
Hint: find minimum and maximum numbers, say $a$ and $b$. Define $g = (b - a)/(n - 1)$. Clearly, the maximum gap has $A[j] - A[i] \geq g$. Split the numbers in the array into intervals of the form $[a + kg, a + (k + 1)g)$. The maximum gap cannot be within one of these intervals. This method is inspired by counting sort because you map $A[m]$ into the interval number $\lfloor (A[m] - a)/g \rfloor$, and we can sort them according to these interval numbers.

**Problem 2.** Radix sort. In a railroad yard, we have three tracks labeled $B$, $C$ and $D$, which are connected by a switch to track $A$:

A railroad car can move from $A$ to $B$, $C$ or $D$ by rolling down on a gentle slope. We can release and engage the brakes in a car by remote control, and we can operate the switch to direct the car to $B$, $C$ or $D$.

\[ \text{track A} \quad \text{track B} \quad \text{track C} \quad \text{track D} \]

a. Suppose that we have some number of cars on tracks $B$, $C$ and $D$, and that we have one locomotive on track $A$. How can we move all of the cars to $A$ using the one locomotive? The locomotive can push or pull cars, and you can connect cars to each other or disconnect them.

b. Now suppose that we have a train consisting of a locomotive followed by a string of cars on track $A$, with the train facing away from the switch (that is, the locomotive is farther from the switch than the cars are). The cars have destinations 1, 2, $\cdots$, 9 in some random order. (Note that many cars may have the same destination.) We want to rearrange the cars in the train so that cars to 1 are in front, followed by cars to 2, then to 3, and so forth. Describe a method for sorting the cars in this fashion using tracks $A$, $B$, $C$ and $D$ efficiently; that is, you want the total number of times that cars cross the switch to be as small as you can make it.

Hint: it suffices to roll cars from $A$ to the other tracks and return them to $A$, repeating this roll-return process a suitable number of times, if you do it correctly.
Problem 3. Radix sort. You are given an array of characters $C[N]$ and an array $A[n + 1]$ such that $0 = A[0] \leq A[1] < \cdots \leq A[n + 1] = N$. String $s_i$ is the sequence of characters in the array sector $(C, A[i], A[i + 1])$. We define lexicographic order of strings:

- if $\lambda$ is an empty string and $s$ any other string then $\lambda < s$;
- if $c, d$ are characters, $s, t$ are strings and $c < d$ then $cs < dt$;
- if $c$ is a character, $s, t$ are strings and $s < t$ then $cs < ct$.

Your task is to sort the strings, e.g. initialize

```plaintext
for (i = 0; i < n; i++)
    S[i] = i;
```

and then to permute $S[n]$ in such a way that $s_{S[i]} \leq s_{S[i+1]}$ for $i = 0, 1, \cdots, n-2$. You should use $O(N)$ time.

Hint: start by sorting strings by their length. Let $L$ be the maximum length. You should use Radix sort, but when you are performing census and distribution according to $k$-th characters of the strings you cannot inspect strings that do not have that characters, i.e. that are shorter than $k$.

Problem 7 from page 271.