CSE 331
Software Design & Implementation

Winter 2023

Section 3 – HW4, Abstract Data Types, and JUnit

Administrivia

- HW3 due today (1/19) at 11PM!
- HW4 due next Thursday at 11PM
  - This one takes a while, so get an early start!

Agenda

- Overview of HW4
- Quick review of polynomial arithmetic
- Unit testing with JUnit – an initial tour for HW4
- Review abstract data types (ADTs) by example

Abstract Data Types (ADTs)

- Abstraction representing some set of data
  - Meant to express the meaning/concept behind some Java class and the operations on it

- Different from implementation/instance fields!
  - Same ADT can have many different implementations
  - For instance, we can store the same point as an **`(x,y)`** or **`(r, theta`**
    - Both can be used to calculate the distance from origin, or create a line, etc.

- Stay tuned for more details in lecture tomorrow…

HW4 – Polynomial calculator

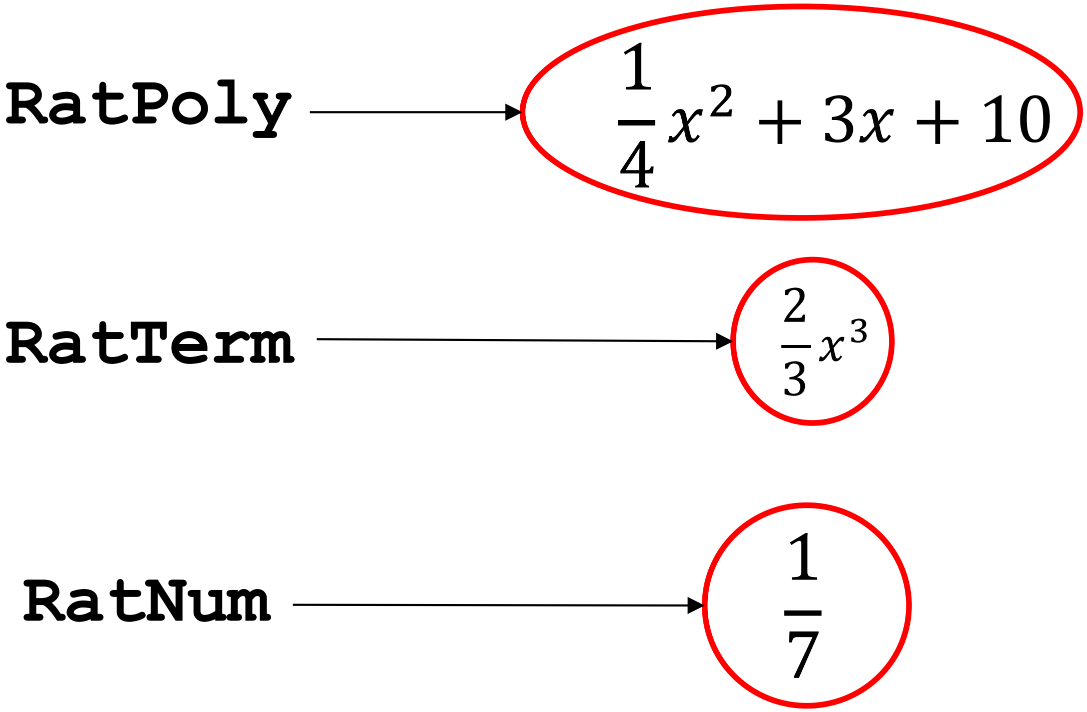
A homework in 6 parts:

0. Pseudocode algorithms for polynomial arithmetic
1. Conceptual questions about **`RatNum`**
2. Implement **`RatTerm`**
3. Implement **`RatPoly`**
4. Implement **`RatPolyStack`**
5. Try out your finished calculator!
6. Run your code against our tests to make sure it works!

The RatThings

- **`RatNum`** ADT
  - A rational number
  - Also includes a NaN (“not a number”) value
- **`RatTerm`** ADT
  - A polynomial term (rational coefficient w/ integer degree)
- **`RatPoly`** ADT
  - A polynomial expression (sum of polynomial terms)
- **`RatPolyStack`** ADT
  - An ordered collection of polynomial expressions

The RatThings

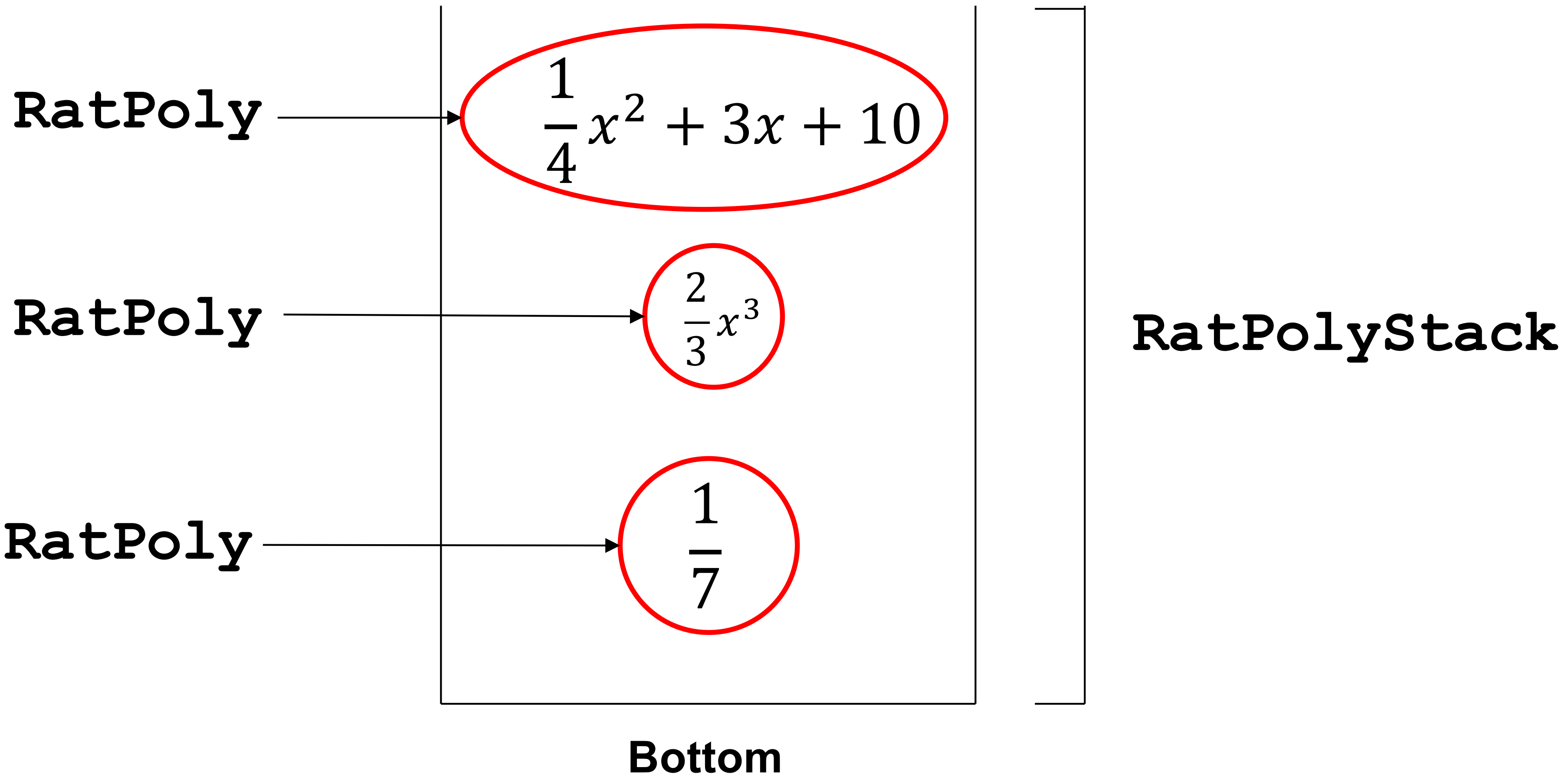


RatPoly → $\frac{1}{4}x^2 + 3x + 10$

RatTerm → $\frac{2}{3}x^3$

RatNum → $\frac{1}{7}$

The RatThings



RatPoly

RatPoly

RatPoly

$\frac{1}{4}x^2 + 3x + 10$

$\frac{2}{3}x^3$

$\frac{1}{7}$

RatPolyStack

Bottom

Polynomial arithmetic

Review arithmetic operations over polynomial expressions:

1. Addition
2. Subtraction
3. Multiplication
4. Division

Polynomial addition

$$(5x^4 + 4x^3 - x^2 + 5) + (3x^5 - 2x^3 + x - 5)$$

Polynomial addition

$$(5x^4 + 4x^3 - x^2 + 5) + (3x^5 - 2x^3 + x - 5)$$

$$\begin{array}{rrrrrrr} & & 5x^4 & +\,4x^3 & -\,1x^2 & & +\,5 \\ + & 3x^5 & & -\,2x^3 & & +\,1x & -\,5 \\ \hline \end{array}$$

Polynomial addition

$$(5x^4 + 4x^3 - x^2 + 5) + (3x^5 - 2x^3 + x - 5)$$

$$\begin{array}{lccccccccccc} & 0x^5 & + & 5x^4 & + & 4x^3 & - & 1x^2 & + & 0x & + & 5 \\ + & 3x^5 & + & 0x^4 & - & 2x^3 & + & 0x^2 & + & 1x & - & 5 \\ \hline \end{array}$$

Polynomial addition

$$(5x^4 + 4x^3 - x^2 + 5) + (3x^5 - 2x^3 + x - 5)$$

$$
\begin{array}{rrrrrrrrrrrrr}
 & 0x^5 & + & 5x^4 & + & 4x^3 & - & 1x^2 & + & 0x & + & 5 \\
+ & 3x^5 & + & 0x^4 & - & 2x^3 & + & 0x^2 & + & 1x & - & 5 \\
\hline
 & 3x^5 & + & 5x^4 & + & 2x^3 & - & 1x^2 & + & 1x & + & 0
\end{array}
$$

Polynomial subtraction

$$(5x^4 + 4x^3 - x^2 + 5) - (3x^5 - 2x^3 + x - 5)$$

Polynomial subtraction

$$(5x^4 + 4x^3 - x^2 + 5) - (3x^5 - 2x^3 + x - 5)$$

$$\begin{array}{lrrrrrr} & & 5x^4 & +\ 4x^3 & -\ 1x^2 & & +\ 5 \\ - & 3x^5 & & -\ 2x^3 & & +\ 1x & -\ 5 \\ \hline \end{array}$$

Polynomial subtraction

$$(5x^4 + 4x^3 - x^2 + 5) - (3x^5 - 2x^3 + x - 5)$$

$$\begin{array}{lcccccc}
 & 0x^5 + & 5x^4 & + \; 4x^3 & - \; 1x^2 & + \; 0x & + \; 5 \\
- & 3x^5 + & 0x^4 & - \; 2x^3 & + \; 0x^2 & + \; 1x & - \; 5 \\
\hline
\end{array}$$

Polynomial subtraction

$$(5x^4 + 4x^3 - x^2 + 5) - (3x^5 - 2x^3 + x - 5)$$

$$\begin{array}{rrrrrrr}
 & 0x^5 & + 5x^4 & + 4x^3 & - 1x^2 & + 0x & + 5 \\
- & 3x^5 & + 0x^4 & - 2x^3 & + 0x^2 & + 1x & - 5 \\
\hline
 & -3x^5 & + 5x^4 & + 6x^3 & - 1x^2 & - 1x & + 10
\end{array}$$

Polynomial multiplication

$$(4x^3 - x^2 + 5) \times (x - 5)$$

Polynomial multiplication

$$(4x^3 - x^2 + 5) \times (x - 5)$$

$$\begin{array}{rrrrr} & 4x^3 & -\ 1x^2 & & +\ 5 \\ \times & & & 1x & -\ 5 \\ \hline \end{array}$$

Polynomial multiplication

$(4x^3 - x^2 + 5) \times (x - 5)$

$$\begin{array}{rrrrr} & 4x^3 & -\ 1x^2 & & +\ 5 \\ \times & & & 1x & -\ 5 \\ \hline & -20x^3 & +\ 5x^2 & & -\ 25 \end{array}$$

Polynomial multiplication

$$(4x^3 - x^2 + 5) \times (x - 5)$$

$$\begin{array}{rrrrrr}
 & & 4x^3 & -\ 1x^2 & & +\ 5 \\
\times & & & & 1x & -\ 5 \\
\hline
 & & -20x^3 & +\ 5x^2 & & -\ 25 \\
 & 4x^4 & -\ 1x^3 & & +\ 5x &
\end{array}$$

Polynomial multiplication

$$(4x^3 - x^2 + 5) \times (x - 5)$$

$$\begin{array}{lrrrrr}
 & & 4x^3 & -\ 1x^2 & & +\ 5 \\
\times & & & & 1x & -\ 5 \\
\hline
 & & -20x^3 & +\ 5x^2 & & -\ 25 \\
+ & 4x^4 & -\ 1x^3 & & +\ 5x & \\
\hline
 & 4x^4 & -21x^3 & +\ 5x^2 & +\ 5x & -\ 25
\end{array}$$

Polynomial division

$$(5x^6 + 4x^4 - x^3 + 5) / (x^3 - 2x - 5)$$

Polynomial division

$$(5x^6 + 4x^4 - x^3 + 5) \,/\, (x^3 - 2x - 5)$$

$$\begin{array}{r|l} 1x^3 \qquad -2x \quad -5 & \overline{\;5x^6 \qquad +4x^4 \quad -1x^3 \qquad +5\;} \end{array}$$

Polynomial division

$$(5x^6 + 4x^4 - x^3 + 5) / (x^3 - 2x - 5)$$

$$1x^3 + 0x^2 \;\; -2x \;\; -5 \,\overline{\big)\; 5x^6 \quad +0x^5 \quad +4x^4 \quad -1x^3 \quad +0x^2 \quad +0x \quad\quad +5}$$

Polynomial division

$$(5x^6 + 4x^4 - x^3 + 5) \,/\, (x^3 - 2x - 5)$$

$$\begin{array}{r|rrrrrrr} & & & & 5x^3 & & & \\ \cline{2-8} 1x^3 +0x^2 \;\; -2x \;\; -5 & 5x^6 & +0x^5 & +4x^4 & -1x^3 & +0x^2 & +0x & +5 \end{array}$$

Polynomial division

$$(5x^6 + 4x^4 - x^3 + 5) \,/\, (x^3 - 2x - 5)$$

$$\begin{array}{r|rrrrrrr}
 & & & & 5x^3 & & & \\ \cline{2-8}
1x^3 +0x^2 \;\; -2x \;\; -5 & 5x^6 & +0x^5 & +4x^4 & -1x^3 & +0x^2 & +0x & +5 \\
 & 5x^6 & +0x^5 & -10x^4 & -25x^3 & & &
\end{array}$$

Polynomial division

$$(5x^6 + 4x^4 - x^3 + 5) \;/\; (x^3 - 2x - 5)$$

$$\begin{array}{r|rrrrrrr}
\multicolumn{1}{r}{} & & & & 5x^3 & & & \\ \cline{2-8}
1x^3 \; +0x^2 \;\; -2x \;\; -5 & 5x^6 & +0x^5 & +4x^4 & -1x^3 & +0x^2 & +0x & +5 \\
\multicolumn{1}{r}{-} & 5x^6 & +0x^5 & -10x^4 & -25x^3 & & & \\ \cline{2-5}
\multicolumn{1}{r}{} & 0x^6 & +0x^5 & +14x^4 & +24x^3 & & &
\end{array}$$

Notice (quotient * divisor) + remainder is always equal to $(5x^6 + 4x^4 – x^3 + 5)$.

Hint: this can help us determine the invariant

Polynomial division

$(5x^6 + 4x^4 - x^3 + 5) / (x^3 - 2x - 5)$

$$\begin{array}{r|rrrrrrr}
 & & & & 5x^3 & & & \\ \hline
1x^3 +0x^2 \;\; -2x \;\; -5 & 5x^6 & +0x^5 & +4x^4 & -1x^3 & +0x^2 & +0x & +5 \\
- & 5x^6 & +0x^5 & -10x^4 & -25x^3 & & & \\ \cline{2-5}
 & 0x^6 & +0x^5 & +14x^4 & +24x^3 & & &
\end{array}$$

Polynomial division

$$(5x^6 + 4x^4 - x^3 + 5) \,/\, (x^3 - 2x - 5)$$

$$
\begin{array}{r|rrrrrrr}
 & & & & 5x^3 & & & \\ \cline{2-8}
1x^3 +0x^2 \;-2x \;-5 & 5x^6 & +0x^5 & +4x^4 & -1x^3 & +0x^2 & +0x & +5 \\
- & 5x^6 & +0x^5 & -10x^4 & -25x^3 & & & \\ \cline{2-5}
 & 0x^6 & +0x^5 & +14x^4 & +24x^3 & +0x^2 & &
\end{array}
$$

Polynomial division

$$(5x^6 + 4x^4 - x^3 + 5) / (x^3 - 2x - 5)$$

$$
\begin{array}{r|rrrrrrr}
 & & & & 5x^3 & +0x^2 & & \\ \cline{2-8}
1x^3 +0x^2 \;\; -2x \;\; -5 & 5x^6 & +0x^5 & +4x^4 & -1x^3 & +0x^2 & +0x & +5 \\
- & 5x^6 & +0x^5 & -10x^4 & -25x^3 & & & \\ \cline{2-5}
 & 0x^6 & +0x^5 & +14x^4 & +24x^3 & +0x^2 & &
\end{array}
$$

Polynomial division

$(5x^6 + 4x^4 - x^3 + 5)$ / $(x^3 - 2x - 5)$

$$\begin{array}{r|rrrrrrr}
 & & & & 5x^3 & +0x^2 & & \\ \cline{2-8}
1x^3 \; +0x^2 \; -2x \; -5 & 5x^6 & +0x^5 & +4x^4 & -1x^3 & +0x^2 & +0x & +5 \\
- & 5x^6 & +0x^5 & -10x^4 & -25x^3 & & & \\ \cline{2-5}
 & 0x^6 & +0x^5 & +14x^4 & +24x^3 & +0x^2 & +0x &
\end{array}$$

Polynomial division

$$(5x^6 + 4x^4 - x^3 + 5) / (x^3 - 2x - 5)$$

$$\begin{array}{r|rrrrrrr}
 & & & & 5x^3 & +0x^2 & +14x & \\ \hline
1x^3 +0x^2 \; -2x \; -5 & 5x^6 & +0x^5 & +4x^4 & -1x^3 & +0x^2 & +0x & +5 \\
- & 5x^6 & +0x^5 & -10x^4 & -25x^3 & & & \\ \hline
 & 0x^6 & +0x^5 & +14x^4 & +24x^3 & +0x^2 & +0x &
\end{array}$$

Polynomial division

$$(5x^6 + 4x^4 - x^3 + 5) / (x^3 - 2x - 5)$$

$$\begin{array}{r|rrrrrrr}
 & & & & 5x^3 & +0x^2 & +14x & \\ \cline{2-8}
1x^3 +0x^2 \;\; -2x \;\; -5 & 5x^6 & +0x^5 & +4x^4 & -1x^3 & +0x^2 & +0x & +5 \\
- & 5x^6 & +0x^5 & -10x^4 & -25x^3 & & & \\ \cline{2-5}
 & 0x^6 & +0x^5 & +14x^4 & +24x^3 & +0x^2 & +0x & \\
 & & & 14x^4 & +0x^3 & -28x^2 & -70x &
\end{array}$$

Polynomial division

$$(5x^6 + 4x^4 - x^3 + 5) / (x^3 - 2x - 5)$$

$$\begin{array}{r|rrrrrrr}
 & & & & 5x^3 & +0x^2 & +14x & \\ \cline{2-8}
1x^3 +0x^2 \; -2x \; -5 & 5x^6 & +0x^5 & +4x^4 & -1x^3 & +0x^2 & +0x & +5 \\
- & 5x^6 & +0x^5 & -10x^4 & -25x^3 & & & \\ \cline{2-5}
 & 0x^6 & +0x^5 & +14x^4 & +24x^3 & +0x^2 & +0x & \\
 & & - & 14x^4 & +0x^3 & -28x^2 & -70x & \\ \cline{4-7}
 & & & 0x^4 & +24x^3 & +28x^2 & +70x &
\end{array}$$

Polynomial division

$$(5x^6 + 4x^4 - x^3 + 5) \;/\; (x^3 - 2x - 5)$$

$$\begin{array}{r|rrrrrrr}
\multicolumn{1}{r}{} & & & & 5x^3 & +0x^2 & +14x & \\ \cline{2-8}
1x^3 +0x^2 \;\; -2x \;\; -5 & 5x^6 & +0x^5 & +4x^4 & -1x^3 & +0x^2 & +0x & +5 \\
\multicolumn{1}{r}{-} & 5x^6 & +0x^5 & -10x^4 & -25x^3 & & & \\ \cline{2-5}
\multicolumn{1}{r}{} & 0x^6 & +0x^5 & +14x^4 & +24x^3 & +0x^2 & +0x & \\
\multicolumn{1}{r}{} & & - & 14x^4 & +0x^3 & -28x^2 & -70x & \\ \cline{3-7}
\multicolumn{1}{r}{} & & & 0x^4 & +24x^3 & +28x^2 & +70x &
\end{array}$$

Polynomial division

$$(5x^6 + 4x^4 - x^3 + 5) / (x^3 - 2x - 5)$$

$$
\begin{array}{rr|rrrrrrr}
 & & & & & 5x^3 & +0x^2 & +14x & \\
\cline{3-9}
1x^3 +0x^2 \;\; -2x \;\; -5 & & 5x^6 & +0x^5 & +4x^4 & -1x^3 & +0x^2 & +0x & +5 \\
 & - & 5x^6 & +0x^5 & -10x^4 & -25x^3 & & & \\
\cline{3-6}
 & & 0x^6 & +0x^5 & +14x^4 & +24x^3 & +0x^2 & +0x & \\
 & & & - & 14x^4 & +0x^3 & -28x^2 & -70x & \\
\cline{5-8}
 & & & & 0x^4 & +24x^3 & +28x^2 & +70x & +5
\end{array}
$$

Polynomial division

$$(5x^6 + 4x^4 - x^3 + 5) / (x^3 - 2x - 5)$$

$$\begin{array}{r|rrrrrrr}
 & & & & 5x^3 & +0x^2 & +14x & +24 \\ \cline{2-8}
1x^3 +0x^2 \;\; -2x \;\; -5 & 5x^6 & +0x^5 & +4x^4 & -1x^3 & +0x^2 & +0x & +5 \\
- & 5x^6 & +0x^5 & -10x^4 & -25x^3 & & & \\ \cline{2-5}
 & 0x^6 & +0x^5 & +14x^4 & +24x^3 & +0x^2 & +0x & \\
 & & - & 14x^4 & +0x^3 & -28x^2 & -70x & \\ \cline{4-7}
 & & & 0x^4 & +24x^3 & +28x^2 & +70x & +5
\end{array}$$

Polynomial division

$$(5x^6 + 4x^4 - x^3 + 5) \,/\, (x^3 - 2x - 5)$$

$$\begin{array}{r|rrrrrrr}
 & & & & 5x^3 & +0x^2 & +14x & +24 \\
\cline{2-8}
1x^3 +0x^2 \;\; -2x \;\; -5 & 5x^6 & +0x^5 & +4x^4 & -1x^3 & +0x^2 & +0x & +5 \\
- & 5x^6 & +0x^5 & -10x^4 & -25x^3 & & & \\
\cline{2-5}
 & 0x^6 & +0x^5 & +14x^4 & +24x^3 & +0x^2 & +0x & \\
 & & - & 14x^4 & +0x^3 & -28x^2 & -70x & \\
\cline{4-7}
 & & & 0x^4 & +24x^3 & +28x^2 & +70x & +5 \\
 & & & & 24x^3 & +0x^2 & -48x & -120
\end{array}$$

Polynomial division

$$(5x^6 + 4x^4 - x^3 + 5) \; / \; (x^3 - 2x - 5)$$

$$\begin{array}{rrrrrrrr}
 & & & & 5x^3 & +0x^2 & +14x & +24 \\
\cline{2-8}
\multicolumn{1}{r|}{1x^3 +0x^2 \;\; -2x \;\; -5} & 5x^6 & +0x^5 & +4x^4 & -1x^3 & +0x^2 & +0x & +5 \\
- & 5x^6 & +0x^5 & -10x^4 & -25x^3 & & & \\
\cline{2-5}
 & 0x^6 & +0x^5 & +14x^4 & +24x^3 & +0x^2 & +0x & \\
 & & - & 14x^4 & +0x^3 & -28x^2 & -70x & \\
\cline{4-7}
 & & & 0x^4 & +24x^3 & +28x^2 & +70x & +5 \\
 & & & - & 24x^3 & +0x^2 & -48x & -120 \\
\cline{5-8}
 & & & & 0x^3 & +28x^2 & +118x & +125
\end{array}$$

Polynomial division

$$(5x^6 + 4x^4 - x^3 + 5) / (x^3 - 2x - 5)$$

$$
\begin{array}{rrrrrrrr}
 & & & & \multicolumn{4}{r}{\overbrace{5x^3 \quad +0x^2 \quad +14x \quad +24}^{\text{quotient}}} \\
\cline{2-8}
\multicolumn{1}{r|}{1x^3 +0x^2 \;\; -2x \;\; -5} & 5x^6 & +0x^5 & +4x^4 & -1x^3 & +0x^2 & +0x & +5 \\
- & 5x^6 & +0x^5 & -10x^4 & -25x^3 & & & \\
\cline{2-5}
 & 0x^6 & +0x^5 & +14x^4 & +24x^3 & +0x^2 & +0x & \\
 & & - & 14x^4 & +0x^3 & -28x^2 & -70x & \\
\cline{4-7}
 & & & 0x^4 & +24x^3 & +28x^2 & +70x & +5 \\
 & & & - & 24x^3 & +0x^2 & -48x & -120 \\
\cline{5-8}
 & & & & 0x^3 & \multicolumn{3}{r}{\underbrace{+28x^2 \quad +118x \quad +125}_{\text{remainder}}}
\end{array}
$$

Notice (quotient * divisor) + remainder is still equal to $(5x^6 + 4x^4 – x^3 + 5)$.

Polynomial division

$$(5x^6 + 4x^4 - x^3 + 5) / (x^3 - 2x - 5)$$

$$5x^3 + 14x + 24 + \frac{28x^2 + 118x + 125}{x^3 - 2x - 5}$$

What is a `final` variable in Java?

- Once assigned, it can never be reassigned.

- What is the difference between these two?

```
final int x = 42;

final List<Integer> y = new ArrayList<>();
```

- How does this relate to immutability?
  - **`x`** cannot change, **`y`** still can! (for instance, I can still do **`y.add(10)`**)
    - More precisely: **`y`** itself never changes – it always references the same ArrayList, but the ArrayList that it references can change, so the list is not immutable

HW4 Starter Code

Let’s look at the HW4 starter code, Javadoc, and tests…

Testing: A quick introduction

- For HW 4, you’ll be running our test suite to verify your RatThings work.

- Just know how it works; don’t need to know how to write tests (yet)!

JUnit

- Industry-standard Java toolkit for unit testing
  - We're using JUnit 4
  - Some other classes use JUnit 5—please make sure to use JUnit 4 and its syntax!
    - Biggest difference is testing to verify an exception is thrown
- A unit test is a test for one "component" by itself
  - "Component" typically a class or a method
- Each unit test written as a method
  - We'll see the particulars in a moment…
- Closely related unit tests should be grouped into a class
  - For example, all unit tests for the same ADT implementation

Tests in JUnit

A method annotated with `@Test` is flagged as a JUnit test

```
import org.junit.*;
import static org.junit.Assert.*;

/** Unit tests for my Foo ADT implementation */
public class FooTests {
  @Test
  public void testBar() {
    ... /* use JUnit assertions in here */
  }
}
```

Using JUnit assertions

- JUnit assertions establish success or failure of the test method
  - *Note*: JUnit assertions are *different* from Java's **`assert`** statement

- Use to check that an actual result matches the expected value
  - Example: **`assertEquals(42, meaningOfLife());`**
  - Example: **`assertTrue(list.isEmpty());`**

- A test method stops immediately after the first assertion failure
  - If no assertion fails, then the test method passes
  - Other test methods still run either way

- JUnit results show details of any test failures

Common JUnit assertions

JUnit’s documentation has a full list, but these are the most common assertions.

Assertion	Failure condition
`assertTrue(test)`	`test == false`
`assertFalse(test)`	`test == true`
`assertEquals(expected, actual)`	***expected*** and ***actual*** are not equal
`assertSame(expected, actual)`	`expected != actual`
`assertNotSame(expected, actual)`	`expected == actual`
`assertNull(value)`	`value != null`
`assertNotNull(value)`	`value == null`

Any JUnit assertion can also take a string to show in case of failure, *e.g.*, **`assertEquals(“helpful message”, expected, actual).`**

Checking for a thrown exception

- Should test that your code throws exceptions as specified
- This kind of test method fails if its body does *not* throw an exception of the named class
  - May not need any JUnit assertions inside the test method unlike our previous guideline

```
@Test(expected=IndexOutOfBoundsException.class)
public void testGetEmptyList() {
  List<String> list = new ArrayList<String>();
  list.get(0);
}
```

Note: This is different in JUnit 5, so make sure to use this syntax!

Test ordering, setup, clean-up

JUnit does not promise to run tests in any particular order.

However, JUnit can run helper methods for common setup/cleanup

- Run before/after *each* test method in the class:

```
@Before
public void m() { ... }
@After
public void m() { ... }
```

- Run once before/after running *all* test methods in the class:

```
@BeforeClass
public static void m() { ... }
@AfterClass
public static void m() { ... }
```

Junit Tests Example

Let’s look at some example Junit tests…

Abstract data types by example

Review ADT concepts through two examples:

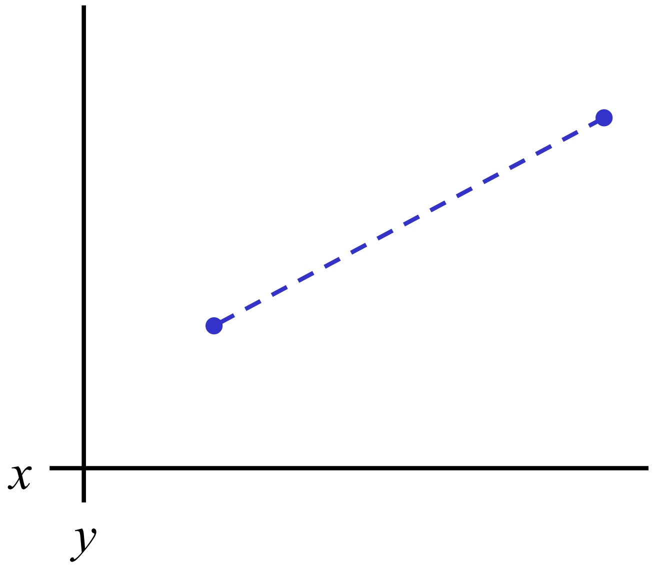
- A **`Line`** ADT
- A **`Circle`** ADT

On the course website, see “Resources” → “Class and Method Specifications” for a handy guide with full details.

We won’t cover abstraction functions today (see upcoming lecture).

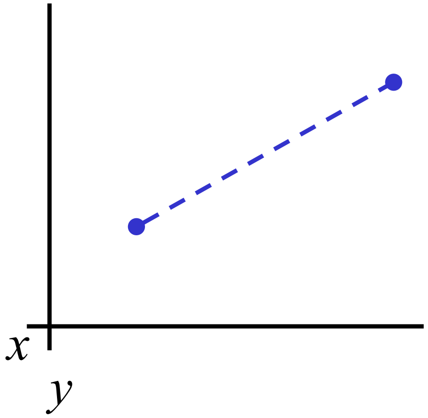
Line ADT

Concept: A line segment in the Cartesian co-ordinate plane



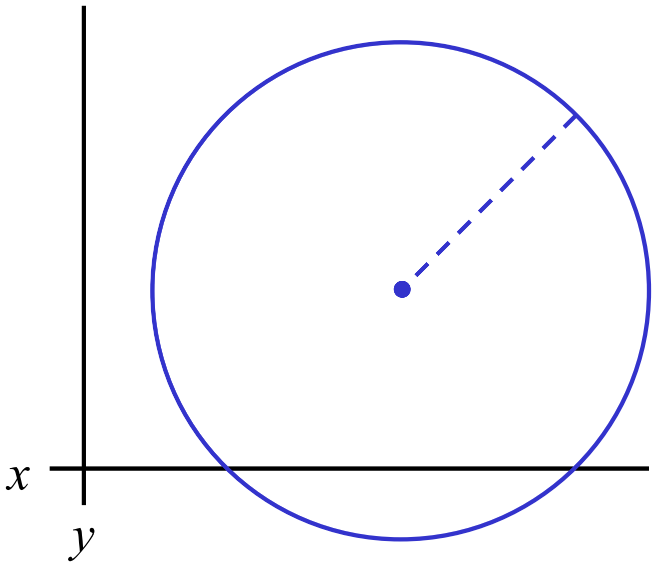
Line ADT: Class specification

```
/**
 * A Line is a mutable 2D line segment with endpoints
 * p1 and p2.
 */
public class Line {
  ... // rep invariant, fields, methods, etc.
}
```



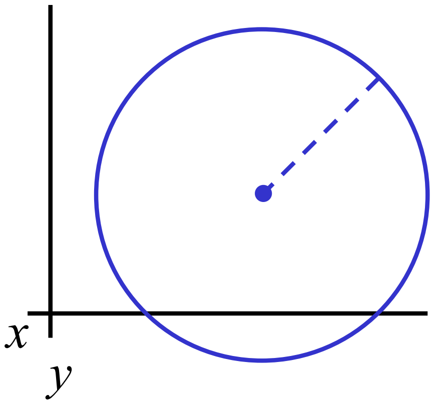
Circle ADT

Concept: A circle in the Cartesian co-ordinate plane



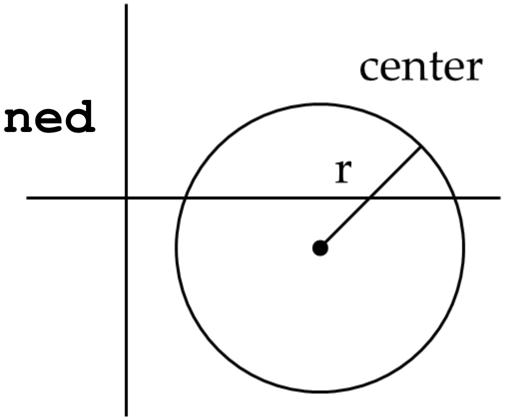
Circle ADT: Class specification

```
/**
 * A Circle is a mutable 2D circle, defined by a
 * center point p and radius r.
 */
public class Circle {
  ... // fields, rep invariant, methods, etc.
}
```



Circle ADT: Representation #1

```
/**
 * A Circle is a mutable 2D circle, defined
 * by a center point p and radius r.
 */
public class Circle {
  private Point center;
  private double radius;

  ...
}
```



Interlude: Representation invariant

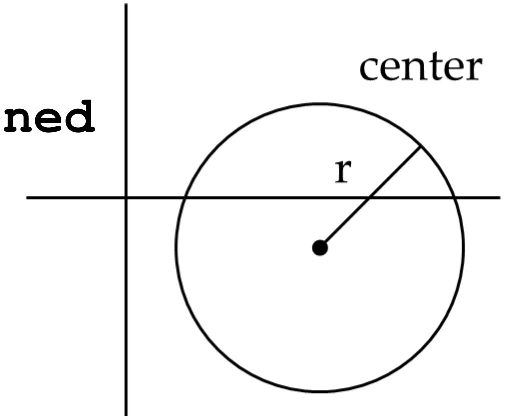
An ADT implementation has a representation invariant:

- Restricts concrete representation of the ADT
- Maps each object’s internal state to a boolean for validity

If the representation invariant is violated by (*i.e.*, false for) some object, that object is “broken.”

- The object doesn’t map to any abstract value
- Indicates a bug in the ADT implementation!

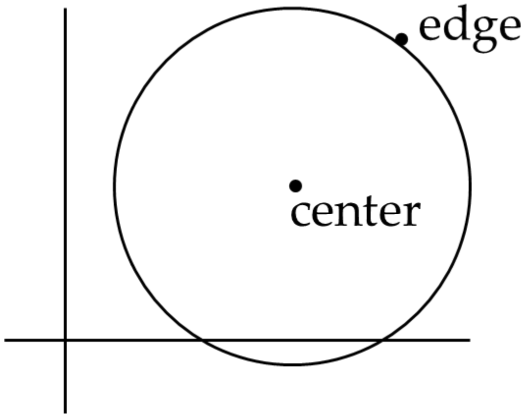
Circle ADT: Representation #1



```
/**
 * A Circle is a mutable 2D circle, defined
 * by a center point p and radius r.
 */
public class Circle {
  private Point center;
  private double radius;

  // Representation Invariant:
  //   center != null && radius > 0
  ...
}
```

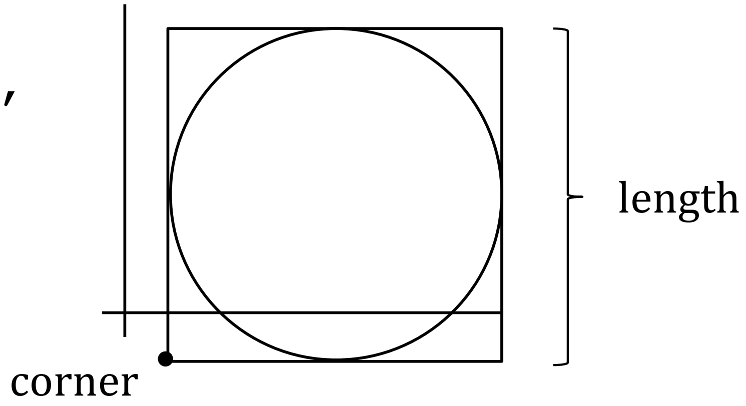
Circle ADT: Representation #2

```
/**
 * A Circle is a mutable 2D circle,
 * defined by a center point p and
 * radius r.
 */
public class Circle {
  private Point center;
  private Point edge;

  // Representation Invariant:
  //    center != null &&
  //    edge != null &&
  //    !center.equals(edge)
}
```



Circle ADT: Representation #3

```
/**
 * A Circle is a mutable 2D circle,
 * defined by a center point p and
 * radius r.
 */
public class Circle {
  private Point corner;
  private double length;

  // Representation Invariant:
  //    corner != null &&
  //    length > 0
}
```



Checking the representation invariant

The rep. invariant must hold before and after each public method.

Write and use a **`checkRep()`** method:

- Call at entry and exit of each public method
  - Only call at the exit of constructors
- Bug-finding value well worth the little extra code
- If slow to check, add code to conditionally do expensive checks when desired and omit when appropriate (more later with hw5, hw6, etc.)
- Much more about this in lectures

```
public void m(...) {
   checkRep();
   ...
   checkRep();
}
```

`checkRep()`

Do we still need to call our **`checkRep`** in every method if our object is immutable?

Yes! The fields of an immutable ADT can still change (even by accident!), and we need to ensure our rep invariant holds.

With one exception…if every field is strictly immutable **and `final`**, then we only need to call it at the end of constructor because immutability is guaranteed

```
public class A {
  final int x;
  // only constructor
}
```

```
public class B {
  final List<Integer> x;
  // everywhere
}
```

Try it yourself!

Write your own specification of a Rectangle ADT on the handout.

Then give two different possible representations for your Rectangle ADT and write checkRep functions for them