepping	
	3 Correct explanation of both e.g. to stop program execution at specific point / line / statement / instruction and to execute one line / statement / instruction at a time to (locate an/the error)	
	One mark for each bulleted point	

7(a)		
	One mark for each:	
	1 Label A1 X1 added on line from S1 to state labelled S2	
	2 States S3, S4 and S5 labelled	
7(b)		
	One mark per point:	
	1 All boxes correctly labelled	
	2 Iteration arrow	
	3 Parameter to Restart	
	4 Return value from Modify and parameter to Final	
	5 BYREF parameter to Update	

8(a)

Example solution:

```

PROCEDURE OKToBorrow(ThisStudentID : STRING)
  DECLARE Index, Count : INTEGER

  CONSTANT Max = 5
  Count ← 0
  Index ← 1 / 0

  WHILE Index <= 5000 / Index <= 4999 AND Count < Max
    IF Loan[Index].StudentID = ThisStudentID AND
      Loan[Index].OnLoan =
TRUE THEN
      Count ← Count + 1
    ENDIF
    Index ← Index + 1
  ENDWHILE

  IF Count < Max THEN
    OUTPUT "Student may borrow another book"
  ELSE
    OUTPUT "Student may not borrow another book"
  ENDIF

ENDPROCEDURE

```

Mark as follows:

- 1 Procedure heading, parameter and ending
- 2 Loop through **all elements** in Loan array
- 3 ... terminating if Max reached
- 4 Test for correct StudentID field **in a loop**
- 5 ... and that OnLoan = TRUE **in a loop**
... if so, then increment Count **in a loop**
- 6 Output an appropriate **message** in both cases **after the loop**

8(b)

Example solution:

```

FUNCTION ReturnBook(ThisStudentID, ThisBookID : STRING)
  RETURNS BOOLEAN

  DECLARE Index : INTEGER

  Index ← 1

  WHILE Index < 5001
    IF Loan[Index].StudentID = ThisStudentID AND
      Loan[Index].BookID = ThisBookID THEN
      IF Loan[Index].OnLoan = TRUE THEN // Not
        returned
        Loan[Index].OnLoan ← FALSE // Mark as
        returned now
        RETURN TRUE //Found and not already returned
      ELSE
        RETURN FALSE // Found and already returned
      ENDIF
    ENDIF
    Index ← Index + 1
  ENDWHILE

  RETURN False // loan not found
ENDFUNCTION

```

Mark as follows:

- 1 Loop through **all elements** in Loan array
- 2 ...and book loan not found
- 3 Attempt to reference an individual data item **in a loop**
- 4 Correct test for StudentID **and** correct BookID **in a loop**
- 5 ... If book not returned yet, then set OnLoan field to FALSE
- 6 RETURN TRUE if book loan Found and not already returned
- 7 RETURN FALSE if book loan Found and loan already returned
- 8 RETURN FALSE if book loan not found

Alterative example solution – not immediate return

```

FUNCTION ReturnBook(ThisStudentID, ThisBookID : STRING)
  RETURNS BOOLEAN

  DECLARE Index : INTEGER
  DECLARE Found, OK : BOOLEAN

  Index ← 1
  Found ← FALSE
  OK ← TRUE

```

8(b)	<pre> WHILE Index < 5001 AND NOT Found IF Loan[Index].StudentID = ThisStudentID AND ___ Loan[Index].BookID = ThisBookID THEN Found ← TRUE IF Loan[Index].OnLoan = FALSE THEN // Already returned OK ← FALSE ELSE Loan[Index].OnLoan ← FALSE // Mark as returned now ENDIF ENDIF Index ← Index + 1 ENDWHILE RETURN Found AND OK ENDFUNCTION </pre>
8(c)	Award marks for reference to following design features: Change to existing Record: 1 Store date due back as a new data item / in the loan record 2 Store date of loan and the loan length category as new data items / in the loan record Max 1 New Record 3 New Records with Category (and Loan length field) Change to Program: 4 (Use the loan length category of a book to) calculate the date due date 5 Compare due date with today's date (to check if overdue / to calculate fine) 6 Map loan length category to length of loan / date due back Max 1 Max 2 marks

3(a)	One mark per mark point (Max 2) MP1 Protocols set a standard for communication // Protocols establish a standard set of rules for communication MP2 Protocols enable compatibility between devices from different manufacturers/platforms MP3 Two devices wouldn't be able to communicate/send messages to each other if they were using different protocols
------	--

3(b)	One mark for a protocol and one mark for a description (Max 4) Example answers: SMTP [1] a protocol used to send emails between mail servers // a protocol used to send emails from a computer to a mail server [1] IMAP [1] allows users to access/read their emails from any device without removing the message from the mail server // synchronises emails on any device [1]
3(c)	Two marks for each description mark as 2 x 2 (Max 4) Example answers: Packets checked at receiving end / on arrival [1] ... if packets arrive damaged or don't arrive at all, a re-send request is sent [1] Packets routed through different paths / sent individually [1] ... if a route is blocked, the packet is sent through a different route to ensure it arrives [1] If the packet's hop count is exceeded [1] ... the packet will be retired, which can generate a re-send request [1]
4(a)	One mark per scheduling routine (Max 2) from: • Round robin • Shortest job first • First come first served • Shortest remaining time
4(b)	One mark for identification and one mark for a description (Max 4) Two from: Provision of a User Interface // Provision of a Graphical User Interface [1] Allows the user to interact with the computer in a more intuitive way // Icons and menus are used to control devices by simply 'pointing and clicking' [1] Use of device drivers [1] Makes it easier to control peripherals such as printers within the operating system of the computer rather than on the separate device itself [1] Device mapping [1] Different devices (physical and virtual) are easy to identify on the network, check their status, or use [1] The user interacts only with the Application / top layer (of the TCP/IP protocol suite) [1] leaving the lower layers and their complexities hidden from the user [1]

6(a)	
------	---

2(a)

One mark for two's complement version and one mark for denary version

Mantissa

0	1	1	1	1	1	1	1
---	---	---	---	---	---	---	---

Exponent

0	1	1	1	1	1	1	1
---	---	---	---	---	---	---	---

$127 \times 2^{120} \parallel 0.9921875 \times 2^{127} \parallel 127/128 \times 2^{127}$

2(b)

One mark per mark point for working (Max 2)

- number converted to binary e.g., positive binary version of 3.59375 = (0)11.10011
- negative two's complement version - bits flipped and 1 added = 100.01101
- $-4 + 1/4 + 1/8 + 1/32 \parallel -4 + 0.25 + 0.03125 \parallel -(64 + 32 + 16 + 2 + 1)/32$
- $-2^2 + 2^{-2} + 2^{-3} + 2^{-5}$
- $2^2 (-2^0 + 2^{-4} + 2^{-5} + 2^{-7})$

One mark per mark point

- correct mantissa
- correct exponent, with working seen.

Mantissa

1	0	0	0	1	1	0	1
---	---	---	---	---	---	---	---

Exponent

0	0	0	0	0	0	1	0
---	---	---	---	---	---	---	---

8(a)	<pre> FUNCTION DeleteComment(Line : STRING) RETURNS STRING DECLARE NewLine, TwoChars : STRING DECLARE Count, TrimTo : INTEGER CONSTANT Comment = "/" NewLine ← Line TrimTo ← 0 Count ← 1 WHILE Count < LENGTH(Line) AND TrimTo = 0 TwoChars ← MID(Line, Count, 2) // extract 2 chars IF TwoChars = Comment THEN TrimTo ← Count ENDIF Count ← Count + 1 ENDWHILE IF TrimTo <> 0 THEN NewLine ← LEFT(Line, TrimTo - 1) ENDIF RETURN NewLine ENDFUNCTION </pre> Mark as follows: - Loop to length of Line (parameter) // length of Line -1 - Terminate loop on first double slash - Attempt to extract one/two characters in a loop - Check for "/" after attempt at extraction in a loop - Record start position of comment // Calculate amount of trim required - Attempt to trim Line from start of comment - Completely correct trimmed Line from start of comment - Return Line after a reasonable overall attempt
------	--

8(b)

Example:

```

FUNCTION Stage_1(StudentName : STRING) RETURNS INTEGER

    DECLARE OldFile, NewFile, Line : STRING
    DECLARE Count : INTEGER

    OldFile ← StudentName & "_src.txt"
    NewFile ← StudentName & "_S1.txt"
    OPENFILE OldFile FOR READ
    OPENFILE NewFile FOR WRITE
    Count ← 0

    WHILE NOT EOF(OldFile)
        READFILE OldFile, Line
        Line ← DeleteComment(Line)
        IF LENGTH(Line) <> 0 THEN
            WRITEFILE NewFile, Line
            Count ← Count + 1
        ENDIF
    ENDWHILE

    CLOSEFILE OldFile
    CLOSEFILE NewFile

    RETURN Count
ENDFUNCTION

```

Mark as follows:

- 1 Generate filenames condone missing "_"
- 2 Open both files in correct modes **and** subsequently close
- 3 Loop to EOF(OldFile)
- 4 Read a line from OldFile **and** execute DeleteComment() **in a loop**
- 5 Skip blank lines after DeleteComment() **in a loop**
- 6 Write Line to stage 1 file **and** increment Count
- 7 Return the number of lines after a reasonable attempt at counting