CSE 331
Software Design & Implementation

Section 3 – HW4, Abstract Data Types, and JUnit
Administrivia

• HW3 due tonight at 11 PM!
• HW2 due Monday at 5 PM!
• Any questions?
Agenda

• ADTs!

• Overview of HW4

• Quick review of polynomial arithmetic

• Abstraction functions

• Unit testing with Junit – an initial tour for HW4
Abstract Data Types (ADTs)

• Abstraction representing some set of data
  – Meant to express the meaning/concept behind some Java class

• Different from implementation/Java fields!
  – Same ADT can have many different implementations

• Any questions?
HW4 – Polynomial calculator

A homework in 6 parts:

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

Start early, and use your knowledge of invariants to unblock yourself.
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 $\rightarrow \frac{1}{4}x^2 + 3x + 10$

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

RatNum $\rightarrow \frac{1}{7}$
The RatThings

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

\[
\frac{2}{3} x^3
\]

\[
\frac{1}{7}
\]

RatPolyStack

Bottom
Polynomial arithmetic

Review arithmetic operations over polynomial expressions:

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

Defining and following invariants is critical to making sure that these operations are implemented correctly.
Polynomial addition

\[(5x^4 + 4x^3 - x^2 + 5) + (3x^5 - 2x^3 + x - 5)\]
Polynomial addition

\[ (5x^4 + 4x^3 - x^2 + 5) + (3x^5 - 2x^3 + x - 5) \]

\[ 5x^4 + 4x^3 - x^2 + 5 + 3x^5 - 2x^3 + x - 5 \]
Polynomial addition

\[(5x^4 + 4x^3 - x^2 + 5) + (3x^5 - 2x^3 + x - 5)\]

\[0x^5 5 + x^4 4 + x^3 1 - x^2 0 + 0x 0 + 5\]

\[+ 3x^5 + 0x^4 - 2x^3 0 + 0x^2 0 + 1x - 5\]

\[= 3x^5 + 4x^4 - x^2 + 6x - 10\]
Polynomial addition

\[(5x^4 + 4x^3 - x^2 + 5) + (3x^5 - 2x^3 + x - 5)\]

\[
\begin{align*}
0x^5 + 5x^4 & + 4x^3 - 1x^2 + 0x + 5 \\
+ & 3x^5 + 0x^4 - 2x^3 + 0x^2 + 1x - 5 \\
\hline
3x^5 + 5x^4 & + 2x^3 - 1x^2 + 1x + 0
\end{align*}
\]
Polynomial subtraction

\[(5x^4 + 4x^3 - x^2 + 5) - (3x^5 - 2x^3 + x - 5)\]
Polynomial subtraction

\[(5x^4 + 4x^3 - x^2 + 5) - (3x^5 - 2x^3 + x - 5)\]

\[
\begin{array}{c}
5x^4 + 4x^3 - 1x^2 + 5 \\
- 3x^5 - 2x^3 + 1x - 5 \\
\hline
\end{array}
\]
Polynomial subtraction

\[(5x^4 + 4x^3 - x^2 + 5) - (3x^5 - 2x^3 + x - 5)\]

\[0x^5 + 5x^4 + 4x^3 - 1x^2 + 0x + 5\]

\[- 3x^5 + 0x^4 - 2x^3 + 0x^2 + 1x - 5\]
Polynomial subtraction

\[(5x^4 + 4x^3 - x^2 + 5) - (3x^5 - 2x^3 + x - 5)\]

\[
\begin{align*}
0x^5 & + 5x^4 & + 4x^3 & - 1x^2 & + 0x & + 5 \\
- & 3x^5 & + 0x^4 & - 2x^3 & + 0x^2 & + 1x & - 5 \\
\hline
-3x^5 & + 5x^4 & + 6x^3 & - 1x^2 & - 1x & + 10
\end{align*}
\]
Polynomial multiplication

\[(4x^3 - x^2 + 5) \times (x - 5)\]
Polynomial multiplication

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

\[
\begin{array}{cccc}
4x^3 & - & 1x^2 & + 5 \\
\times & & & \\
1x & - & 5 \\
\end{array}
\]
Polynomial multiplication

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

\[
\begin{array}{ccc}
4x^3 & - & 1x^2 \\
\times & & + 5 \\
\hline \\
-20x^3 & + & 5x^2 \\
& & -25
\end{array}
\]
Polynomial multiplication

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

\[
\begin{array}{c}
4x^3 - 1x^2 \\
\times \\
1x - 5
\end{array}
\]

\[
\begin{array}{c}
-20x^3 + 5x^2 \\
4x^4 - 1x^3 \\
+ 5x
\end{array}
\]

\[
\begin{array}{c}
-20x^3 + 5x^2 \\
4x^4 - 1x^3 \\
+ 5x
\end{array}
\]
Polynomial multiplication

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

\[
\begin{array}{c|c|c}
4x^3 & -1x^2 & +5 \\
\times & 1x & -5 \\
\hline
-20x^3 & +5x^2 & -25 \\
+ & 4x^4 & -x^3 & +5x \\
\hline
4x^4 & -21x^3 & +5x^2 & +5x & -25 \\
\end{array}
\]
Polynomial division

\[(5x^6 + 4x^4 - x^3 + 5) / (x^3 - 2x - 5)\]
Polynomial division

\[(5x^6 + 4x^4 - x^3 + 5) / (x^3 - 2x - 5)\]

\[
\begin{array}{c|cc}
1x^3 & -2x & -5 \\
\hline
5x^6 & +4x^4 & -1x^3 & +5
\end{array}
\]
Polynomial division

\[
(5x^6 + 4x^4 - x^3 + 5) \div (x^3 - 2x - 5)
\]
Polynomial division

\[(5x^6 + 4x^4 - x^3 + 5) / (x^3 - 2x - 5)\]
Polynomial division

\[
\frac{(5x^6 + 4x^4 - x^3 + 5)}{(x^3 - 2x - 5)}
\]

\[
\begin{array}{cccccc}
1x^3 & +0x^2 & -2x & -5 & & \\
\hline
5x^6 & +0x^5 & +4x^4 & -1x^3 & +0x^2 & +0x & +5 \\
5x^6 & +0x^5 & -10x^4 & -25x^3 & & & \\
\end{array}
\]
Polynomial division

\[(5x^6 + 4x^4 - x^3 + 5) / (x^3 - 2x - 5)\]

\[
\begin{array}{ccccccc}
1x^3 + 0x^2 & -2x & -5 & & & & \\
\hline
5x^6 & +0x^5 & +4x^4 & -1x^3 & +0x^2 & +0x & +5 \\
-5x^6 & +0x^5 & -10x^4 & -25x^3 & & & \\
\hline
0x^6 & +0x^5 & +14x^4 & +24x^3 & & & \\
\end{array}
\]

Notice \((\text{quotient} \times \text{divisor}) + \text{remainder}\) is always equal to \((5x^6 + 4x^4 - x^3 + 5)\).

We can use this fact to produce an invariant.
Polynomial division

\[
(5x^6 + 4x^4 - x^3 + 5) \div (x^3 - 2x - 5)
\]

\[
\begin{array}{rrrrrr}
1x^3+0x^2 & -2x & -5 & \multicolumn{5}{c}{5x^3} \\
\underline{5x^6} & \underline{+0x^5} & \underline{+4x^4} & -1x^3 & +0x^2 & +0x & +5 \\
-5x^6 & +0x^5 & -10x^4 & -25x^3 \\
\hline
0x^6 & +0x^5 & +14x^4 & +24x^3
\end{array}
\]
Polynomial division

\[(5x^6 + 4x^4 - x^3 + 5) / (x^3 - 2x - 5)\]
Polynomial division

\[
\frac{5x^6 + 4x^4 - x^3 + 5}{x^3 - 2x - 5}
\]
Polynomial division

\[(5x^6 + 4x^4 - x^3 + 5) / (x^3 - 2x - 5)\]
Polynomial division

\[
\frac{(5x^6 + 4x^4 - x^3 + 5)}{(x^3 - 2x - 5)}
\]

\[
\begin{array}{cccccc}
1x^3 & +0x^2 & -2x & -5 & \underline{5x^3} & +0x^2 & +14x \\
\hline
5x^6 & +0x^5 & +4x^4 & -1x^3 & +0x^2 & +0x & +5 \\
-5x^6 & +0x^5 & -10x^4 & -25x^3 & & & \\
0x^6 & +0x^5 & +14x^4 & +24x^3 & +0x^2 & +0x \\
\end{array}
\]
Polynomial division

\[(5x^6 + 4x^4 - x^3 + 5) / (x^3 - 2x - 5)\]
Polynomial division

\[
(5x^6 + 4x^4 - x^3 + 5) \div (x^3 - 2x - 5)
\]

\[
\begin{array}{c|ccccc}
& 5x^3 & +0x^2 & +14x & +5 \\
\hline
1x^3 +0x^2 & -2x & -5 \\
\hline
5x^6 & +0x^5 & +4x^4 & -1x^3 & +0x^2 & +0x & +5 \\
-5x^6 & 0x^5 & -10x^4 & -25x^3 \\
\hline
0x^6 & +0x^5 & +14x^4 & +24x^3 & +0x^2 & +0x \\
-14x^4 & +0x^3 & -28x^2 & -70x \\
\hline
0x^4 & +24x^3 & +28x^2 & +70x
\end{array}
\]
Polynomial division

\[
\frac{(5x^6 + 4x^4 - x^3 + 5)}{(x^3 - 2x - 5)}
\]

\[
\begin{array}{c|cccc}
& 5x^3 & +0x^2 & +14x & +5 \\
1x^3 + 0x^2 - 2x - 5 & 5x^6 & +0x^5 & +4x^4 & -1x^3 & +0x^2 & +0x & +5 \\
- & 5x^6 & +0x^5 & -10x^4 & -25x^3 & & & \\
\hline
0x^6 & +0x^5 & +14x^4 & +24x^3 & +0x^2 & +0x & & \\
- & 14x^4 & +0x^3 & -28x^2 & -70x & & & \\
\hline
0x^4 & +24x^3 & +28x^2 & +70x & & & & \\
\end{array}
\]
Polynomial division

\[(5x^6 + 4x^4 - x^3 + 5) \div (x^3 - 2x - 5)\]

\[
\begin{array}{c|ccccc}
1x^3 + 0x^2 & -2x & -5 & & & \\
\hline
5x^6 & +0x^5 & +4x^4 & -1x^3 & +0x^2 & +0x & +5 \\
-5x^6 & +0x^5 & -10x^4 & -25x^3 & & & \\
\hline
0x^6 & +0x^5 & +14x^4 & +24x^3 & +0x^2 & +0x & \\
-14x^4 & +0x^3 & -28x^2 & -70x & & & \\
\hline
0x^4 & +24x^3 & +28x^2 & +70x & +5 & & \\
\end{array}
\]
Polynomial division

\[(5x^6 + 4x^4 - x^3 + 5) \div (x^3 - 2x - 5)\]

\[
\begin{array}{r|rrrr}
     & 5x^3 & +0x^2 & +14x & +24 \\
1x^3 + 0x^2 -2x -5 & \hline
5x^6 & +0x^5 & +4x^4 & -1x^3 & +0x^2 & +0x & +5 \\
-5x^6 & +0x^5 & -10x^4 & -25x^3 \\
0x^6 & +0x^5 & +14x^4 & +24x^3 & +0x^2 & +0x \\
-14x^4 & +0x^3 & -28x^2 & -70x \\
0x^4 & +24x^3 & +28x^2 & +70x & +5 \\
\end{array}
\]
Polynomial division

\[(5x^6 + 4x^4 - x^3 + 5) / (x^3 - 2x - 5)\]
Polynomial division

\((5x^6 + 4x^4 - x^3 + 5) \div (x^3 - 2x - 5)\)

\[
\begin{array}{c|cccc}
1x^3 + 0x^2 & -2x & -5 & & \\
\hline
& 5x^6 & +0x^5 & +4x^4 & -1x^3 & +0x^2 & +0x & +5 \\
- & 5x^6 & +0x^5 & -10x^4 & -25x^3 & & & \\
\hline
& 0x^6 & +0x^5 & +14x^4 & +24x^3 & +0x^2 & +0x & \\
& - & 14x^4 & +0x^3 & -28x^2 & -70x & & \\
\hline
& 0x^4 & +24x^3 & +28x^2 & +70x & +5 & & \\
& - & 24x^3 & +0x^2 & -48x & -120 & & \\
\hline
& 0x^3 & +28x^2 & +118x & +125 & & & 
\end{array}
\]
Polynomial division

\[(5x^6 + 4x^4 - x^3 + 5) \div (x^3 - 2x - 5)\]
Polynomial division

\[
(5x^6 + 4x^4 - x^3 + 5) \div (x^3 - 2x - 5)
\]

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

Notice that the loop invariant, \( q \cdot y + r = x \) and \( 0 \leq r \) where \( q \) is the quotient, \( y \) is the divisor, \( r \) is the remainder and \( x \) is the polynomial that is being divided is always correct after each subtraction step.
Abstraction Functions (AFs)

• Let’s say we have an ADT
  – And we choose some way to implement it

• How does the concrete implementation relate to our ADT?

• This is an **abstraction function**
  – Maps object implementation (our Java fields) to the abstract state
  – Ex: “How does a Triangle object from Triangle.java represent a Triangle ADT?”
  – Note: specific to implementation

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

ADT specification

Abstract States

Abstraction Barrier

Fields in our Java class

Abstraction function (AF): Relationship between ADT specification and implementation
Line ADT

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

How might we implement this?
Line ADT: Representation #1

/**
 * A Line is a mutable 2D line segment with endpoints
 * p1 and p2.
 */

public class Line {

    private int x1, x2;
    private int y1, y2;

}

What is our abstraction function?
Line ADT: Representation #1

/**
 * A Line is a mutable 2D line segment with endpoints p1 and p2.
 */

public class Line {
    // Abstract state is line with endpoints (x1, y1) and (x2, y2)
    private int x1, x2;
    private int y1, y2;
}

\[ x \]
\[ y \]
Line ADT: Representation #2

/**
 * A Line is a mutable 2D line segment with endpoints p1 and p2.
 */
public class Line {
    private Point pointA, pointB;
}

What is our abstraction function?
Line ADT: Representation #2

/**
 * A Line is a mutable 2D line segment with endpoints p1 and p2.
 */
public class Line {
    // Abstract state is line with endpoints p1 and p2
    private Point pointA, pointB;
}

Does this representation have any advantages?
Line ADT: Representation #3

/**
 * A Line is a mutable 2D line segment with endpoints
 * p1 and p2.
 */
public class Line {

    private int x1, y1;
    private double angle;
    private double len;

}
/**
 * A Line is a mutable 2D line segment with endpoints p1 and p2.
 */

public class Line {
    // Abstract state is line with endpoints (x1, y1) and
    // (x1 + len * cos(angle), y1 + len * sin(angle))
    private int x1, y1;
    private double angle;
    private double len;
}

Does this representation have any advantages?
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 abstraction functions for them.
Testing: A quick introduction

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

• Let’s do a quick walkthrough of our test suite
  – 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

• 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
Writing tests with JUnit

Annotate a method with `@Test` to flag it as a JUnit test

```java
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.

<table>
<thead>
<tr>
<th>Assertion</th>
<th>Failure condition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>assertTrue(test)</code></td>
<td><code>test == false</code></td>
</tr>
<tr>
<td><code>assertFalse(test)</code></td>
<td><code>test == true</code></td>
</tr>
<tr>
<td><code>assertEquals(expected, actual)</code></td>
<td><code>expected and actual</code> are not equal</td>
</tr>
<tr>
<td><code>assertSame(expected, actual)</code></td>
<td><code>expected != actual</code></td>
</tr>
<tr>
<td><code>assertNotSame(expected, actual)</code></td>
<td><code>expected == actual</code></td>
</tr>
<tr>
<td><code>assertNull(value)</code></td>
<td><code>value != null</code></td>
</tr>
<tr>
<td><code>assertNotNull(value)</code></td>
<td><code>value == null</code></td>
</tr>
</tbody>
</table>

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

```java
@Test(expected=IndexOutOfBoundsException.class)
public void testGetEmptyList() {
    List<String> list = new ArrayList<>();
    list.get(0);
}
```
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:
  ```java
  @Before
  public void m() { ... }
  @After
  public void m() { ... }
  ```
• Run once before/after running all test methods in the class:
  ```java
  @BeforeClass
  public static void m() { ... }
  @AfterClass
  public static void m() { ... }
  ```