

19 Linear search – Part 1

Name: _____ Class: _____ Date: _____

Vocabulary

Write each word three times.

English	Write it 3 times (English)
array	_____
linear search	_____

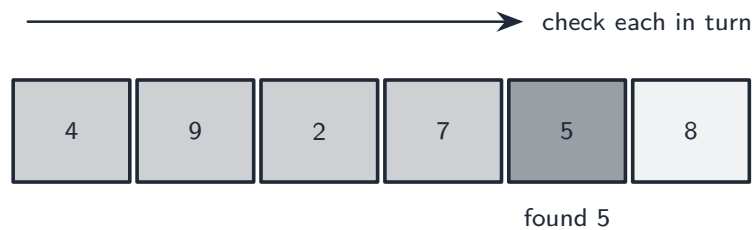
Recall

A **search** finds a target value in a collection (often an **array**) and returns its position, or “not found”.

New ideas

Read these first. Each idea has a short worked example. Then do the Practice.

■ 1 Linear search



A **linear search** walks from start to end, comparing each element with the target. No preparation is needed.

Worked example.

```
FOR i ← 1 TO n
  IF list[i] = target THEN
    RETURN i          // found at position i
  ENDIF
ENDFOR
RETURN -1             // not found
```

■ 2 When to use it



Linear search works on **any** list, sorted or not. Its worst case checks every element, so it is best for unsorted data or short lists.

Worked example. To find 7 in [3, 7, 2, 9], a linear search checks 3 (no), then 7 (yes!) and stops at position 2. If 7 were missing, it would check all four and report "not found".

Watch out: linear search checks up to n elements in the worst case, but it needs no sorting – it works on *any* list, unlike binary search.

Practice

Try these in order.

19.1 What does a *search* return? [1]

19.2 How does a *linear search* work? [1]

19.3 Does linear search need the list to be sorted? [1]

19.4 In the worst case, how many of the 10 elements might a linear search check? [1]

19.5 When is linear search a good choice? [1]

19.6 A linear search looks for 5 in the list [8, 3, 5, 1]. List the values it checks, in order, before it stops. [2]

19.7 Challenge. A linear search over n items checks about n of them in the worst case. Explain what happens to that worst-case work if the list size doubles. [2]
