MODULE L STANDARD ALGORITHMS · L6

Comparing the search and sort algorithms

Original practice questions · 22 marks · ~25 minutes · OCR J277 spec 2.1.3

Name _____ Date _____

Instructions. Answer all questions in the spaces provided. The marks for each part are shown in brackets [ ]. The total for this worksheet is **22 marks**.

1

Total: 3 marks

(a) Tick (✓) **one** box in each row to show whether each statement is true for a **linear search** only, a **binary search** only, or **both**. **[3]**

Statement	Linear search only	Binary search only	Both
It can search a list that has not been sorted			
It compares the target with the middle item of the items that remain			
It can search a list that is in ascending order			

2

Total: 3 marks

(a) Tick (✓) **one** box in each row to show which sorting algorithm each statement describes. **[3]**

Statement	Bubble sort	Insertion sort	Merge sort
Of these three, it usually needs the most extra memory, because it creates new lists as it works			
It can stop early if it completes a pass through the list without making any swaps			
It takes each item in turn and inserts it into the correct place among the items before it			

3

Total: 3 marks

A parcel depot's system uses the function `findCode()` to look for a locker number (`target`) in the array `codes`. It is written in OCR Exam Reference Language.

```
01 function findCode(codes, target)
02     low = 0
03     high = codes.length - 1
04     while low <= high
05         mid = (low + high) DIV 2
06         if codes[mid] == target then
07             return mid
08         elseif codes[mid] < target then
09             low = mid + 1
10         else
11             high = mid - 1
12         endif
13     endwhile
14     return -1
15 endfunction
```

(a) Identify the searching algorithm used in `findCode()`. **[1]**

(b) State the pre-requisite that the array `codes` must meet for this algorithm to work. **[1]**

(c) Identify the line number of the line that works out the position of the middle item. **[1]**

4 **Total: 6 marks**

The array `codes` holds the 15 locker numbers that are in use, in ascending order. The first item is at position 0.

```
3   8   11   15   19   24   27   31   36   40   44   52   58   63   70
```

(a) A **linear search** is used to look for the locker number 58. State how many items the linear search checks to find 58, including 58 itself. **[1]**

(b) The function `findCode()` from Question 3 is used to look for 58 instead. Show the steps of the binary search, giving each item that is compared with 58 and the items that are discarded. **[3]**

(c) The depot then looks for the locker number 8. Explain why a linear search would find 8 using fewer comparisons than the binary search in `findCode()`. **[2]**

5 **Total: 2 marks**

A museum stores the badge numbers of the 25 visitors who are inside the building, in the order they arrived. The list is not sorted, and badge numbers are added and removed all day.

(a) Give **two** reasons why a linear search is more suitable than a binary search for checking whether a badge number is in this list. **[2]**

1

2

6 **Total: 5 marks**

(a) A weather service sorts about 2 million temperature readings every night. The sort runs on a large server that has plenty of spare memory. The service uses a merge sort. **[2]**
Explain why a merge sort is a suitable choice for this task.

(b) A small battery-powered sensor has very little memory. Every minute it sorts its last 20 readings, which are nearly always already in order. **[3]**
Recommend a sorting algorithm for the sensor. Justify your choice.

Algorithm

Total for this worksheet: 22 marks