

1F Order of an Algorithm/ Run Time

How changes in size affect time taken.

Bubble sort

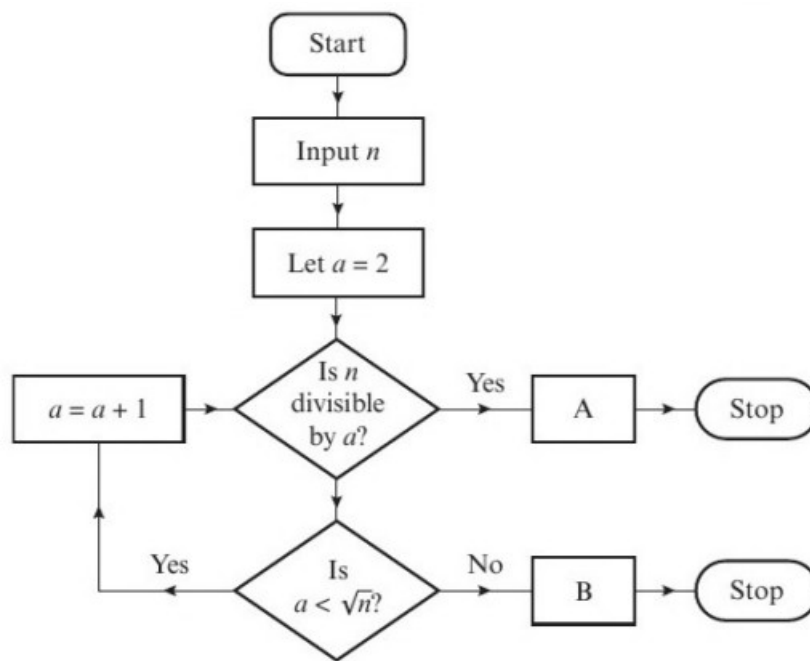
If a bubble sort has n items, how many comparisons are made on the first pass?

Given at least one item has been sorted, how many potential comparisons could be made on the second pass?

What would the total number of comparisons potentially be?

The order of an algorithm can be described as a function of its size. If an algorithm has order $O(n^2)$ then increasing the size of the problem from n to $2n$ will increase the run time by a factor of approximately

1. An algorithm is defined by the below flow diagram, where n and k is an integer.



a) Describe what the algorithm does

b) Suggest suitable output text for boxes A and B.

c) Determine the order of the algorithm (how many potential steps are there?).

In general, if the size of a problem is increased by some factor :

- An algorithm of linear order will take approximately times as long to complete
 - An algorithm of quadratic order will take approximately times as long to complete
 - An algorithm of cubic order will take approximately times as long to complete
2. A given list-searching algorithm has linear order.

To search a list with 500 values, the algorithm takes approximately 0.036 seconds.

Estimate the time taken by this algorithm to search a list containing 1250 values.

3. The bubble sort algorithm has quadratic order
 - a) Given that it takes a computer 0.2 seconds to sort a list of 1600 numbers using the bubble sort, estimate the time needed to sort a list of 480000 numbers.
 - b) Comment on the suitability of the bubble sort algorithm for sorting large data sets.