CE2101/ CZ2101: Algorithm Design and Analysis

Introduction to Sorting

Ke Yiping, Kelly

NANYANG TECHNOLOGICAL UNIVERSITY

Instructor's Information

- **Name:** Ke Yiping, Kelly
- **Phone:** 6790-5046
- **Email:** ypke@ntu.edu.sg
- **Office:** N4-02a-12

Learning Objectives

At the end of this lecture, students should be able to:

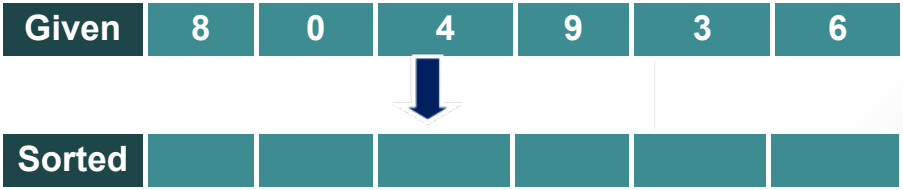
- Define what is sorting
- Explain why we learn sorting
- Analyse the objective and evaluation of sorting algorithms

What is Sorting?

Definition (sorting in ascending order):

- Given a set of records **r1, r2, …, rn** with key values **k1, k2,…, kn,** arrange records in order ***s*** such that records **rs1, rs2,…, rsn** have keys with property **ks1 ≤ ks2 ≤ … ≤ ksn.**

Example:

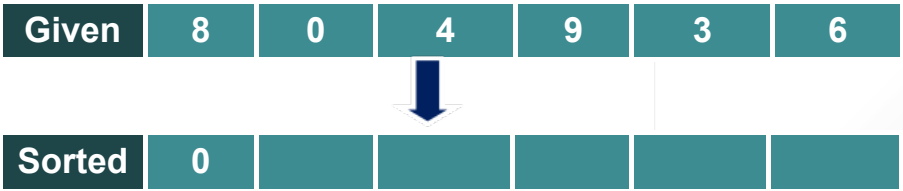


What is Sorting?

Definition (sorting in ascending order):

- Given a set of records **r1, r2, …, rn** with key values **k1, k2,…, kn,** arrange records in order ***s*** such that records **rs1, rs2,…, rsn** have keys with property **ks1 ≤ ks2 ≤ … ≤ ksn.**

Example:

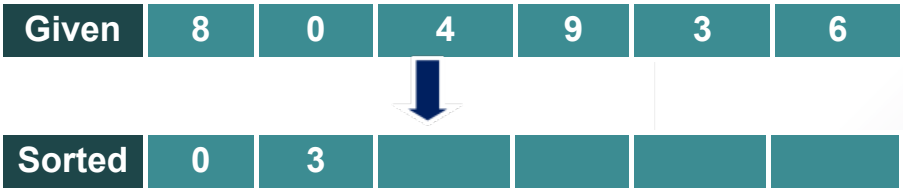


What is Sorting?

Definition (sorting in ascending order):

- Given a set of records **r1, r2, …, rn** with key values **k1, k2,…, kn,** arrange records in order ***s*** such that records **rs1, rs2,…, rsn** have keys with property **ks1 ≤ ks2 ≤ … ≤ ksn.**

Example:

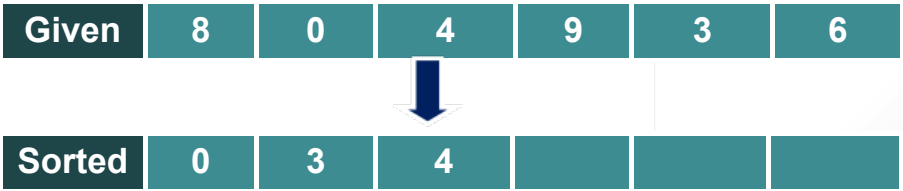


What is Sorting?

Definition (sorting in ascending order):

- Given a set of records **r1, r2, …, rn** with key values **k1, k2,…, kn,** arrange records in order ***s*** such that records **rs1, rs2,…, rsn** have keys with property **ks1 ≤ ks2 ≤ … ≤ ksn.**

Example:

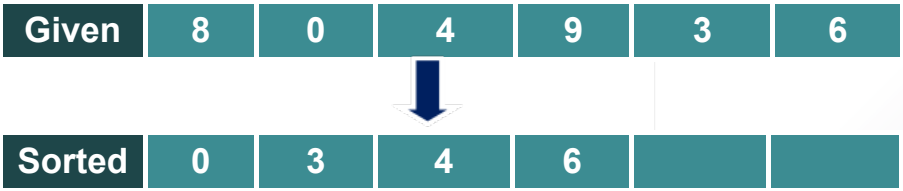


What is Sorting?

Definition (sorting in ascending order):

- Given a set of records **r1, r2, …, rn** with key values **k1, k2,…, kn,** arrange records in order ***s*** such that records **rs1, rs2,…, rsn** have keys with property **ks1 ≤ ks2 ≤ … ≤ ksn.**

Example:

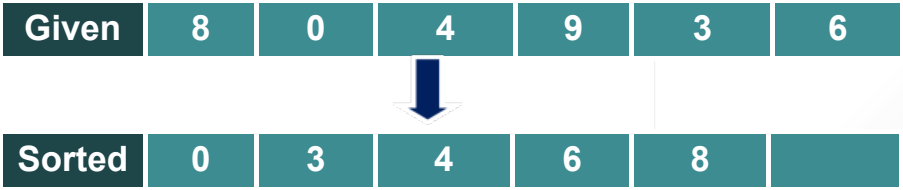


What is Sorting?

Definition (sorting in ascending order):

- Given a set of records **r1, r2, …, rn** with key values **k1, k2,…, kn,** arrange records in order ***s*** such that records **rs1, rs2,…, rsn** have keys with property **ks1 ≤ ks2 ≤ … ≤ ksn.**

Example:

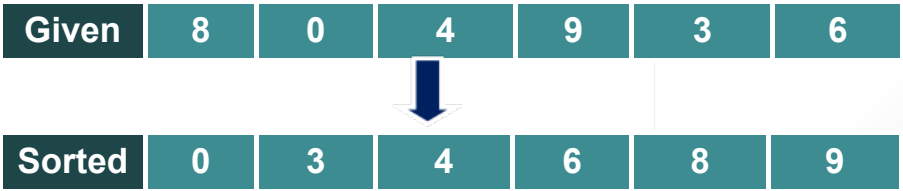


What is Sorting?

Definition (sorting in ascending order):

- Given a set of records **r1, r2, …, rn** with key values **k1, k2,…, kn,** arrange records in order ***s*** such that records **rs1, rs2,…, rsn** have keys with property **ks1 ≤ ks2 ≤ … ≤ ksn.**

Example:



NANYANG TECHNOLOGICAL UNIVERSITY

Spark 10^{15}

TeraSort 10^{12}

Why do we learn sorting?

Why do we learn sorting?

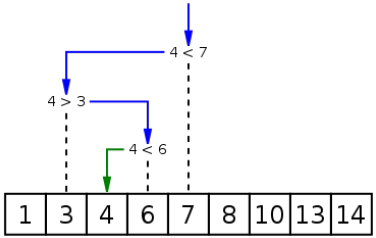
- Things must be kept in some order if we want to find them quickly.

Reference: Row Of Books Clipart. (n.d.). Retrieved May 16, 2016, from http://cliparts.co/clipart/39651.

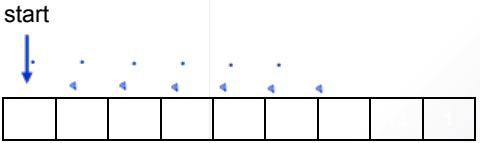
Why do we learn sorting?

- Things must be kept in some order if we want to find them quickly.
- How to arrange things in order? Sorting algorithms.
- Sorting is a basic building block for many algorithms.

Binary Search



Sequential Search



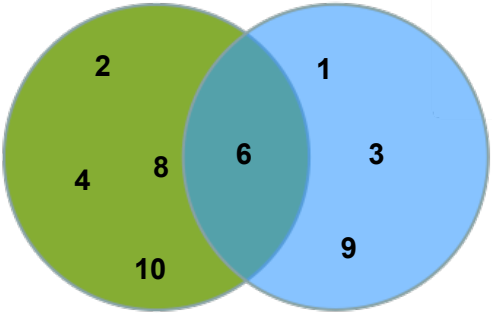
Reference: T. (2015, April 19). Binary search in a sorted array. Retrieved May 18, 2016, from https://commons.wikimedia.org/wiki/File:Binary_search_into_array.png#/media/File:Binary_search_into_array.png.

Why do we learn sorting?

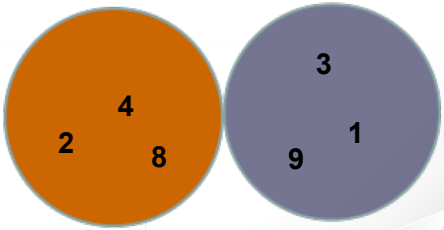
- Things must be kept in some order if we want to find them quickly.
- How to arrange things in order? Sorting algorithms.
- Sorting is a basic building block for many algorithms.
- Most thoroughly studied problem in Computer Science.
- To learn ideas in Algorithm Design derived from techniques in sorting.

Example: Disjoint Sets

- **Problem:** Determine whether two sets (both of size *n*) are disjoint.

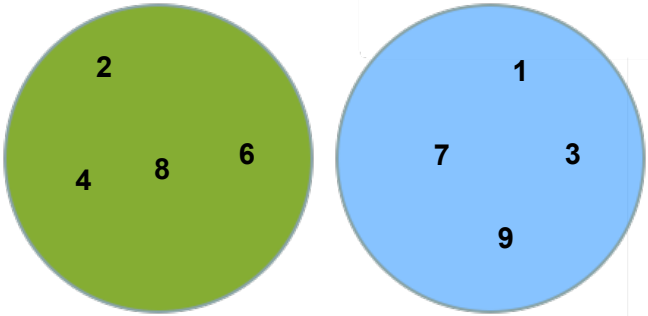


Disjoint



Example: Disjoint Sets

- **Problem:** Determine whether two sets (both of size *n*) are disjoint.
  - **Solution 1:** Compare each element of the 1st set with each element of the 2nd set. That is, ***n2*** comparisons.



Example: Disjoint Sets

- **Problem:** Determine whether two sets (both of size *n*) are disjoint.
  - **Solution 1:** Compare each element of the 1st set with each element of the 2nd set. That is, ***n2*** comparisons.
  - **Solution 2:**

    Step 1: We first **sort the first set into ascending order**. This takes $O(n \lg n)$ effort using Mergesort or Heapsort.

    Step 2: For each element in the 2nd set, we **use Binary Search** to find it in the 1st set. This takes $O(n \lg n)$ time.

Comparison of Performance

Solution 1: $O(n2)$

Solution 2: $O(n \lg n)$

Savings:

n	= 64	128	256	512
$n2$	= 4,096	16,384	65,536	262,144
$n \lg n$	= 384	896	2,048	4,608

Comparison of Performance

The data items to be sorted:

- Given a (very large) list of records.
- Each record has the following form: key; rest info of record:

```
class ALIST {
    KeyType     key;
    DataType    data;
};
```

- Key domain is an ordered set.
- **Objective:** To arrange records in 'ascending' or 'descending' order.

Comparison of Performance

The data items to be sorted:

- Given a (very large) list of records.
- Each record has the following form: key; rest of record:

```
class ALIST {
    KeyType     key;
    DataType    data;
};
```

- Key domain is an ordered set.
- **Objective:** To arrange records in 'ascending' or 'descending' order.

Comparison of Performance

- Sorting can be classified into **internal sorting** and **external sorting**!
  - We focus on internal sorting only,

    i.e., all records are in (high speed) main memory during sorting.
- **Sorting involves two basic actions:**

1) key comparisons between two records
2) swapping records around

- **Goal:** Use minimum working space and do as few key comparisons as possible.

Summary

- Sorting is to arrange a set of records so that their key values are in ascending or descending order.
- It is important to learn sorting, because:
  - Sorting has important applications
  - Ideas of sorting can be used for other algorithms
- Objective is to design sorting algorithms with:
  - Minimum usage of memory
  - Minimum number of key comparisons or swaps