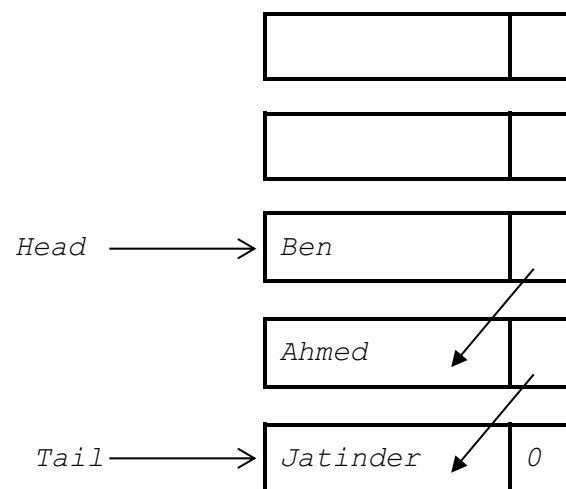


QUESTION 7.



6 (a)



1 mark for Head and Tail pointers
1 mark for 3 correct items – linked as shown
1 mark for correct order with null pointer in last nod

[3]



(b) (i)

		Queue	
HeadPointer		Name	Pointer
0	[1]		
	[2]		
	[3]		
TailPointer	[4]		5
	[5]		6
	[6]		7
FreePointer	[7]		8
	[8]		9
	[9]		10
	[10]		0

Mark as follows:

HeadPointer = 0 & TailPointer = 0
FreePointer assigned a value
Pointers[1] to [9] links the nodes together
Pointer[10] = 'Null'

[4]

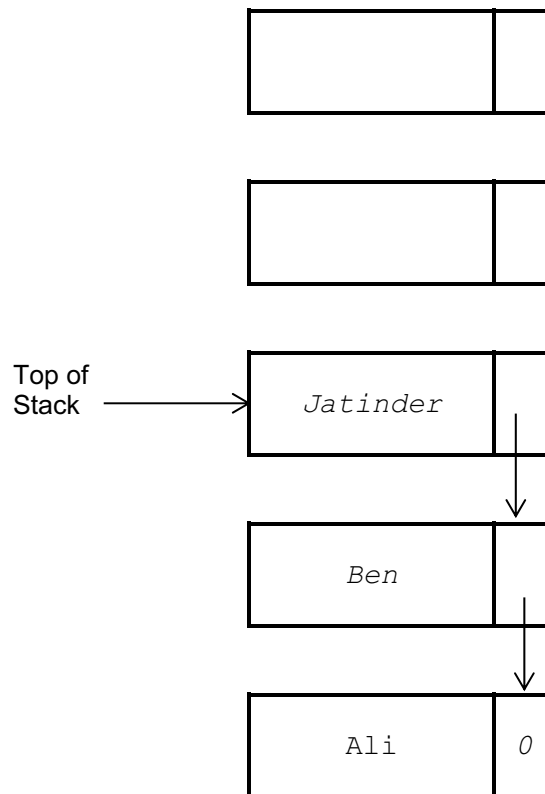
(ii) **PROCEDURE RemoveName()**
// Report error if Queue is empty
IF HeadPointer = 0
THEN
Error
ELSE
OUTPUT Queue[HeadPointer].Name
// current node is head of queue
CurrentPointer ← HeadPointer
// update head pointer
HeadPointer ← Queue[CurrentPointer].Pointer
//if only one element in queue, then update tail pointer
IF HeadPointer = 0
THEN
TailPointer ← 0
ENDIF
// link released node to free list
Queue[CurrentPointer].Pointer ← FreePointer
FreePointer ← CurrentPointer
ENDIF
ENDPROCEDURE

[max 6]

QUESTION 8.



5 (a)



1 mark for Top of Stack pointer

1 mark for 3 correct items

1 mark for correct order with null pointer in last node

[3]



(b) (i)

Stack

TopOfStackPointer		Name		Pointer
0		[1]		2
		[2]		3
		[3]		4
		[4]		5
		[5]		6
		[6]		7
		[7]		8
		[8]		9
		[9]		10
		[10]		0
FreePointer				
1				

Mark as follows:
 TopOfStackPointer
 FreePointer
 Pointers[1] to [9]
 Pointer[10]

[4]

Page 9	Mark Scheme	
	Cambridge International A Level – May/June 2015	



```

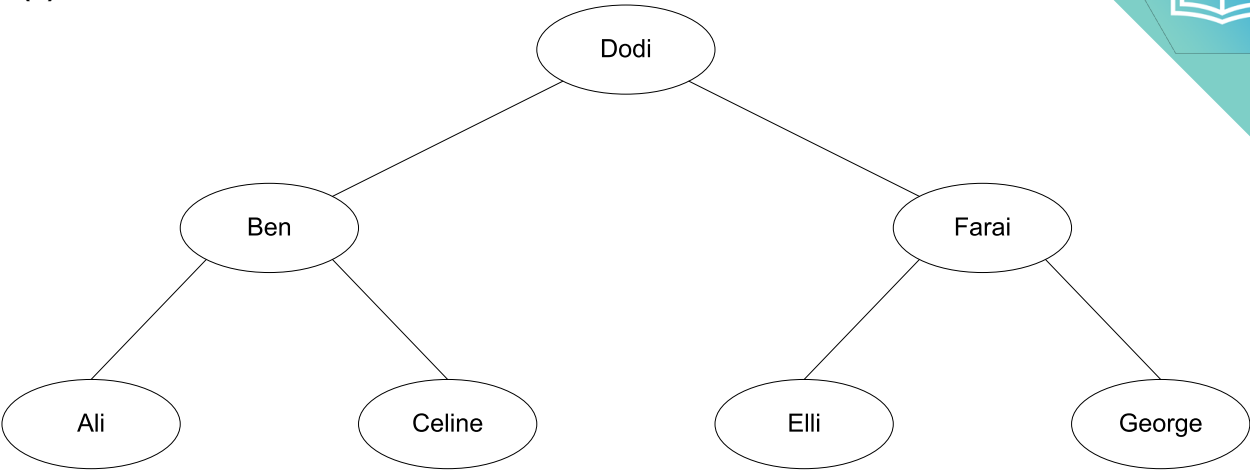
(ii) PROCEDURE Pop()
    // Report error if Stack is empty
    {
        IF TopOfStackPointer = 0
        THEN
            Error
        ELSE
            OUTPUT Stack[TopOfStackPointer].Name
            // take a copy of the current top of stack pointer
            TempPointer ← TopOfStackPointer
            // update the top of stack pointer
            TopOfStackPointer ← Stack[TempPointer].Pointer
            // link released node to free list
            Stack[TempPointer].Pointer ← FreePointer
            FreePointer ← TempPointer
        ENDIF
    }
ENDPROCEDURE

```

QUESTION 9.



4 (a)



[4]

(b)

Tree			
RootPointer		Name	LeftPointer RightPointer
1	[1]	Dodi	5 2
	[2]	Farai	3 4
	[3]	Elli	0 0
FreePointer	[4]	George	0 0
8	[5]	Ben	7 6
	[6]	Celine	0 0
	[7]	Ali	0 0
	[8]		9 0
	[9]		10 0
	[10]		0 0

[7]

Page 9	Mark Scheme	
	Cambridge International A Level – October/November 2015	



(c) (i) 01 PROCEDURE TraverseTree (BYVALUE Root : INTEGER)
02 IF Tree[Root].LeftPointer < > 0
03 THEN
04 TraverseTree (Tree[Root].LeftPointer)
05 ENDIF
06 OUTPUT Tree[Root].Name
07 IF Tree[Root].RightPointer < > 0
08 THEN
09 TraverseTree (Tree[Root].RightPointer)
10 ENDIF
11 ENDPROCEDURE

[5]

(ii) A procedure that calls itself // is defined in terms of itself
Line number: 04/09

[2]

QUESTION 10.



1	(a) (i)	TYPE LinkedList	1	
		(DECLARE) Surname : STRING	1	
		(DECLARE) Ptr : INTEGER }		
		ENDTYPE	1	
		Accept: LinkedList : RECORD	1	
		Surname : STRING	1	
		Ptr : INTEGER }		
		ENDRECORD	1	
		Accept: TYPE LinkedList = RECORD	1	
		Surname : STRING	1	
		Ptr : INTEGER }		
		ENDTYPE / ENDRECORD	1	
		Accept: STRUCTURE LinkedList	1	
		(DECLARE) Surname : STRING	1	
		(DECLARE) Ptr : INTEGER }		
		ENDSTRUCTURE	1	
		Accept AS / OF instead of :		
	(ii)	(DECLARE) <u>SurnameList[1:5000]</u> : <u>LinkedList</u>		2
		Accept AS / OF instead of : Accept () instead of [] Accept without lower bound Index separator can be , : ...		
	(b) (i)	Wu Accept with quotes		1
	(ii)	6		1
	(c) (i)	IsFound + relevant description BOOLEAN	1 1	2



Question	Answer
(ii)	Accept () instead of [] <pre> 01 Current ← <u>StartPtr</u> 02 IF Current = 0 03 THEN 04 OUTPUT "<u>Empty List</u>" (or similar message) (accept without quotes) Reject "Error" 05 ELSE 06 IsFound ← <u>FALSE</u> 07 INPUT ThisSurname 08 REPEAT 09 IF <u>SurnameList[Current].Surname</u> = ThisSurname 10 THEN 11 IsFound ← TRUE 12 OUTPUT "Surname found at position ", Current 13 ELSE 14 // move to the next list item 15 <u>Current ← SurnameList[Current].Ptr</u> 16 ENDIF 17 UNTIL IsFound = TRUE OR <u>Current = 0</u> 18 IF IsFound = FALSE 19 THEN 20 OUTPUT "Not Found" 21 ENDIF 22 ENDIF </pre>

QUESTION 11.



Question	Answer	Marks
4(a)(i)	containment/aggregation	1
4(a)(ii)	<pre>classDiagram class LinkedList class Node LinkedList "1" o-- "0..*" Node</pre> 1 mark for the two classes (in boxes) and connection with correct end point 1 mark for 0 ..* 0	Max 2



Question	Answer	
4(b)	mark as follows: • Class heading and ending • Constructor heading and ending • Parameters in constructor heading • Declaration of (private) attributes : Pointer, Data • Assignment of parameters to Pointer and Data Python Example <pre> class Node: def __init__(self, D, P): self.__Data = D self.__Pointer = P return </pre> Example Pascal <pre> type Node = class private Data : String; Pointer : Integer; public constructor Create(D : string; P : integer); procedure SetPointer(P : Integer); procedure SetData(D : String); function GetData() : String; function GetPointer() : Integer; end; constructor Node.Create(D : string; P : integer); begin Data := D; Pointer := P; end; </pre> Example VB.NET <pre> Class Node Private Data As String Private Pointer As Integer Public Sub New(ByVal D As String, ByVal P As Integer) Data = D Pointer = P End Sub End Class </pre>	1 1 + 1 1 1 1 1 ignore 1+1 1 1 1 1+1 1
4(c)(i)	A pointer that doesn't point to any data/node/address	1



Question	Answer
4(c)(ii)	-1 (accept NULL) The array only goes from 0 to 7 // the value is not an array index
4(c)(iii)	mark as follows: • Class and constructor heading and ending • Declare private attributes (HeadPointer, FreeListPointer, NodeArray) • Initialise HeadPointer to null • Initialise FreeListPointer to 0 • Looping 8 times ... • Creating empty node in NodeArray • Use .SetPointer method to point each new node to next node • Set last node pointer to null pointer Python Example <pre> class LinkedList: def __init__(self): self.__HeadPointer = - 1 self.__FreeListPointer = 0 self.__NodeArray = [] for i in range(8): ThisNode = Node("", (i + 1)) self.__NodeArray.append(ThisNode) self.__NodeArray[7].SetPointer(- 1) </pre> Example Pascal <pre> type LinkedList = class private HeadPointer : Integer; FreeList : Integer; NodeArray : Array[0..7] of Node; public constructor Create(); procedure FindInsertionPoint(NewData : string; var PreviousPointer, NextPointer : integer); procedure AddToList(NewData : string); procedure OutputListToConsole(); end; constructor LinkedList.Create(); var i : integer; begin HeadPointer := -1; FreeList := 0; for i := 0 To 7 do NodeArray[i] := Node.Create('', (i + 1)); NodeArray[7].SetPointer(-1); end; </pre>



Question	Answer	
	Example VB.NET <pre> Class LinkedList Private HeadPointer As Integer Private FreeList As Integer Private NodeArray(7) As Node } Public Sub New() HeadPointer = -1 FreeList = 0 For i = 0 To 7 NodeArray(i) = New Node("", (i + 1)) Next NodeArray(7).SetPointer(-1) End Sub End Class </pre>	1 1 1 1 1 1 1
4(c)(iv)	• Creating instance of LinkedList assigned to contacts Python Example <pre>contacts = LinkedList()</pre> Pascal Example <pre>var contacts : LinkedList; contacts := LinkedList.Create;</pre> VB.NET Example <pre>Dim contacts As New LinkedList</pre>	1



Question	Answer
4(c)(v)	mark as follows: • Start with HeadPointer • Output node data • Loop until null pointer • Following pointer to next node • Use of getter (ie GetData/GetPointer) Python Example <pre>def OutputListToConsole(self) : Pointer = self.__HeadPointer while Pointer != -1 : print(self.__NodeArray[Pointer].GetData()) Pointer = self.__NodeArray[Pointer].GetPointer() print() return</pre> Pascal Example <pre>procedure LinkedList.OutputListToConsole(); var Pointer : integer; begin Pointer := HeadPointer; while Pointer <> -1 do begin WriteLn(NodeArray[Pointer].GetData); Pointer := NodeArray[Pointer].GetPointer; end; end;</pre> VB.NET Example <pre>Public Sub OutputListToConsole() Dim Pointer As Integer Pointer = HeadPointer Do While Pointer <> -1 Console.WriteLine(NodeArray(Pointer).GetData) Pointer = NodeArray(Pointer).GetPointer Loop End Sub</pre>



Question	Answer
4(c)(vi)	mark as follows: • Store free list pointer as NewNodePointer • Store new data item in free node • Adjust free pointer • F list is currently empty • Make the node the first node • Set pointer of this node to Null Pointer • Find insertion point • If previous pointer is Null pointer • Link this node to front of list • Link new node between Previous node and next node Python Example <pre>def AddToList(self, NewData): NewNodePointer = self.__FreeListPointer self.__NodeArray[NewNodePointer].SetData(NewData) self.__FreeListPointer = self.__NodeArray[self.__FreeListPointer].GetPointer() if self.__HeadPointer == -1: self.__HeadPointer = NewNodePointer self.__NodeArray[NewNodePointer].SetPointer(-1) else: PreviousPointer, NextPointer = self.FindInsertionPoint(NewData) if PreviousPointer == -1 : self.__NodeArray[NewNodePointer].SetPointer (self.__HeadPointer) self.__HeadPointer = NewNodePointer else: self.__NodeArray[NewNodePointer].SetPointer(NextPointer) self.__NodeArray[PreviousPointer].SetPointer(NewNodePointer)</pre>



Question	Answer
	Pascal Example <pre> procedure LinkedList.AddToList(NewData : string); var NewNodePointer , PreviousPointer, NextPointer : integer; begin // make a copy of free list pointer NewNodePointer := FreeListPointer; // store new data item in free node NodeArray[NewNodePointer].SetData(NewData); // adjust free pointer FreeListPointer := NodeArray[FreeListPointer].GetPointer; // if list is currently empty if HeadPointer = -1 then // make the node the first node begin HeadPointer := NewNodePointer; // set pointer to Null pointer NodeArray[NewNodePointer].SetPointer(-1); end else // find insertion point begin FindInsertionPoint(NewData, PreviousPointer, NextPointer); // if previous pointer is Null pointer if PreviousPointer = -1 then // link node to front of list begin NodeArray[NewNodePointer] .SetPointer(HeadPointer); HeadPointer := NewNodePointer ; end else // link new node between Previous node and next node begin NodeArray[NewNodePointer] .SetPointer(NextPointer); NodeArray[PreviousPointer] .SetPointer(NewNodePointer); end; end; end; end; end; </pre>



Question	Answer
	VB.NET Example <pre> Public Sub AddToList(ByVal NewData As String) Dim NewNodePointer, PreviousPointer, NextPointer As Integer ' make copy of free list pointer NewNodePointer= FreeListPointer ' store new data item in free node NodeArray(NewNodePointer).SetData(NewData) ' adjust free pointer FreeListPointer = NodeArray(FreeListPointer).GetPointer ' if list is currently empty If HeadPointer = -1 Then ' make the node the first node HeadPointer = NewNodePointer ' set pointer to Null pointer NodeArray(NewNodePointer).SetPointer(-1) Else ' find insertion point FindInsertionPoint(NewData, PreviousPointer, NextPointer) ' if previous pointer is Null pointer If PreviousPointer = -1 Then ' link to front of list NodeArray(NewNodePointer).SetPointer(HeadPointer) HeadPointer = NewNodePointer Else ' link new node between Previous node and next node NodeArray(NewNodePointer).SetPointer(NextPointer) NodeArray(PreviousPointer).SetPointer(NewNodePointer) End If End If End Sub </pre>



Question	Answer
	Pseudocode for reference: <pre> PROCEDURE AddToList(NewData) // remember value of free list pointer NewNodePointer ← FreeListPointer // add new data item to free node pointed to by free list NodeArray[NewNodePointer].Data ← NewData // adjust free pointer to point to next free node FreeListPointer ← NodeArray[FreeList].Pointer // is list currently empty? IF HeadPointer = NullPointer THEN // make the node the first node HeadPointer ← NewNodePointer // set pointer of new node to Null pointer NodeArray[NewNodePointer].Pointer ← NullPointer ELSE // find insertion point CALL FindInsertionPoint(NewData, PreviousPPointer, NextPointer) // if previous pointer is Null pointer IF PreviousPointer = NullPointer THEN // link new node to front of list NodeArray[NewNodePointer].Pointer ← HeadPointer HeadPointer ← NewNodePointer ELSE // link new node between previous node and next node NodeArray[NewNodePointer].Pointer ← NextPointer NodeArray[PreviousPointer].Pointer ← NewNodePointer END IF ENDIF END PROCEDURE </pre>

9608/42
QUESTION 12.

Question	Answer	Marks																																	
2(a)	A pointer that doesn't point to another node/other data/address // indicates the end of the branch	1																																	
2(b)	one mark per bullet - node with 'Athens' linked to left pointer of Berlin (ignore null pointer) - null pointers in left and right pointers of Athens	2																																	
2(c)(i)	RootPointer 0 [0] [1] [2] [3] [4] [5] [6] [7] [8] [9] <table border="1" style="border-collapse: collapse; text-align: center;"> <thead> <tr> <th>LeftPointer</th><th>Tree Data</th><th>RightPointer</th></tr> </thead> <tbody> <tr><td>2</td><td>Dublin</td><td>1</td></tr> <tr><td>-1/∅</td><td>London</td><td>3</td></tr> <tr><td>6</td><td>Berlin</td><td>5</td></tr> <tr><td>4</td><td>Paris</td><td>-1/∅</td></tr> <tr><td>-1/∅</td><td>Madrid</td><td>-1/∅</td></tr> <tr><td>-1/∅</td><td>Copenhagen</td><td>-1/∅</td></tr> <tr><td>-1/∅</td><td>Athens</td><td>-1/∅</td></tr> <tr><td>8</td><td></td><td>-1/∅</td></tr> <tr><td>9</td><td></td><td>-1/∅</td></tr> <tr><td>-1/∅</td><td></td><td>-1/∅</td></tr> </tbody> </table> FreePointer 7 1 mark	LeftPointer	Tree Data	RightPointer	2	Dublin	1	-1/∅	London	3	6	Berlin	5	4	Paris	-1/∅	-1/∅	Madrid	-1/∅	-1/∅	Copenhagen	-1/∅	-1/∅	Athens	-1/∅	8		-1/∅	9		-1/∅	-1/∅		-1/∅	5
LeftPointer	Tree Data	RightPointer																																	
2	Dublin	1																																	
-1/∅	London	3																																	
6	Berlin	5																																	
4	Paris	-1/∅																																	
-1/∅	Madrid	-1/∅																																	
-1/∅	Copenhagen	-1/∅																																	
-1/∅	Athens	-1/∅																																	
8		-1/∅																																	
9		-1/∅																																	
-1/∅		-1/∅																																	
2(c)(ii)	-1 - It is not the number for any node.	2																																	

Question	Answer	Marks
2(d)(i)	<pre> TYPE Node LeftPointer : INTEGER RightPointer : INTEGER Data : STRING ENDTYPE DECLARE Tree : ARRAY[0 : 9] OF Node DECLARE FreePointer : INTEGER DECLARE RootPointer : INTEGER PROCEDURE CreateTree() DECLARE Index : INTEGER RootPointer ← -1 FreePointer ← 0 FOR Index ← 0 TO 9 // link nodes Tree[Index].LeftPointer ← Index + 1 Tree[Index].RightPointer ← -1 ENDFOR Tree[9].LeftPointer ← -1 ENDPROCEDURE </pre>	7 1 1 1 1 1 1

Question	Answer	Marks
2(d)(ii)	<pre> PROCEDURE AddToTree (ByVal NewDataItem : STRING) // if no free node report an error IF FreePointer = -1 THEN ERROR("No free space left") ELSE // add new data item to first node in the free list NewNodePointer ← FreePointer Tree[NewNodePointer].Data ← NewDataItem // adjust free pointer FreePointer ← Tree[FreePointer].LeftPointer // clear left pointer Tree[NewNodePointer].LeftPointer ← -1 // is tree currently empty ? IF RootPointer = -1 THEN // make new node the root node RootPointer ← NewNodePointer ELSE // find position where new node is to be added Index ← RootPointer CALL FindInsertionPoint(NewDataItem, Index, Direction) </pre>	8 1 1 1 1 1 1

Question	Answer	Marks
	<pre> IF Direction = "Left" THEN // add new node on left Tree[Index].LeftPointer ← NewNodePointer 1 ELSE // add new node on right Tree[Index].RightPointer ← NewNodePointer 1 ENDIF ENDIF ENDIF ENDPROCEDURE </pre>	
2(e)	1 mark per bullet • test for base case (null/-1) • recursive call for left pointer • output data • recursive call for right pointer • order, visit left, output, visit right <pre> IF Pointer <> NULL 1 THEN TraverseTree(Tree[Pointer].LeftPointer) 1 OUTPUT Tree[Pointer].Data 1 + 1 TraverseTree(Tree[Pointer].RightPointer) 1 ENDIF ENDPROCEDURE </pre>	5

Question	Answer	Marks
2(a)	1 mark per bullet point to max 4: • declaration of type Book • Title, Author and ISBN as String • Fiction as Boolean • LastRead as Date For example: <pre> TYPE Book DECLARE Title : String DECLARE Author : String DECLARE ISBN : String DECLARE Fiction : Boolean DECLARE LastRead : Date ENDTYPE </pre>	4

PUBLISHED

Question	Answer	Marks
2(b)	1 mark per bullet point to max 4: • Function header • ... taking ISBN as parameter • Converting ISBN to integer • Calculating Hash (ISBN mod 2000 + 1) • Returning the calculated Hash Examples: Python: <pre>def Hash(ISBN): ISBNint = int(ISBN) Hash = (ISBNint % 2000) + 1</pre> VB.NET: <pre>Function Hash (ISBN As String) As Integer ISBNint = convert.ToInt32(ISBN) Hash = (ISBNint MOD 2000) + 1 End Function</pre> Pascal: <pre>function Hash(ISBN : String) : Integer begin ISBNint = StrToInt(ISBN) Hash = (ISBNint MOD 2000) + 1 end;</pre>	4

PUBLISHED

Question	Answer	Marks
2(c)	1 mark per bullet point to max 8: • Procedure FindBook declaration and prompt and input ISBN • Validate data input has 13 characters • ... and are all numeric • ..loop until valid • Call Hash() with input data and store return data • Open MyBooks.dat for reading as random file and close • Finding the record using return value Hash() • Get the data for the record • ...store in variable of type Book • ...output all the data for the record	8

Question	Answer	Marks
2(c)	Example: <pre> PROCEDURE FindBook() DECLARE BookInfo : Book REPEAT ISBN ← input("Enter the ISBN number") Valid ← TRUE Size ← LENGTH(ISBN) IF size <> 13 THEN Valid ← FALSE ELSE FOR i ← 1 to 13 IF NOT(MID(ISBN,i,1) >= '0' AND MID(ISBN,i,1)<= '9') THEN Valid ← FALSE ENDIF ENDFOR ENDIF UNTIL Valid Filename ← "myBooks.dat" OPENFILE Filename FOR RANDOM RecordLocation ← Hash(ISBN) SEEK FileName, RecordLocation GETRECORD Filename, BookInfo CLOSEFILE Filename OUTPUT (BookInfo.Title & " " & BookInfo.Author & " " & BookInfo.ISBN & " " & BookInfo.Fiction & " " & BookInfo.LastRead) ENDPROCEDURE </pre>	

Question	Answer	Marks
5(c)(ii)	1 mark per bullet point to max 2: Example: • If there is any delay to a task which is part of the critical path • ... then the overall project will be delayed • Gives the earliest possible completion time • ... this allows you to organise/allocate resources (efficiently) • Frequently recalculate the critical path ... • ... to see if there are any delays/new critical path has arisen • Can identify where there is slack in activities • ... so they can start later without affecting the critical path	2

Question	Answer	Marks
6(a)(i)	1 mark per bullet point: • Declaring a record type // class etc. • Declaring Country as a string • Declaring Pointer as an integer Example: <pre> TYPE ListElement DECLARE Country : STRING DECLARE Pointer : INTEGER ENDTYPE </pre>	3

Question	Answer	Marks
6(a)(ii)	1 mark per bullet point: • Declaring <code>CountryList</code> as an array with 15 elements • Of type <code>ListElement</code> Example: <pre>DECLARE CountryList : ARRAY[1 : 15] OF ListElement</pre>	2
6(b)	1 mark for each completed statement <pre>PROCEDURE DeleteNode(NodeValue: STRING, ThisPointer: INTEGER, PreviousPointer: INTEGER) IF CountryList[ThisPointer].Value = NodeValue THEN CountryList[ThisPointer].Value ← "" IF ListHead = ThisPointer THEN ListHead ← CountryList[ThisPointer].Pointer ELSE CountryList[PreviousPointer].Pointer ← CountryList[ThisPointer].Pointer ENDIF CountryList[LastNode].Pointer ← ThisPointer LastNode ← ThisPointer CountryList[ThisPointer].Pointer ← -1 ELSE IF CountryList[ThisPointer].Pointer <> -1 THEN CALL DeleteNode(NodeValue, CountryList[ThisPointer].Pointer, ThisPointer) ELSE OUTPUT "DOES NOT EXIST" ENDIF ENDIF ENDPROCEDURE</pre>	5

QUESTION 15.



1(b)	1 mark per bullet point to max 3 • (Linear) data structure • First in First out // FIFO // An item is added to the end of the queue and an item is removed from the front • All items are kept in the order they are entered • It has a head pointer and a tail pointer • Can be static or dynamic • A queue can be circular ... • ...when the (tail) pointer reaches the last position it returns to the first	3



Question	Answer
2(a)(i)	1 mark per bullet point • 95 to left of 97 • 109 to left of 121 • 135 to right of 125 • 149 to right of 135 • Null points in all places and no inappropriate pointers RootPointer <pre> graph TD RootPointer[] --> N99[99] N99 -- LeftPointer --> N97[97] N99 -- RightPointer --> N125[125] N97 --> N95[95] N125 --> N121[121] N125 --> N135[135] N121 --> N109[109] N135 --> N149[149] N95 --> Ø1[Ø] N97 --> Ø2[Ø] N121 --> Ø3[Ø] N125 --> Ø4[Ø] N135 --> Ø5[Ø] N149 --> Ø6[Ø] </pre>



Question	Answer																																								
2(a)(ii)	1 mark per bullet point • FreePointer as 8 • 99 • 125 • 121 and 97 • 109 and 95 • 135 and 149 <table><tr><td>RootPointer</td></tr><tr><td>0</td></tr></table> <table><tr><td>FreePointer</td></tr><tr><td>8</td></tr></table> <table><tr><th>Index</th><th>LeftPointer</th><th>Data</th><th>RightPointer</th></tr><tr><td>[0]</td><td>3</td><td>99</td><td>1</td></tr><tr><td>[1]</td><td>2</td><td>125</td><td>6</td></tr><tr><td>[2]</td><td>4</td><td>121</td><td>null</td></tr><tr><td>[3]</td><td>5</td><td>97</td><td>null</td></tr><tr><td>[4]</td><td>null</td><td>109</td><td>null</td></tr><tr><td>[5]</td><td>null</td><td>95</td><td>null</td></tr><tr><td>[6]</td><td>null</td><td>135</td><td>7</td></tr><tr><td>[7]</td><td>null</td><td>149</td><td>null</td></tr></table>	RootPointer	0	FreePointer	8	Index	LeftPointer	Data	RightPointer	[0]	3	99	1	[1]	2	125	6	[2]	4	121	null	[3]	5	97	null	[4]	null	109	null	[5]	null	95	null	[6]	null	135	7	[7]	null	149	null
RootPointer																																									
0																																									
FreePointer																																									
8																																									
Index	LeftPointer	Data	RightPointer																																						
[0]	3	99	1																																						
[1]	2	125	6																																						
[2]	4	121	null																																						
[3]	5	97	null																																						
[4]	null	109	null																																						
[5]	null	95	null																																						
[6]	null	135	7																																						
[7]	null	149	null																																						

9608/42 **QUESTION 16.**

Question	Answer	Marks								
2(a)	1 mark for correct tick <table><tr><th>Statement</th><th>Tick (✓)</th></tr><tr><td>Last in first out</td><td>✓</td></tr><tr><td>First in first out</td><td></td></tr><tr><td>Last in last out</td><td></td></tr></table>	Statement	Tick (✓)	Last in first out	✓	First in first out		Last in last out		1
Statement	Tick (✓)									
Last in first out	✓									
First in first out										
Last in last out										
2(b)(i)	1 mark for correct stack <table><tr><td></td></tr><tr><td></td></tr><tr><td></td></tr><tr><td></td></tr><tr><td></td></tr><tr><td>10</td></tr><tr><td>35</td></tr><tr><td>20</td></tr></table>						10	35	20	1
10										
35										
20										
2(b)(ii)	1 mark for correct stack <table><tr><td></td></tr><tr><td></td></tr><tr><td></td></tr><tr><td></td></tr><tr><td>65</td></tr><tr><td>50</td></tr><tr><td>35</td></tr><tr><td>20</td></tr></table>					65	50	35	20	1
65										
50										
35										
20										

Question	Answer	Marks
2(b)(iii)	1 mark for each bullet • Function Push ... • ...taking parameter (returning Boolean) • Checking if Top = 7 ... • ...returning FALSE if full • ...returning TRUE otherwise • if not full, increment Top • ... add parameter to Top of ArrayStack <pre> FUNCTION Push (BYVALUE DataItem : Integer) (RETURNS Boolean) IF Top = 7 THEN RETURN FALSE ELSE Top ← Top + 1 ArrayStack[Top] ← DataItem RETURN TRUE ENDIF ENDFUNCTION </pre>	7

Question	Answer	Marks
2(a)	1 mark per completed statement <pre> FUNCTION AddToQueue (Number : INTEGER) RETURNS BOOLEAN CONSTANT FirstIndex = 0 CONSTANT LastIndex = 7 TempPointer ← EndPointer + 1 IF TempPointer > LastIndex THEN TempPointer ← FirstIndex ENDIF IF TempPointer = StartPointer THEN RETURN FALSE ELSE EndPointer ← TempPointer NumberQueue[EndPointer] ← Number RETURN TRUE ENDIF ENDFUNCTION </pre>	5
2(b)	1 mark per bullet point 1 mark for: ... if the start pointer reaches the end of the queue, it becomes the index of the first element in the queue Max 3 from: - Checks if the circular queue is empty // Checks if the queue has any data in it ... if it is empty it reports that it is empty - If not empty, return the value at the position of the start pointer then increments the start pointer	4

PUBLISHED

Question	Answer	Marks
2(c)	1 mark per bullet point to max 3 e.g. • Stack • Linked list • Dictionary • (Binary) tree	3

Question	Answer	Marks						
3(a)	1 mark per test data type • Normal / valid • Abnormal / erroneous / invalid • Boundary / extreme	3						
3(b)(i)	1 mark for each name <table><tr><th>Description</th><th>Name of debugging feature</th></tr><tr><td>A point where the program can be halted to see if the program works to this point</td><td>Breakpoint</td></tr><tr><td>One statement is executed and then the program waits for input from the programmer to move to the next statement.</td><td>Stepping // step through/over/into</td></tr></table>	Description	Name of debugging feature	A point where the program can be halted to see if the program works to this point	Breakpoint	One statement is executed and then the program waits for input from the programmer to move to the next statement.	Stepping // step through/over/into	2
Description	Name of debugging feature							
A point where the program can be halted to see if the program works to this point	Breakpoint							
One statement is executed and then the program waits for input from the programmer to move to the next statement.	Stepping // step through/over/into							
3(b)(ii)	1 mark for name, 1 for description e.g. • variable watch window • observe how variables change during execution // view current status of variables • Error list • describes the error // gives line number of error	2						