2.3 ALGORITHMS · 2.3.1(F)

Linear & binary search

Original practice questions · 30 marks · about 45 minutes · spec 2.3.1(f)

Instructions. Answer all questions. Marks are shown in brackets [ ]. Show trace tables in full.

1

Total: 6 marks

(a) Describe how a linear search works. **[2]**

(b) State the worst-case time complexity of a linear search. _____ **[1]**

(c) State one advantage of a linear search over a binary search. _____ **[1]**

(d) State the precondition for a binary search. _____ **[1]**

(e) State the worst-case time complexity of a binary search. _____ **[1]**

2

Total: 8 marks

```
[5, 11, 19, 23, 34, 40, 55, 68, 72] (indexes 0–8)
```

(a) Complete the trace table for a **binary search** for the target **23**. Add rows as needed. **[6]**

low	high	mid	list[mid]
0	8		

(b) State the index at which 23 is found, and the number of steps taken. _____ **[2]**

3 Total: 6 marks

(a) Write an algorithm for a **linear search** that sets `found` to True if `target` is in `list`. **[4]**

(b) State the maximum number of comparisons a linear search makes on a list of n items. **[1]**

(c) State when this worst case occurs. **[1]**

4 Total: 4 marks

(a) A program repeatedly searches a large, sorted list. State which search is more suitable and why. **[2]**

(b) A program searches a small, unsorted list once. State which search is more suitable and why. **[2]**

5 Total: 6 marks

(a) A sorted list has 1000 items. State the approximate maximum number of steps a binary search would take, and explain why. **[2]**

(b) Explain why a binary search cannot be used on an unsorted list. **[2]**

(c) A list is unsorted but will be searched many times. Suggest what could be done so a binary search can be used, and one drawback. **[2]**

END OF QUESTIONS