AS · Cambridge (CIE) · Computer Science

1 hour 13 questions

Exam Questions

Arrays

Array basics / Array pseudocode

1 (a) A factory produces food items. The items must be used within a certain number of days after their production date. The number of days is known as the shelf life. It is different for each type of item but is always a whole number in the range 1 to 21 (inclusive).

The latest date that an item can be used is called the 'use-by' date.

A program is needed to produce labels which show the 'use-by' date.

Part of the program is a function GetDate() which will:

- take two parameters: a production date and a value representing the shelf life
- return the corresponding 'use-by' date.

The program contains a global 1D array DaysInMonth of type integer which stores the number of days in each month (index 1 is January):

Index	Value
1	31
2	28
3	31
4	30
11	30
12	31

Note: Leap years are **not** considered

An algorithm uses the array DaysInMonth to calculate a 'use-by' date. An alternative design would involve the use of multiple selection statements.

An array-based technique is often used when there is a large number of different values to check and where no pattern exists.

One advantage of using an array-based technique is the speed of execution compared to the use of multiple selection statements.

Give **two other** advantages of using an array for this type of operation rather than a solution based on multiple selection statements.

Answer

Mark Scheme and Guidance

One mark per point:

1. Fewer lines of code are needed
2. The program is simpler/ less complex // Program is easier to design / code / maintain / modify / test / debug
3. Direct access to days in a month / data (using month number as index) // Can use index / month to (directly) access days in month / data

Max 2 marks

(2 marks)

(b) Complete the pseudocode for the function GetDate().

Date functions from the **insert** should be used in your solution.

FUNCTION GetDate(ProductionDate : DATE, ShelfLife : INTEGER) RETURNS DATE

Answer

Mark Scheme and Guidance

Example Solution

```
FUNCTION GetDate(ProductionDate : DATE, ShelfLife : INTEGER) RETURNS DATE
DECLARE NewDate : DATE
DECLARE DD, MM, YY, LastDay : INTEGER

DD ← DAY(ProductionDate)
MM ← MONTH(ProductionDate)
YY ← YEAR(ProductionDate)
```

```
DD ← DD + ShelfLife
LastDay ← DaysInMonth[MM]

IF DD > LastDay THEN
MM ← MM + 1
DD ← DD – LastDay
IF MM = 13 THEN
MM ← 1
YY ← YY + 1
ENDIF
ENDIF
NewDate ← SETDATE(DD, MM, YY)
RETURN NewDate
ENDFUNCTION
```

Mark as follows:

1. Declare three variables of TYPE INTEGER to store DD, MM and YY
2. Correct use of date functions from Insert to extract three integer values representing DD, MM and YY and use/assign to a variable
3. Add parameter ShelfLife to DD and use / assign result to a variable
4. Extract LastDay from DaysInMonth using MM as Index
5. Test for new DD after end of month / DaysInMonth[MM]
   … and if DD > DaysInMonth[MM]:
6. MM is incremented by 1 and DD is set to DD – DaysInMonth[MM]
7. If MM = 13 / MM > 12 then increment YY **and** set MM to 1
8. Use of SETDATE to assign value to NewDate, NewDate must have been delared as a type DATE
9. Return date

Max 7 marks

(7 marks)

2 (a) An exam paper has a maximum of 75 marks. One of five pass grades (A to E) is assigned, depending on the mark obtained. The lowest mark for a given grade is known as the grade boundary. For example, if the grade boundary for an A grade is 65 marks, then any candidate who achieves a mark of 65 or above will be awarded an A. A grade of U is awarded for marks below the E grade boundary.

The five grade boundaries are stored in a global 1D array GradeBoundary of type integer.

For example:

Element	Value	Comment
GradeBoundary[1]	65	The minimum mark for an A grade.
GradeBoundary[2]	57	The minimum mark for a B grade.
GradeBoundary[3]	43	The minimum mark for a C grade.
GradeBoundary[4]	35	The minimum mark for a D grade.
GradeBoundary[5]	27	The minimum mark for an E grade.

A global 2D array Result of type integer contains candidate marks for the exam. Each row relates to one candidate. Column 1 contains the candidate mark and column 2 contains the unique candidate ID.

For example, for the fourth and fifth candidates:

Element	Mark
Result[4, 1]	56
Result[5, 1]	54

Element	ID
Result[4, 2]	1074832
Result[5, 2]	2573839

There are more rows in the array than candidates who sit the exam. Any unused rows will be at the end of the array.

Candidate papers that are given a mark within two marks of any grade boundary must be checked.

For example, given the values in the example grade boundaries above, any paper that was awarded between 41 and 45 marks (inclusive) would need to be checked.

A program is being written to identify papers that need to be checked.

The programmer has defined the first program module as follows:

Module	Description
CheckMark()	• called with a parameter of type integer representing a candidate mark • returns TRUE if the mark is within 2 of any of the five grade boundaries, otherwise returns FALSE

Write pseudocode for module CheckMark().

Answer

Mark Scheme and Guidance

Loop example solution

```
FUNCTION CheckMark(Mark : INTEGER) RETURNS BOOLEAN
DECLARE Index, Lower, Upper : INTEGER

FOR Index ← 1 TO 5
Lower ← GradeBoundary[Index] - 2
Upper ← GradeBoundary[Index] + 2
IF Mark >= Lower AND Mark <= Upper THEN
RETURN TRUE
ENDIF
NEXT Index

RETURN FALSE
ENDFUNCTION
```

Mark as follows:

1. Loop through all elements in GradeBoundary array
2. Attempt to calculate range, both Lower and Upper, **in a loop**
3. Completely correct range calculation **in a loop**
4. Test if given mark / parameter, is within range **in a loop**
5. Set variable / immediate RETURN if recheck required **in a loop**
6. Return Boolean in both cases following a reasonable attempt

Selection example solution

```
FUNCTION CheckMark(Mark : INTEGER) RETURNS BOOLEAN

CASE OF Mark
GradeBoundary[1] – 2 TO GradeBoundary[1] + 2 : RETURN TRUE
GradeBoundary[2] – 2 TO GradeBoundary[2] + 2 : RETURN TRUE
GradeBoundary[3] – 2 TO GradeBoundary[3] + 2 : RETURN TRUE
GradeBoundary[4] – 2 TO GradeBoundary[4] + 2 : RETURN TRUE
GradeBoundary[5] – 2 TO GradeBoundary[5] + 2 : RETURN TRUE
OTHERWISE : RETURN FALSE
ENDCASE

ENDFUNCTION
```

Mark as follows:

1. Correct syntax used for selection structure(s)
2. Correct checks for two ranges

3. Correct checks for three ranges
4. Correct checks for four ranges
5. Correct checks for all five ranges
6. Return Boolean in both cases following a reasonable

(6 marks)

(b) A second module is defined:

Module	Description
CheckAll()	• called with a parameter of type integer representing the number of candidate marks in the Result array • uses CheckMark() to check each candidate mark • for each paper that needs to be checked, write the corresponding candidate ID on a separate line in a new file named GRList.txt • outputs a message with a count of how many papers need to be checked

Write pseudocode for module CheckAll().

CheckMark() must be used to check each individual mark.

Answer

Mark Scheme and Guidance

Example Solution:

```
PROCEDURE CheckAll(CNum : INTEGER)
DECLARE Index, Count, ThisMark: INTEGER
```

```
Count ← 0
OPENFILE "GRList.txt" FOR WRITE
FOR Index ← 1 to CNum
ThisMark ← Result[Index, 1] // 2D array: mark + ID
IF CheckMark(ThisMark) = TRUE THEN
WRITE "GRList.txt", NUM_TO_STR(Result[Index, 2])
Count ← Count + 1
ENDIF
NEXT Index
CLOSEFILE "GRList.txt"
OUTPUT "There are ", Count, " papers to check"

ENDPROCEDURE
```

Mark as follows:

1. Procedure heading, parameter and ending.
2. Open file in write mode and subsequently close
3. Loop through all CNum candidates
4. Extract candidate mark from Result array **in a loop**
5. Use of CheckMark() with candidate mark as parameter **in a loop**
6. Test return value and if TRUE then increment Count **in a loop**
7. ... write ID to file ...
8. ... after conversion to a string **in a loop**
9. Final output of message giving number of papers to check **not in a loop**

Max 8 marks

(8 marks)

3 The requirements have been changed. Conceal() will now be written as a procedure which will process 100 card numbers each time it is called.

The card numbers will be stored in a 2D array CardNumber. The original string will be stored in column one and the modified string in column two.

Write pseudocode to declare the array.

Answer

Mark Scheme and Guidance

DECLARE CardNumber : ARRAY [1:100, 1:2] OF STRING

MP1 Correct dimensions
MP2 All other parts of the statement correct

(2 marks)

4 The implementation of a linked list uses an integer variable and a 1D array List of type Node.

Record type Node is declared in pseudocode as follows:

```
TYPE Node
DECLARE Data : STRING
DECLARE Pointer : INTEGER
ENDTYPE
```

The array List is declared in pseudocode as follows:

```
DECLARE List : ARRAY[1:200] OF Node
```

The linked list will operate as follows:

- Integer variable HeadPointer will store the array index for the first node in the linked list.
- The Pointer field of a node contains the index value of the next array element (the next node) in the linked list.
- The value 0 is used as a null pointer.

Give the range of valid values that could be assigned to variable HeadPointer.

Answer

Mark Scheme and Guidance

0 to 200

(1 mark)

5 An alternative algorithm to determine if a paper needs to be checked uses a global 1D array Check, containing 76 elements of type BOOLEAN. The indices of the array are from 0 to 75 (inclusive), corresponding to the range of possible marks.

An element value in Check is TRUE if the index is within 2 marks of a grade boundary. For example, in the case where the C grade boundary is 43 the corresponding part of the Check array would be as follows:

Index	Value	
40	FALSE	
41	TRUE	
42	TRUE	
43	TRUE	← The grade boundary for a C grade
44	TRUE	
45	TRUE	
46	FALSE	

The mark for a given paper is stored in variable Mark.

Describe how an algorithm would use the Check array to determine whether this paper should be checked.

Answer

Mark Scheme and Guidance

MP1 Use Mark as the index to the Check array // to specify an array element

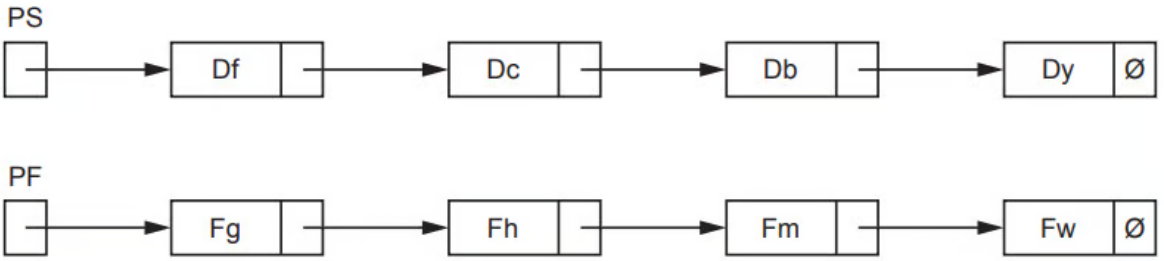
MP2 If value of indexed element is TRUE, then the paper will need to be checked

(2 marks)

6 The diagram shows an Abstract Data Type (ADT) representation of a linked list after data items have been added.

- PS is the start pointer.
- PF is the free list pointer.
- Labels Df, Dc, Db and Dy represent the data items of nodes in the list.

- Labels Fg, Fh, Fm and Fw represent the data items of nodes in the free list.
- The symbol Ø represents a null pointer.



A program will be written to include a linked list to store alphanumeric user IDs.

The design uses two variables and two 1D arrays to implement the linked list.
Each array element contains data of a single data type and **not** a record.

The statements below describe the design.

Complete the statements.

The two variables will be of type ..... .

The two variables will be used as ..... to the arrays.

The values stored in the two variables will indicate ..... .

The first 1D array will be of type ..... .

The first 1D array will be used to ..... .

The second 1D array will be of type ..... .

The second 1D array will be used to ..... .

Answer

Mark Scheme and Guidance

Max 2 marks for 'Variables':

The two variables will be of type **Integer**
The two variables will be used as **pointers / indexes** to the arrays.
The values stored in the two variables will indicate **the first element in each list**
The first 1D array will be of type **String**
The first 1D array will be used to **store the values // data items // User IDs**
The second 1D array will be of type **Integer**
The second 1D array will be used to **store the pointers // point to next item**

Mark as follows:

One mark for **each** of the first three rows
One mark for **both** Array 1 rows
One mark for **both** Array 2 rows

(5 marks)

7 A factory needs a program to help manage its production of items.

Data will be stored about each item.

The data for each item will be held in a record structure of type Component.

The programmer has started to define the fields that will be needed as shown in the table.

Field	Example value	Comment
Item_Num	123478	a numeric value used as an array index
Reject	FALSE	TRUE if this item has been rejected
Stage	'B'	a letter to indicate the stage of production
Limit_1	13.5	any value in the range 0 to 100 inclusive
Limit_2	26.4	any value in the range 0 to 100 inclusive

A 1D array Item of 2000 elements will store the data for all items.

Write pseudocode to declare the Item array.

Answer

Mark Scheme and Guidance

DECLARE Item : ARRAY [1:2000] OF Component//
DECLARE Item : ARRAY [2000] OF Component//
DECLARE Item : ARRAY [0:1999] OF Component

One mark per underlined phrase

(2 marks)

8 The value zero denotes the split between the two parts of the sequence.

The requirement changes and now there may be up to 20 parts.

(i) Identify a suitable data structure that could be used to store the different total values.

[2]

(ii) Describe **three** benefits of using the data structure given in part (b)(i).

[3]

Answer

Mark Scheme and Guidance

(i)

(1D) array of 20 reals

Marks as follows:
1 mark for array
1 mark for 20 reals

(ii)

One mark per point:

1. (Multiple instances referenced via a single identifier so) **fewer identifiers needed**
2. Easier to process / search / organise / access the data // Values may be accessed via a loop-controlled variable /an index //An array / data can be iterated through
3. Makes the program/algorithm easier to write / design / understand / maintain / modify // Simplifies the program // Easier to debug / test

(5 marks)

9 A global array is declared in pseudocode as follows:

DECLARE Data : ARRAY[1:150] OF STRING

A function TooMany() will:

1. take two parameters:

- a string (the search string)
- an integer (the maximum value)

2. count the number of strings in the array that exactly match the search string

3. return TRUE if the count is greater than the maximum value, otherwise will return FALSE

The global array is changed to a 2D array, organised as 150 rows by 2 columns. It is declared in pseudocode as follows:

DECLARE Data : ARRAY[1:150, 1:2] OF STRING

The algorithm for the function in **part (a)** is changed. Strings will only be counted if **both** of the following conditions are true:

- The current row is an even number.
- The search string exactly matches the value in **either** column.

Write pseudocode to check these conditions.

Assume that the row index is contained in variable Row and the search string in variable Search.

Answer

Mark Scheme and Guidance

MP1 Test for row being even number
MP2 Test for either column value equal to Search

```
IF Row MOD 2 = 0 AND __
(Data[Row, 1] = Search OR Data[Row, 2] = Search) THEN
```

ALTERNATIVE using nested IFs:

```
IF Row MOD 2 = 0 THEN
IF Data[Row, 1] = Search OR Data[Row, 2] = Search THEN
```

MP3 Selection structure is either:

- Single IF statement using AND, or
- Two nested IFs using AND, or
- Single IF and the use of a two-iteration loop

Either of these structures correctly formed scores the mark

ALTERNATIVE SOLUTION: A FOR loop using 'STEP 2'

```
FOR Row ← 2 TO 150 / NEXT Row STEP 2
Data[Row, 1] = Search OR Data[Row, 2] = Search) THEN
```

(3 marks)

10 A global 1D array of integers contains four elements, which are assigned values as shown:

```
Mix[1] ← 1
Mix[2] ← 3
Mix[3] ← 4
Mix[4] ← 2
```

A procedure Process() manipulates the values in the array.

The procedure is written in pseudocode:

```
PROCEDURE Process(Start : INTEGER)
DECLARE Value, Index, Count : INTEGER

Index ← Start
Count ← 0

REPEAT
Value ← Mix[Index]
Mix[Index] ← Mix[Index] - 1
Index ← Value
Count ← Count + 1
UNTIL Count = 5
```

```
    Mix[4] ← Count * Index

ENDPROCEDURE
```

Complete the trace table on the opposite page by dry running the procedure when it is called as follows:

```
CALL Process(3)
```

Index	Value	Count	Mix[1]	Mix[2]	Mix[3]	Mix[4]

Answer

Mark Scheme and Guidance

Index	Value	Count	Mix[1]	Mix[2]	Mix[3]	Mix[4]
3		0	1	3	4	2
	4				3	
4		1				
	2					1
2		2				
	3			2		
3		3				
	3				2	
3		4				
	2				1	
2		5				
						10

One mark for:

- Initialisation row
- Each iteration of Count 1 to 4 with correct Index, Count and array Mix as shown
- Final iteration, Count = 5, correct Index, Count and array Mix as shown plus Mix[4] assignment

(6 marks)

11 A class of students are developing a program to send data between computers. Many computers are connected together to form a wired network. Serial ports are used to connect one computer to another.

Each computer:

- is assigned a unique three-digit ID
- has three ports, each identified by an integer value
- is connected to between one and three other computers.

Data is sent as individual message strings.
Each string contains the destination ID (the ID of the computer that is to receive the message) followed by the data:

<DestinationID ><Data>

Messages may pass through several computers on the way to their destination.
When a message arrives at a computer, that is **not** the destination, the program needs to forward it on to another computer using one of its serial ports.

The port to use is obtained from information that is stored in an array **RouteTable.**

RouteTable is a global 2D array of integers. It is declared in pseudocode as follows:

DECLARE RouteTable : ARRAY[1:6,1:3] OF INTEGER

The values in the first two columns of RouteTable define a range of ID values.
Column 3 gives the corresponding port number to use when forwarding the message to a computer with an ID within this range.

For example, the contents of RouteTable could be:

	Column 1	Column 2	Column 3
Row 1	100	199	1
Row 2	200	259	2
Row 3	-1	<undefined>	<undefined>
Row 4	260	399	2
Row 5	400	599	3
Row 6	600	999	1

In this example, a message that arrives with a DestinationID of "283" will be forwarded using port 2.

Row 3 in the example shows an unused row. These may occur anywhere. Unused rows have the column 1 element set to −1. The value of unused elements in the other two columns is undefined.

The programmer has defined the first program module as follows:

Module	Description
GetPort()	• takes a DestinationID as a parameter of type string • searches for the range corresponding to the DestinationID in the array • returns the port number, or returns −1 if no corresponding range is found

Write pseudocode for module GetPort().

Assume DestinationID contains a valid three-digit string.

Answer

Mark Scheme and Guidance

Solution using a loop

Example solution – using a loop

```
FUNCTION GetPort(ThisDest : STRING) RETURNS INTEGER
DECLARE Index, DNum, Port : INTEGER

DNum ← STR_TO_NUM(ThisDest)
Index ← 1
Port ← -1

REPEAT
IF RouteTable[Index, 1] <> -1 THEN
IF DNum >= RouteTable[Index, 1] AND __
DNum <= RouteTable[Index, 2] THEN
Port ← RouteTable[Index, 3]
ENDIF
ENDIF
Index ← Index + 1
UNTIL Index = 7 OR Port <> -1 // end of array or range found
```

```
RETURN Port
ENDFUNCTION
```

Mark as follows **Max 7:**

1. Function heading and ending **including** parameter and return type
2. Convert **parameter** to a number
3. (Conditional) loop through array
4. Skip unused element **in a loop**
5. Attempt to check one range with destination **in a loop**
6. Test **all** ranges correctly with destination **in a loop**
7. Store port value if destination matched **in a loop** (and exit loop)
8. Return port value including -1 if no match found

Solution using selection statement(s)

Example solution

```
FUNCTION GetPort(ThisDest : STRING) RETURNS INTEGER
DECLARE Index, DNum, Port : INTEGER DNum ← STR_TO_NUM(ThisDest)
Port ← -1

IF RouteTable[1, 1] <> -1 AND RouteTable[1, 1] >= DNum AND RouteTable[1, 2] <=
DNum THEN
Port ← RouteTable[1, 3]
END IF
IF RouteTable[2, 1] <> -1 AND RouteTable[2, 1] >= DNum AND RouteTable[2, 2] <=
DNum THEN
Port ← RouteTable[2, 3]
END IF
IF RouteTable[3, 1] <> -1 AND RouteTable[3, 1] >= DNum AND RouteTable[3, 2] <=
DNum THEN
Port ← RouteTable[3, 3]
END IF
IF RouteTable[4, 1] <> -1 AND RouteTable[4, 1] >= DNum AND RouteTable[4, 2] <=
DNum THEN
Port ← RouteTable[4, 3]
END IF
IF RouteTable[5, 1] <> -1 AND RouteTable[5, 1] >= DNum AND RouteTable[5, 2] <=
DNum THEN
Port ← RouteTable[5, 3]
END IF
IF RouteTable[6, 1] <> -1 AND RouteTable[6, 1] >= DNum AND __ RouteTable[6, 2] <=
DNum THEN
```

```
Port ← RouteTable[6, 3]
END IF
RETURN Port
ENDFUNCTION
```

Mark as follows **Max 7:**

1. Function heading and ending **including** parameter and return type
2. Convert **parameter** to a number
3. Skip **all** unused elements
4. Attempt to check one range
5. Check two ranges with destination correctly
6. Check all ranges with destination correctly
7. Store port value if destination matched
8. 8 Return port value including -1 if no match found

(7 marks)

12 (a) Data is a 1D array of integers, containing 30 elements. All element values are unique.

The 30 data values could have been stored in separate variables rather than in an array.

Explain the benefits of using an array when designing a solution to part **(a)**.

Answer

Mark Scheme and Guidance

MP1 Simplifies the algorithm // easier to write / understand / test / debug
MP2 It is possible to iterate through the values // can use a loop // allows the storage of many values using a single identifier

(2 marks)

(b) The requirement changes. Array Data needs to hold 120 elements and each value may include a decimal place.

Write a pseudocode statement to declare the modified array.

Answer

Mark Scheme and Guidance

One mark per underlined section

DECLARE Data : ARRAY[1:120] OF REAL

(2 marks)

13 Procedure RandList() is modified so that the random numbers are also written into a 1D array Result.

A new module is written to confirm that the numbers in the array are in ascending order.

This module contains an IF statement that performs a comparison between elements:

```
IF (Result[x + 1] = Result[x]) OR (Result[x] > Result[x + 1]) THEN
Sequence ← FALSE
ENDIF
```

Write a simplified version of the conditional clause.

Answer

Mark Scheme and Guidance

One mark for simplified logical expression:

MP1 Result[x + 1] <= Result[x]

ALTERNATIVE SOLUTION:

Result[x] >= Result[x + 1]

(1 mark)