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:

- End-of-semester business
- Sorting strings
  - Why we care about strings
  - String quick sort
  - String bucket sort
  - String radix sort
Projects:

- Last regular project score update on Tues, Apr 26
- “Two minute warning” run of scripts on Fri, Apr 29 (no Schoology update—see report file)
- All projects due on the last day of classes, midnight between Fri, Apr 29 and Sat, Apr 30—not last day of finals.

Final exam

- Our final exam block is Tues, May 3, 10:30am–12:30pm
- The class will be split between the classroom and lab, same as with Test 2. If you want to switch, please ask.
- Test 3 (“written”/conceptual part) will be like Test 1, but covering graphs (ch 4) through strings (ch 8)
- Test 4 (programming part) will work the same way as Test 2, covering dynamic programming, hashing, and strings.
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;
}
```
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;
    }
}
struct employee
{
    char surname[20];
    char first_name[20];
    double salary;
    char extension[4];
};

struct book
{
    char title[100];
    char author[50];
    int pages;
    char call_number[8];
    int status;
};

struct complex_number { double real, double imag };
Quick sort:

\[ i, j \]

\[ \begin{array}{ccccccccccc}
\ldots & 91 & 88 & 44 & 62 & 56 & 33 & 59 & 31 & 59 & 53 & \ldots \\
\text{start} & & & & & & & & & & \text{stop} \\
\hline
\end{array} \]

\text{unsearched}

\text{Invariant 11 (Loop of partition())}

(a) \( \text{start} \leq i \leq j < \text{stop} \).

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

(c) \( \forall k \in [i, j), \text{sequence}[k] \geq \text{sequence}[\text{stop} - 1] \).

(d) \( j - \text{start} \) is the number of iterations completed.
Invariant 37. [Loop of `string_quick_sort_r()]

Let \( c \) be the character in position \( \text{pre} \) in the string in position \( \text{stop} - 1 \).

(a) \( \text{start} \leq i \leq j \leq k < \text{stop} \)

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

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

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

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

\[\begin{array}{cccccccccccccc}
\text{bark} & \text{barb} & \text{axle} & \text{axis} & \text{bard} & \text{card} & \text{care} & \text{cart} & \text{carb} & \text{carp} & \text{dais} & \text{even} & \text{doze} & \text{daze} & \text{exam} \\
\text{start} & & & & & & & & & & & & & \text{stop} \\
\hline
<\text{pivot} & & & & & =\text{pivot} & & & & & \geq\text{pivot} & & \\
\end{array}\]
Invariant 38. [Precondition of string_quick_sort_r()]
∀ i, j ∈ [start, stop), ∀k ∈ [0, pre), sequence[i][k] = sequence[j][k].
Coming up:

Do **Open Addressing Hashtable** project *(suggested by this past Monday, Apr 18)*
Do **Perfect Hashing** project *(suggested by Wednesday, Apr 27)*

**Due Fri, Apr 22 (end of day)**
Read section 8.1
Do exercises 8.(4 & 5) *(from Section 8.1)*

**Due Mon, Apr 25 (end of day)**
Do exercises 8.(4 & 5) *(from Section 8.1)*
Take quiz *(on Section 8.1)*
Read Section 8.2