Linear and Binary Search

Q1.

A programmer wants to implement a search algorithm to be used with small arrays. The figure below shows an example array.

```
[4, 6, 8, 12, 15, 16, 21]
```

(a) Using the figure above, explain how linear search would search for the integer `15`.

(4)

(b) What property of the example array in above the figure above means the programmer could use a binary search algorithm?

(1)

(c) The programmer knows that a binary search algorithm is more efficient than a linear search algorithm. Explain why the efficiency of these two algorithms is not an important factor when choosing what algorithm to implement for the array in the figure above.

(2)

(Total 7 marks)

Q2.

The algorithm below is the binary search algorithm designed to search for a value within an array.

- Line numbers are included but are not part of the algorithm.
- For this algorithm, array indexing starts at 1.

```
1   val ← 43
2   arr ← [3, 5, 13, 43, 655, 872]
3   left ← 1
4   right ← LENGTH(arr)
5   WHILE left ≠ right
6     mid ← (left + right) DIV 2
7     IF val ≤ arr[mid] THEN
8       right ← mid
9     ELSE
10      left ← mid + 1
11    ENDIF
12  ENDWHILE
```

(a) Complete the trace table for the algorithm above (you may not need to use all of the rows in the table). The final value of `left` is already given.

val	**left**	**right**	**mid**	**arr[mid]**
	4			

(5)

(b) Why would the binary search algorithm shown above not work when the array `arr` contains `[5, 3, 13, 872, 655, 43]`?

____

____

(1)

(c) There are alternative statements that could have been used on line 5 of the algorithm shown above that would not change the functionality of the algorithm.

Shade **one** lozenge to show which of the following lines could **not** replace line 5 in the algorithm above as it would change the functionality of the algorithm.

New Line 5

A `WHILE left < right` ○

B `WHILE NOT (left = right)` ○

C `WHILE left < right AND left > right` ○

D `WHILE left < right OR left > right` ○

(1)

(d) The final value of `left` in the algorithm above is `4`. A programmer realises that they can use this value to check whether `val` has been found or not in the algorithm above.

The programmer wants to extend the algorithm and introduce a new variable called `found` that is `true` when the value has been found in the array or `false` otherwise.

Write the pseudo-code or draw the flowchart that is needed to **extend** the algorithm so that when the algorithm finishes, the new variable `found` is:

- `true` when `val` is found in `arr`
- `false` when `val` is not found in `arr`

This code should follow on from the end of the algorithm above.

____

____

____

____

____

____

____

____

____

____

(4)

(Total 11 marks)