Sorting Algorithms (II)

2023

Sarah Chan
sarah.chan@uwaterloo.ca

The Centre for Education in Mathematics and Computing
Faculty of Mathematics, University of Waterloo
www.cemc.uwaterloo.ca

Recap

What did we learn last week?

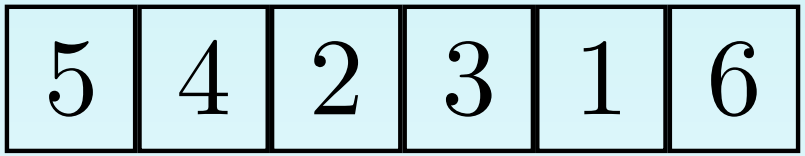
Recap

What did we learn last week?

- There are benefits to having sorted data
- Sorting is an essential skill for computer systems
- There are dozens of different sorting algorithms
- Selection sort is an algorithm that repeatedly finds the smallest value and swaps it to the front
- Insertion sort is an algorithm that places each value in its relative sorted position
- Bubble sort is an algorithm that pushes large values to the back

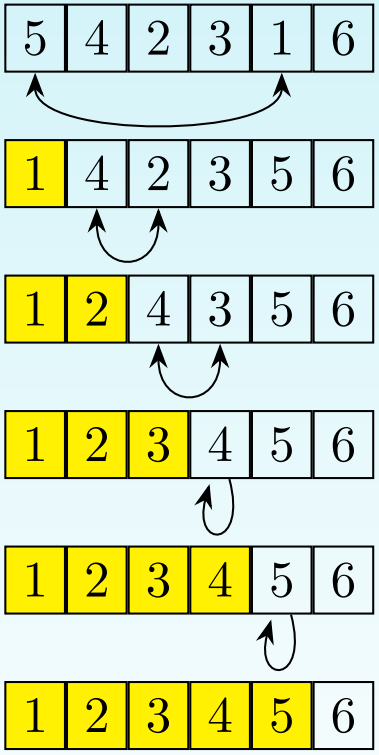
Review

Sort the following list of numbers in ascending order using selection sort, insertion sort, and bubble sort:



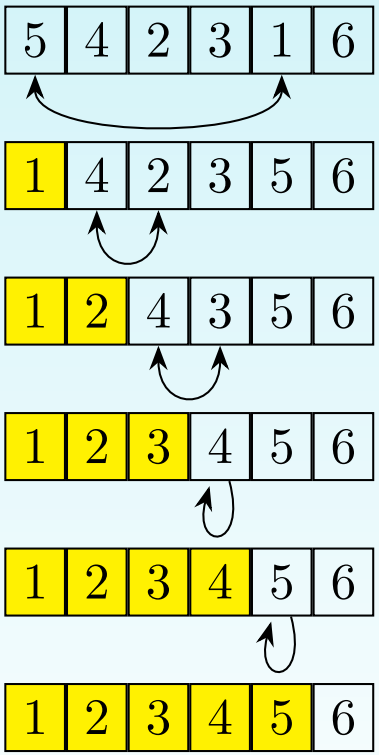
Review

Selection Sort

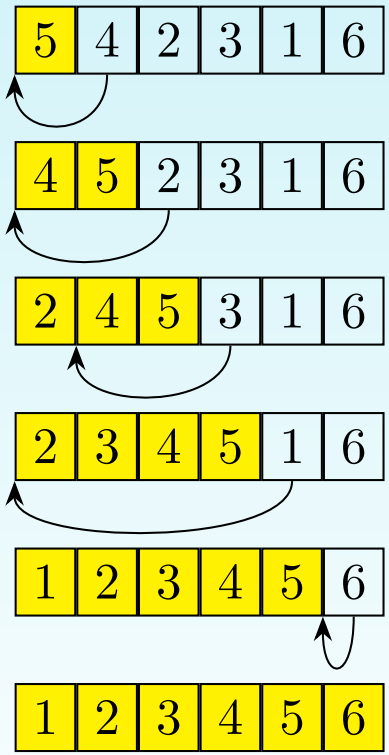


Review

Selection Sort

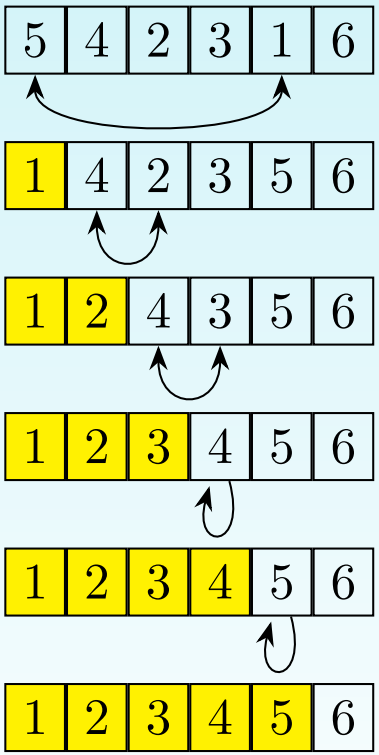


Insertion Sort

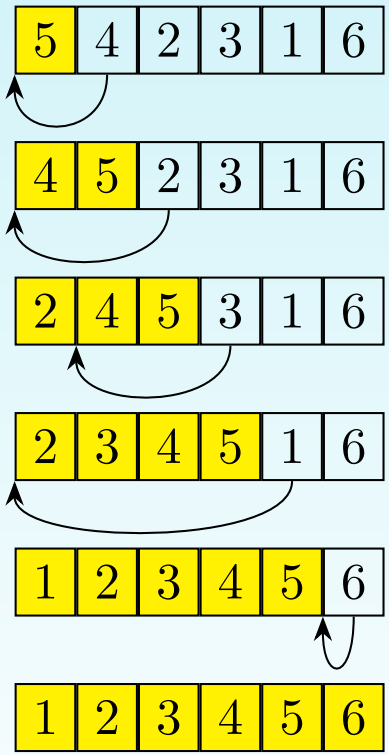


Review

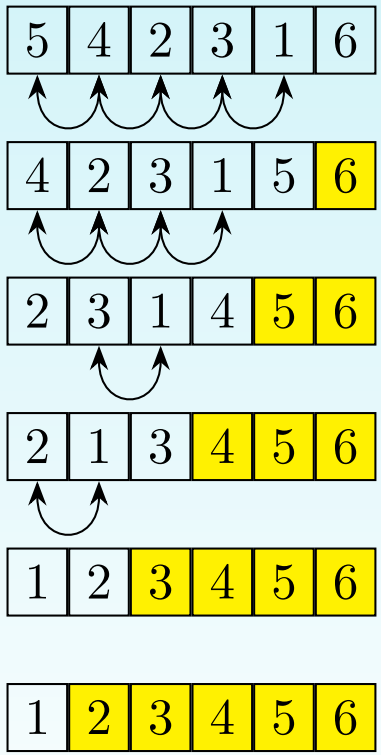
Selection Sort



Insertion Sort



Bubble Sort

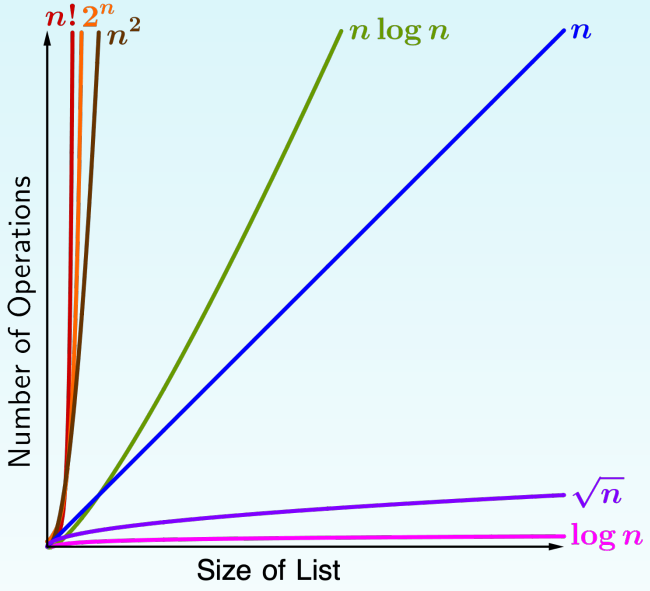


Review

- Selection sort *always* requires $\frac{n(n-1)}{2}$ comparisons
- Insertion sort requires *at most* $\frac{n(n-1)}{2}$ comparisons
- Bubble sort *always* requires $\frac{n(n-1)}{2}$ comparisons

Review

- Selection sort *always* requires $\frac{n(n-1)}{2}$ comparisons
- Insertion sort requires *at most* $\frac{n(n-1)}{2}$ comparisons
- Bubble sort *always* requires $\frac{n(n-1)}{2}$ comparisons



On average, if a list contains n elements, then **selection sort**, **insertion sort**, and **bubble sort** all take time approximately equal to n^2 (quadratic time) in order to sort.

Can we do better?

Quicksort

Pick an element, called the **pivot**. Divide the list into two sublists: those that are less than (or equal to) the pivot and those that are greater than the pivot.

The pivot can be chosen in a variety of ways:

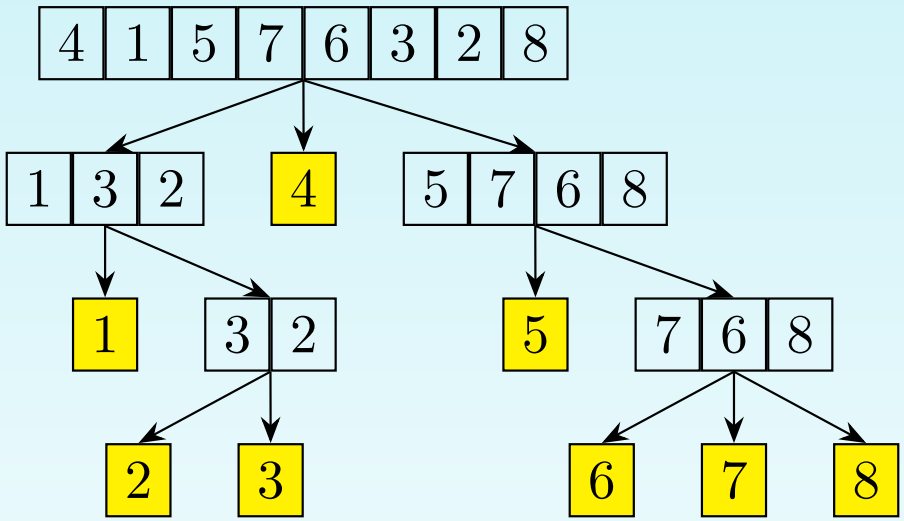
- pick the first element
- pick the last element
- pick a random element
- pick the median element

After dividing the list, sort each sublist using **quicksort**.

A list with one element is considered sorted.

Quicksort: Example

Using the first element as the pivot:



Quicksort: Problem Set

1. Sort the following list of letters in alphabetical order using quicksort, by choosing the *last* element as the pivot:

   D A H G F C B E

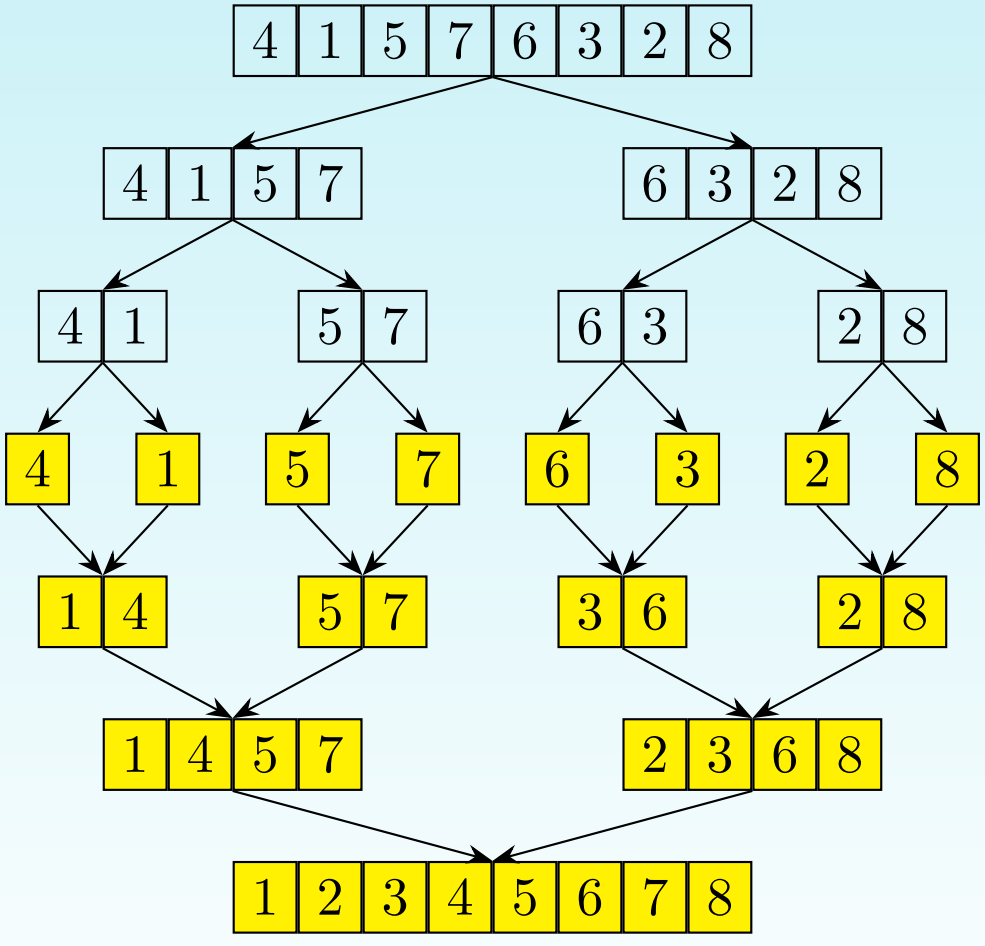
2. What ideal property should the pivot have?
3. On average, if a list contains n elements, then approximately how much time will quicksort take in order to sort the list?

Merge Sort

Divide the list in half. Sort each half using **merge sort**. Merge the two sorted halves back together.

A list with one element is considered sorted.

Merge Sort: Example



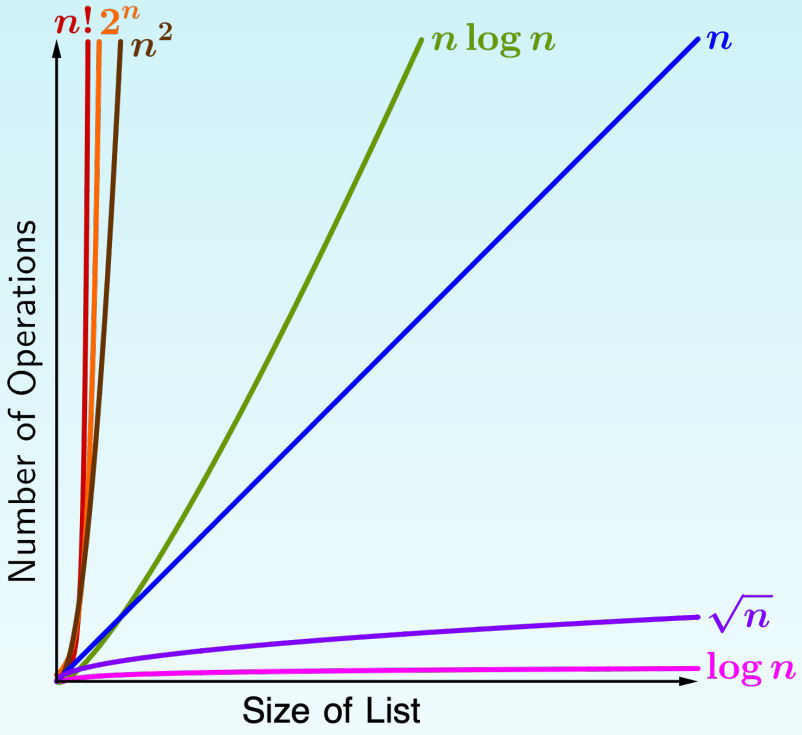
Merge Sort: Problem Set

1. Sort the following list of letters in alphabetical order using merge sort:

   D A E G F C B H

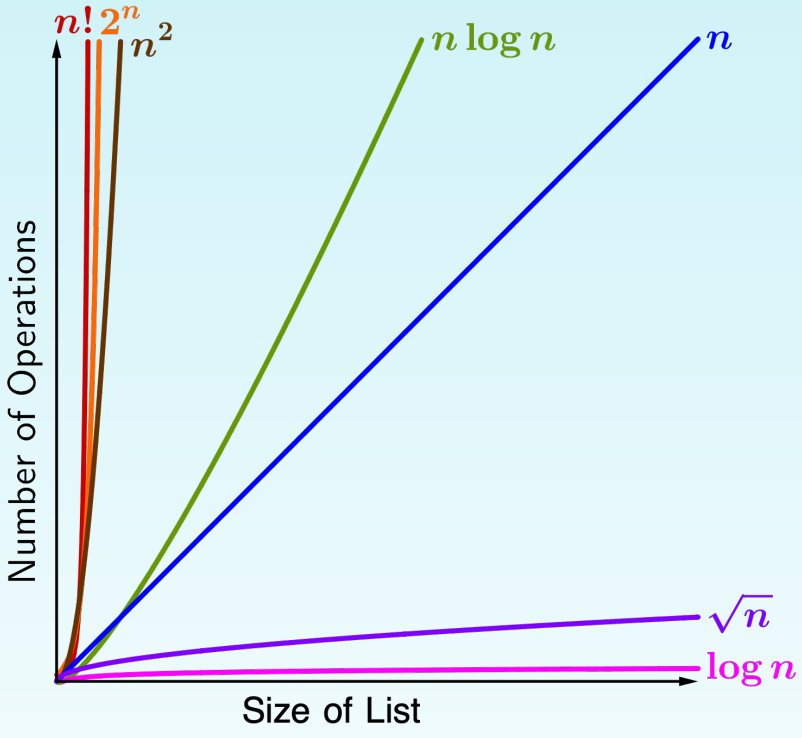
2. On average, if a list contains n elements, then approximately how much time will merge sort take in order to sort the list?

What Does All This Mean?



Selection sort, insertion sort, and bubble sort are all n^2 on average. With lucky data (nearly sorted) insertion sort reduces to n.

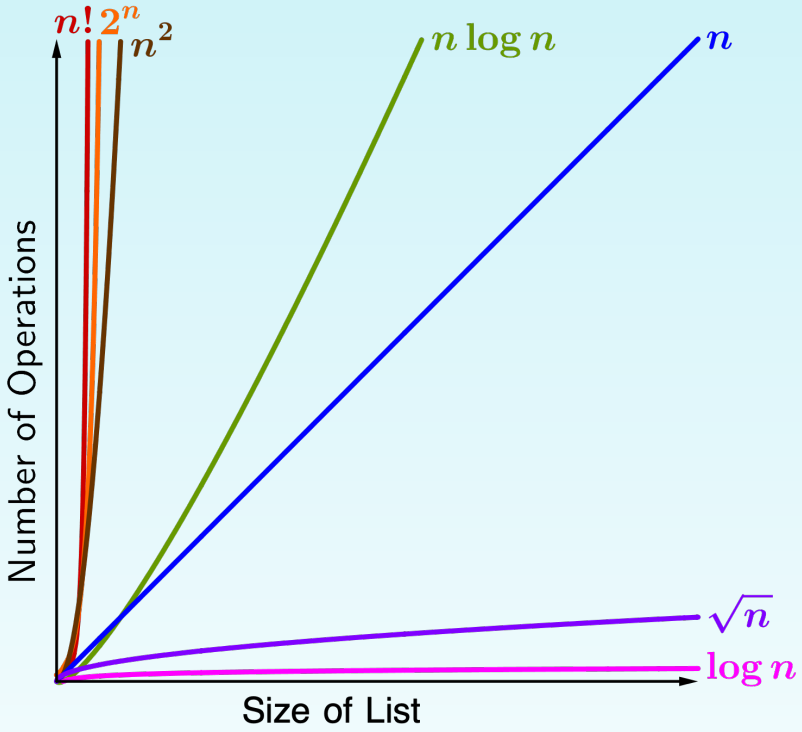
What Does All This Mean?



Selection sort, insertion sort, and bubble sort are all n^2 on average. With lucky data (nearly sorted) insertion sort reduces to n.

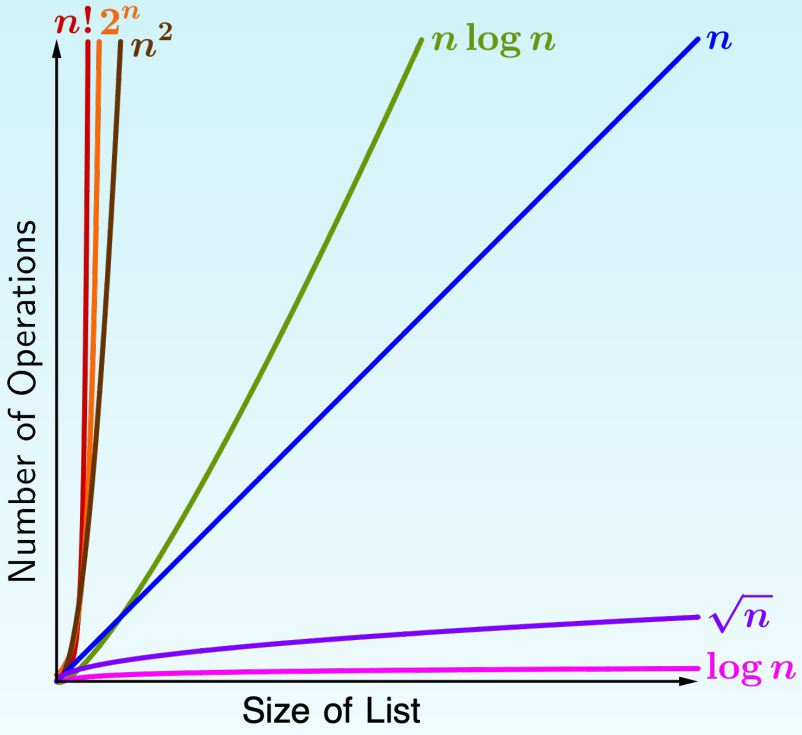
Quicksort and merge sort are both $n \log n$ on average. With unlucky data (bad pivots) quicksort grows to n^2.

What Does All This Mean?



It can be proven that comparison based sorting algorithms cannot do better than $n \log n$ on average. Radix sort is not comparison based and it runs in n (linear) time.

What Does All This Mean?



It can be proven that comparison based sorting algorithms cannot do better than $n \log n$ on average. Radix sort is not comparison based and it runs in n (linear) time.

In practice, you need to consider the size and state of the data, the speed of an algorithm, and how much memory space the algorithm needs, before selecting a particular sorting technique.

A Visual Summary

The following video summarizes both the steps and the runtime performance of a variety of sorting algorithms.

https://safeshare.tv/x/qk7b4-iyCJ4

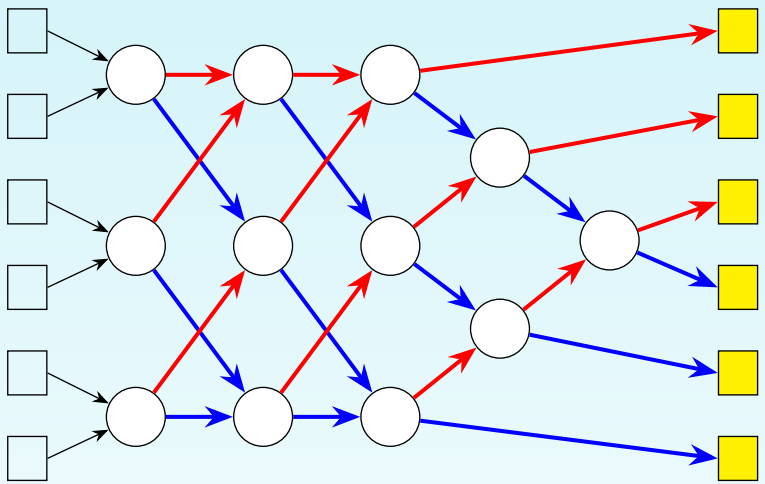
Bubble Sort (00:46)
Selection Sort (1:27)
Insertion Sort (2:15)
Merge Sort (3:09)
Quicksort (4:05)
Heapsort (5:35)
Timsort (7:38)
Introsort (8:49)

Challenge: Problem Set

1. Quicksort and merge sort are examples of **recursive algorithms**. A recursive algorithm solves a problem by combining the solutions to smaller instances of the same problem. Create a recursive algorithm that computes the factorial of a number.
2. A sorting algorithm is considered **stable** if duplicate elements maintain their relative order after sorting. For instance, if the original list contains 5_a and 5_b in that order, a stable sort will keep 5_a and 5_b in that order. If the sorted list ends up having 5_b first and then 5_a, then the sorting algorithm is not stable. Of selection, insertion, bubble, quick, and merge sort, which sorting algorithms are stable?

Challenge: Problem Set

3. Sorting can be sped up using a sorting network. Below is a sorting network that sorts 6 items.



The leftmost column is the unsorted list. The list elements move through the sorting network by following the arrows. Each circle represents a comparison between two elements.

Challenge: Problem Set

The smaller elements in each comparison follow the higher red arrows and the larger elements follow the lower blue arrows. The rightmost column is the sorted list.

Circles that are lined up vertically represent comparisons that can be performed at the same time using **parallel processing**.

a) Use the sorting network to sort the list: **5 1 6 3 4 2**

b) Design a sorting network that sorts 3 items.

c) Design a sorting network that sorts 4 items.