

4 Searching and sorting – Part 2

Name: _____ Class: _____ Date: _____

Vocabulary

Write each word three times.

English

Write it 3 times (English)

Linear search

Binary search

Recall

In Part A you stored many values in an array. Now two everyday tasks:

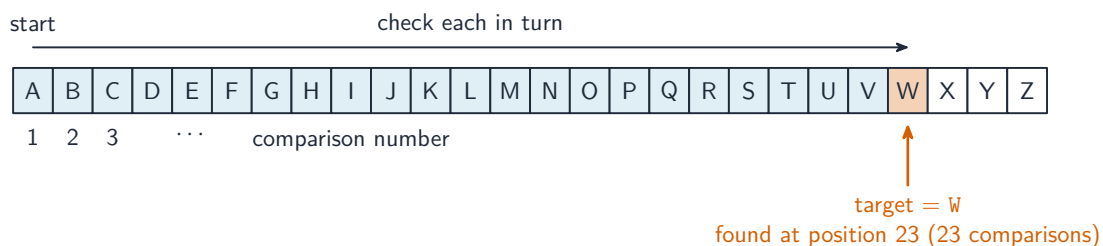
- **Finding** a value in a list.
- **Sorting** a list into order.

Each idea has a worked example before the Practice.

New ideas

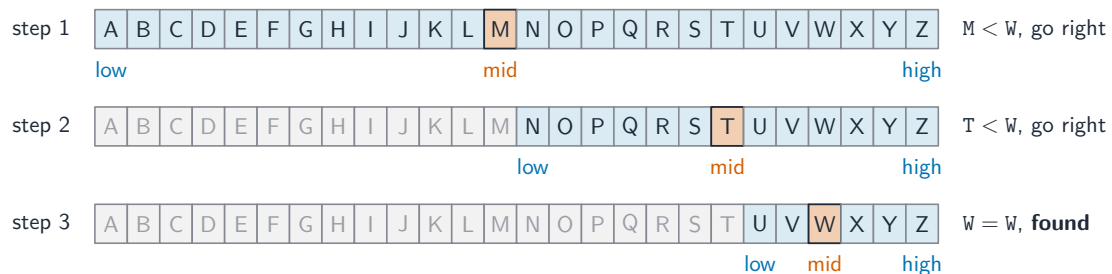
■ 1 Linear search

Linear search checks each element in turn until it finds the target. It works on **any** list and takes up to n steps.



■ 2 Binary search

Binary search works only on a **sorted** list: check the middle, then throw away the half that cannot contain the target, and repeat. Each step halves the list, so it takes about $\log_2 n$ steps.



Worked example. Search a sorted list of 8 items: $8 \rightarrow 4 \rightarrow 2 \rightarrow 1$, at most 3 comparisons – far fewer than checking all 8.

■ 3 Sorting

Sorting puts a list in order. Simple methods like **selection sort** repeatedly find the smallest remaining value and place it next.

■ 4 Why sorting helps

Once a list is sorted, binary search can find items much faster – which is why sorting is so useful.

Watch out: binary search only works on a **sorted** list. Running it on an unsorted list gives wrong answers – you must sort first.

Practice

Try these in order. The methods are all above.

2.1 State how linear search looks for a value. [1]

2.2 State the one thing binary search requires of the list. [1]

2.3 A sorted list has 16 items. State roughly the most comparisons binary search needs ($\log_2 16$). [2]

2.4 Explain why binary search is much faster than linear search on a large sorted list. [2]

2.5 State what would go wrong if you ran binary search on an unsorted list. [2]