0 1 **Figure 3** shows a partial solution to a logic puzzle. To complete the solution each of the letters A-I must appear exactly once in each row of nine cells, exactly once in each column of nine cells and exactly once in each of the collections of three-by-three cells shown with limits shown by a thicker border.

In **Figure 3** the logic puzzle has been completed except for the collection of three-by-three cells in the top-right corner.

Figure 3

D	F	G	B	A	C			
I	E	H	F	G	D			
A	B	C	I	H	E			
B	G	D	E	C	H	A	F	I
F	H	I	D	B	A	E	G	C
C	A	E	G	I	F	B	H	D
H	D	B	C	E	G	F	I	A
G	C	F	A	D	I	H	B	E
E	I	A	H	F	B	D	C	G

0 1 . 1 Complete the solution for the puzzle shown in **Figure 3**.

Copy the contents of the unshaded cells in **Figure 3** into the table in your Electronic Answer Document.

[1 mark]

Figure 4 shows a simpler example of this type of logic puzzle with fewer cells. In this simpler puzzle only the letters A-D are used.

Figure 4

C			B
		A	
	B		
			A

It is possible to represent this type of puzzle as a graph. To do this a unique number is given to each cell and a node containing this unique number is added to the graph. An edge between two nodes denotes a link between those two cells, meaning they cannot contain the same letter as each other.

Figure 5 shows how unique numbers have been allocated to each cell in the puzzle in **Figure 4** and **Figure 6** shows an adjacency matrix that represents this puzzle.

Figure 5

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15

Figure 6

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
0		1	1	1	1	1	0	0	1	0	0	0	1	0	0	0
1			1	1	1	1	0	0	0	1	0	0	0	1	0	0
2				1	0	0	1	1	0	0	1	0	0	0	1	0
3					0	0	1	1	0	0	0	1	0	0	0	1
4						1	1	1	1	0	0	0	1	0	0	0
5							1	1	0	1	0	0	0	1	0	0
6								1	0	0	1	0	0	0	1	0
7									0	0	0	1	0	0	0	1
8										1	1	1	1	1	0	0
9											1	1	1	1	0	0
10												1	0	0	1	1
11													0	0	1	1
12														1	1	1
13															1	1
14																1
15																

The graph in **Figure 6** can be considered to be both a representational abstraction and an abstraction by generalisation of the puzzle from **Figure 4**.

0 1 . 2 What is representational abstraction?

[1 mark]

0 1 . 3 What is abstraction by generalisation?

[1 mark]

0 1 . 4 Other than the contents of the cells, what information has been removed from the puzzle in **Figure 4** when it has been represented as a graph?

[1 mark]

A graph can be represented using an adjacency list or as an adjacency matrix.

0 1 . 5 Explain the circumstances when it would be more appropriate to use an adjacency matrix instead of an adjacency list.

[2 marks]

0 1 . 6 Only the top half of the matrix in **Figure 6** needed to be used to present the puzzle. For which type of graph would the bottom half of the matrix also need to be used?

[1 mark]

0 2 . 1 For each of the statements in **Table 2**, complete each row to indicate if the statement is true or false for Dijkstra's algorithm.

Table 2

	True or False?
Calculates the shortest path between a node and other nodes in a graph.	
Can be used to prove that the Halting Problem cannot be solved.	
Can be used with both directed and undirected graphs.	
Can be used with both weighted and unweighted graphs.	

Copy the contents of the unshaded cells in **Table 2** into the table in your Electronic Answer Document.

[2 marks]

Figure 3 shows a subroutine represented using pseudo-code. The subroutine makes use of an array `Visited` and an array `ConnectedNodes` that stores a graph represented as an adjacency list.

Figure 3

```
FUNCTION G(V, P)
  Visited[V] ← True
  FOR EACH N IN ConnectedNodes[V]
    IF Visited[N] = False THEN
      IF G(N, V) = True THEN
        RETURN True
      ENDIF
    ELSE IF N ≠ P THEN
      RETURN True
    ENDIF
  ENDFOR
  RETURN False
ENDFUNCTION
```

0 2 . 2 The subroutine `G` uses recursion.

Explain what is meant by a recursive subroutine.

[1 mark]

Figure 3 (repeated)

```
FUNCTION G(V, P)
  Visited[V] ← True
  FOR EACH N IN ConnectedNodes[V]
    IF Visited[N] = False THEN
      IF G(N, V) = True THEN
        RETURN True
      ENDIF
    ELSE IF N ≠ P THEN
      RETURN True
    ENDIF
  ENDFOR
  RETURN False
ENDFUNCTION
```

Figure 4 shows a subroutine represented using pseudo-code. The subroutine makes use of the array `Visited`.

Figure 4

```
FUNCTION F()
  FOR Count ← 0 TO LENGTH(Visited) - 1
    IF Visited[Count] = False THEN
      RETURN False
    ENDIF
  ENDFOR
  RETURN True
ENDFUNCTION
```

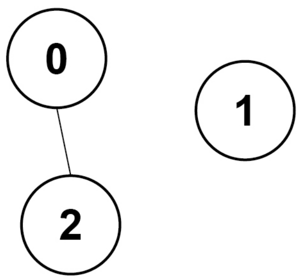
Figure 5 shows a subroutine represented using pseudo-code. The subroutine makes use of the subroutine `G` shown in **Figure 3**, the subroutine `F` shown in **Figure 4** and the array `Visited`.

Figure 5

```
FUNCTION E()
  Set all elements of Visited to False
  IF G(0, -1) = True THEN
    RETURN False
  ELSE
    RETURN F()
  ENDIF
ENDFUNCTION
```

Figure 6 shows a graph consisting of three nodes, the contents of the array `ConnectedNodes` when it is used to represent this graph, and the contents of the array `Visited` after the subroutine call `G(0, -1)`.

Figure 6



ConnectedNodes

[0]	[1]	[2]
2		0

Visited

[0]	[1]	[2]
True	False	True

0 2 . 3 Complete the unshaded cells in **Table 3** to show the result of the subroutine call `F()` when it is applied using the graph shown in **Figure 6**.

Table 3

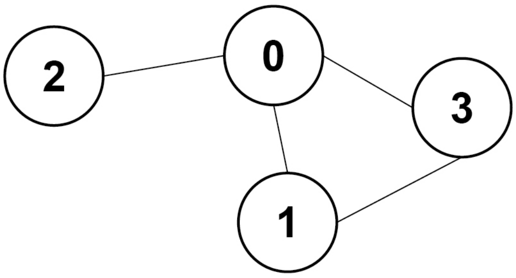
Count	Value returned

Copy the contents of the unshaded cells in **Table 3** into the table in your Electronic Answer Document.

[2 marks]

Figure 7 shows a graph consisting of four nodes and the contents of the array `ConnectedNodes` when it is used to represent this graph.

Figure 7



ConnectedNodes

[0]	[1]	[2]	[3]
1,2,3	0,3	0	0,1

0 2 . 4 Complete the unshaded cells in **Table 4** to show how the graph in **Figure 7** would be represented as an adjacency matrix.

Table 4

	0	1	2	3
0				
1				
2				
3				

Copy the contents of the unshaded cells in **Table 4** into the table in your Electronic Answer Document.

[1 mark]

Figure 3 (repeated)

```
FUNCTION G(V, P)
  Visited[V] ← True
  FOR EACH N IN ConnectedNodes[V]
    IF Visited[N] = False THEN
      IF G(N, V) = True THEN
        RETURN True
      ENDIF
    ELSE IF N ≠ P THEN
      RETURN True
    ENDIF
  ENDFOR
  RETURN False
ENDFUNCTION
```

0 2 . 5 Complete the unshaded cells in **Table 5** to show the result of the subroutine call `G(0, -1)` on the graph shown in **Figure 7**. Some parts of the table, including the initial values in the `Visited` array, have been completed for you.

Table 5

			Visited				
Subroutine call	**V**	**P**	**[0]**	**[1]**	**[2]**	**[3]**	**N**
			False	False	False	False	
G(0, -1)							
Final value returned:							

Copy the contents of the unshaded cells in **Table 5** into the table in your Electronic Answer Document.

[6 marks]

0 2 . 6 What is the purpose of the subroutine `G`?

[1 mark]

0 2 . 7 State the type of graph traversal used in subroutine `G`.

[1 mark]

0 2 . 8 If the graph represented by `ConnectedNodes` is undirected, what can you determine about the graph when a value of `True` is returned by subroutine `E`?

[1 mark]