Coding Rules

- Don't Repeat Yourself (DRY)
- Comments where needed
- Fail fast
- Avoid magic numbers
- One purpose for each variable
- Use good names
- Don't use global variables
- Return results, don't print them
- Use whitespace for readability

Testing and bugs

typical residual defect rates: bugs left over after the software has shipped

kloc: one thousand lines of source code

some testings are not good:

Exhaustive testing, Haphazard testing, and Random or statistical testing.

test cases must be chosen systematically
○ A ***test case*** is a particular choice of inputs, along with the expected output behavior required by the specification
○ A ***test suite*** is a set of test cases for an implementation

designing a test suite with three desirable properties:

Correct, Thorough, and Small.

Test First Programming:

In test first programming, you write tests before you even write any code.

The development of a single function proceeds in this order:

1. Spec : Write a specification for the function
2. Test : Write tests that exercise the specification
3. Implement : Write the actual code

The specification (spec)

The specification (spec) describes the input and output behavior of the function.
In code, the specification consists of the ***method signature*** and the ***comment above it that describes what it does***.

In-Class Quiz 1

- You want to partition the input space of this integer square root function:

```
/**
 * @param x is a nonnegative integer
 * @return nearest integer to the square root of x
 */
public static int intSqrt(int x)
```

- Which one is a **good partition**?
  - Partition: `x < 0` and `x >= 0`
  - Partition: `x` is a perfect square and `x` is an integer `> 0` but not a perfect square
  - Partition: `x = 0, x = 1, x = 5, x = 16`
  - Partition: `x` even, `x` odd, `x >= 100`

In-Class Quiz 2

- Consider the following function and values:

```
/**
 * @param winsAndLosses is a string of length at most 5 consisting of 'W' or 'L' characters
 * @return the fraction of characters in winsAndLosses that are 'W'
 */
public static double winLossRatio(String winsAndLosses)
```

(i) `""` (ii) `"LLLLL"` (iii) `"WLWL"` (iv) `"WWWWW"` (v) `"xxxxx"`

- Which are appropriate **boundary values** for testing this function ?
  - (i)
  - (ii), (iv), (v)
  - (i), (ii), (iv)
  - (i), (ii), (iv), (v)