Algorithm design strategies: decrease and conquer

1. Design strategies

Some algorithm design techniques:

• decrease and conquer (incremental)

• divide and conquer

• transform and conquer

• dynamic programming (solve all subproblems)

• greedy

• randomize

These are design categories, not algorithms themselves.

2. Decrease and conquer strategy

How does it work?

• at start: all input data is unprocessed

• at each iteration part of input data is processed (completely)

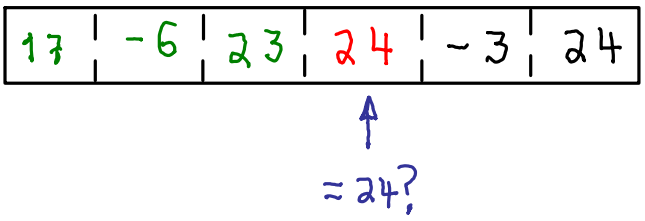
• after each iteration, the amount of processed data increases

• algortihm stops when all data is processed

Example

Linear search through array for particular element

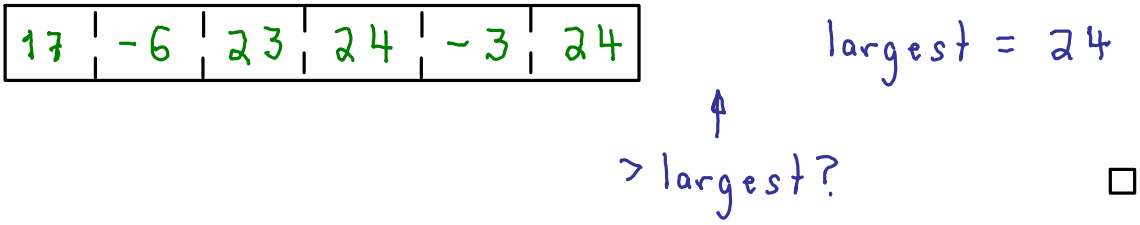
Search for 24 in array



□

Example

Find largest number in numerical array

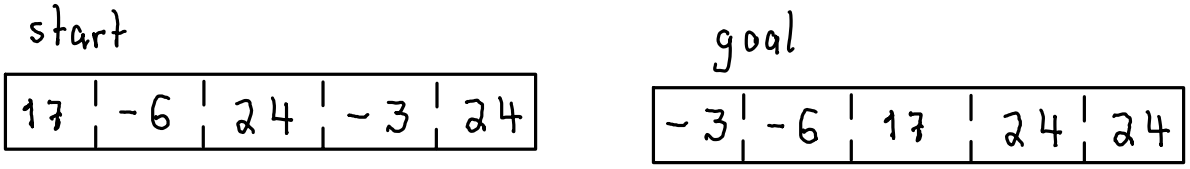


□

Example

Insertion sort: a method for sorting elements in a list/array

Sort numbers in array from smallest to largest



□

Strengths

• clear, intuitive

• easy to implement (small amount of code)

• works well for small amounts of data

Weakness

• often better (=more efficient) alternatives for larger amounts of data