Chapter 8, Strings:

- ▶ General introduction; string sorting (**Today**)
- ▶ Tries (next week Monday)
- ▶ Other string topics (next week Wednesday)
  - ▶ Regular expressions
  - ▶ ~~Huffman encoding~~
  - ▶ ~~Edit distance~~
  - ▶ ~~Grammars and parsing~~

Today:

- ▶ Why we care about strings
- ▶ Sorting strings
  - ▶ String quick sort
  - ▶ String bucket sort
  - ▶ String radix sort

End-of-semester important dates

- Mon, Apr 28: Last project assigned
- Tues, Apr 29: Last "normal" running of project grading script
- Wed, Apr 30: Test 3 & 4 Review sheet distributed
- Thurs, May 1: Review lab (pick practice problems for Test 4)
- Fri, May 2, AM: "Two-minute warning" running of project grading script (Canvas gradebook will not be updated—see project report in your turn-in file)
  Note that Fri, May 2 is the *Last Day of Classes*.
- Fri, May 2, midnight: Official project deadline
- Sat, May 3, when I wake up: Permissions to turn-in folders turned off
- Mon, May 5: Project grading script run for final/semester grades
- Tues, May 6, 1:30-3:30pm: Tests 3 and 4 (in lab)

Why we care about strings

▶ Strings are different
▶ Strings are common
▶ Strings are a representative example

```java
public class DNASequence {
    /** An alphabet for DNA */
    private static enum Nucleotide { A, C, G, T }
    /** The string of nucleotides */
    private Nucleotide[] sequence;
}
```

```java
public class BigInt {

    private byte[] digits;

    /** Compute the sum of this and another BigInt. */
    public BigInt add(BigInt other) {
        // The result object
        BigInt sum = new BigInt();
        // The result object has at most one more digit
        // than the larger number of digits of the two addends
        sum.digits = new byte[(digits.length > other.digits.length?
                digits.length : other.digits.length) + 1];
        // Add by column
        int carry = 0;
        for (int i = 0; i < sum.digits.length; i++) {
            // Digits in current columns of the two addends
            int a = digits.length <= i? digits[i] : 0;
            int b = other.digits.length <= i ? other.digits.length : 0;
            // The sum of the current digits plus carry from previous iteration
            int s = a + b + carry;
            // Mod that sum by 256 to get the appropriate digit in result,
            // divide to get the carry for next time.
            sum.digits[i] = (byte) (s % 256);
            carry = s / 256;
        }
        assert carry == 0;
        return sum;
    }
}
```

Quick sort:
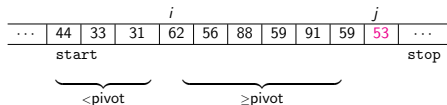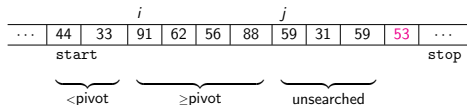
Invariant 11 (Loop of `partition()`**)**

(a) $\texttt{start} \le i \le j < \texttt{stop}$.

(b) $\forall\ k \in [\texttt{start}, i)$, $\texttt{sequence}[k] < \texttt{sequence}[\texttt{stop} - 1]$.

(c) $\forall\ k \in [i, j)$, $\texttt{sequence}[k] \ge \texttt{sequence}[\texttt{stop} - 1]$.

(d) $j - \texttt{start}$ is the number of iterations completed.

dais	card	bark	care	even	barb	doze	cart	carb	axle	daze	exam	axis	bard	carp

card	bark	care	barb	carb	axle	axis	bard	carp	dais	even	doze	cart	daze	exam

barb	axle	axis	bard	card	bark	care	carb	$\cdots$

		i			*j*			*k*						
bark	barb	card	care	cart	dais	even	doze	carb	axle	daze	exam	axis	bard	carp

start   stop

    <pivot   =pivot   ≥pivot   unsearched

Invariant 40. [Loop of string_quick_sort_r() **]**

Let c be the character in position pre in the string in position stop $- 1$.

(a) start $\leq i \leq j \leq k <$ stop

(b) (Informal) For all the strings in range $[\text{start}, i)$, their character in position pre is less than c.

(c) (Informal) For all the strings in range $[i, j)$, their character in position pre is equal to c.

(d) (Informal) For all the strings in range $[i, j)$, their character in position pre is greater than to c.

(e) $k - \text{start}$ is the number of iterations completed.

						i				j				k
bark	barb	axle	axis	bard	card	care	cart	carb	carp	dais	even	doze	daze	exam

start   stop

$\underbrace{\qquad\qquad\qquad}_{<\text{pivot}}$ $\underbrace{\qquad\qquad\qquad}_{=\text{pivot}}$ $\underbrace{\qquad\qquad\qquad}_{\geq\text{pivot}}$
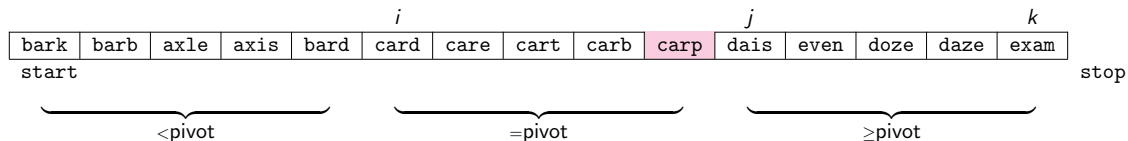
Invariant 41. [Precondition of `string_quick_sort_r()`**]**
$\forall\ i, j \in [\mathtt{start}, \mathtt{stop}), \forall k \in [0, \mathtt{pre}), \mathtt{sequence}[i][k] = \mathtt{sequence}[j][k].$
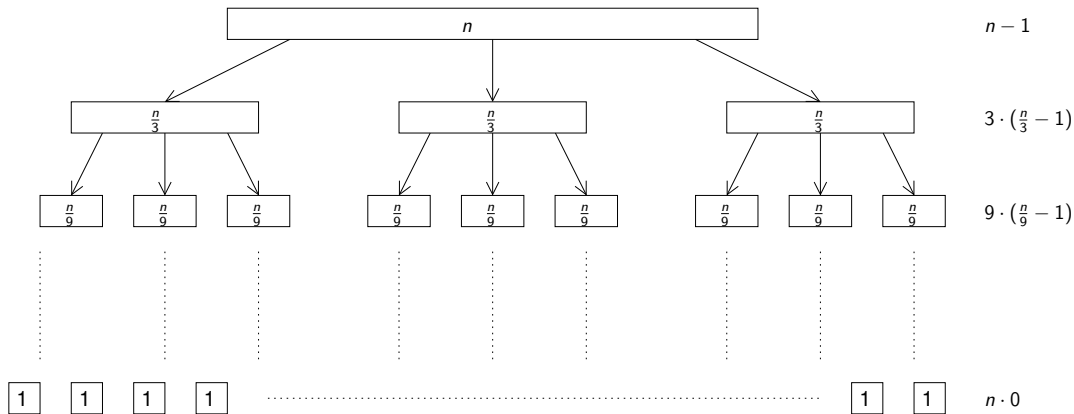
| dais | card | bark | care | even | barb | doze | cart | carb | axle | daze | exam | axis | bard | carp |

| dais | card | bark | care | even | barb | doze | cart | carb | axle | daze | exam | axis | bard | carp |

| barb | carb | card | bard | care | doze | axle | daze | bark | exam | even | carp | dais | axis | cart |

| exam | even | dais | axis | axle | barb | carb | card | bard | care | bark | carp | cart | doze | daze |

| dais | barb | carb | card | bard | care | bark | carp | cart | daze | doze | even | exam | axis | axle |

| axis | axle | barb | bard | bark | carb | card | care | carp | cart | dais | daze | doze | even | exam |

| beach | event | can | core | hope | any | front | ball | done | a | frond | an | i | give | eve |

| can | core | hope | any | ball | done | a | an | i | give | eve | frond | beach | event | front |

| can | any | a | an | i | eve | beach | core | hope | done | give | ball | frond | event | front |

| a | an | i | beach | eve | event | ball | can | done | frond | front | hope | core | give | any |

| a | i | ball | can | beach | give | an | any | done | hope | core | frond | front | eve | event |

| a | an | any | ball | beach | can | core | done | eve | event | frond | front | give | hope | i |

Coming up:

Do **Perfect hashing** *project (due Mon, Apr 28)*

Due **Fri, Apr 25** *)*
Read Section 8.1
Do Exercises 8.(4 & 5)
Take last quiz

Due **Mon, Apr 28**
Read Section 8.2
(No quiz or practice problems)