WARMUP

A programmer's wife tells him, "Would you mind going to the store and picking up a loaf of bread. Also, if they have eggs, get a dozen."

The programmer returns with 12 loaves of bread.

Section 3:

HW4, ADTs, and more

Slides by Vinod Rathnam

with material from Alex Mariakakis, Krysta Yousoufian, Mike Ernst, Kellen Donohue

AGENDA

- Announcements
  - HW3: due yesterday
  - HW4: due next Wednesday April 22nd
- Polynomial arithmetic
- Abstract data types (ADT)
- Representation invariants (RI)
- Abstraction Functions
- Further information found in **Calendar/info** & **docs/handouts** link on website

HW4: POLYNOMIAL GRAPHING CALCULATOR

- **Problem 0:** Write pseudocode algorithms for polynomial operations
- **Problem 1:** Answer questions about RatNum
- **Problem 2:** Implement RatTerm
- **Problem 3:** Implement RatPoly
- **Problem 4:** Implement RatPolyStack
- **Problem 5:** Try out the calculator

RATTHINGS

- RatNum
  - ADT for a Rational Number
  - Has NaN
- RatTerm
  - Single polynomial term
  - Coefficient (RatNum) & degree
- RatPoly
  - Sum of RatTerms
- RatPolyStack
  - Ordered collection of RatPolys

POLYNOMIAL ADDITION

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

POLYNOMIAL ADDITION

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

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

POLYNOMIAL ADDITION

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

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

POLYNOMIAL ADDITION

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

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

$$3x^5 + 5x^4 - 2x^3 - x^2 + x + 0$$

POLYNOMIAL SUBTRACTION

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

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

POLYNOMIAL SUBTRACTION

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

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

POLYNOMIAL SUBTRACTION

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

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

$$-3x^5 + 5x^4 + 6x^3 - x^2 - x + 10$$

POLYNOMIAL MULTIPLICATION

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

POLYNOMIAL MULTIPLICATION

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

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

POLYNOMIAL MULTIPLICATION

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

$$\begin{array}{lr} & 4x^3 - x^2 + 5 \\ * & x - 5 \\ \hline & -20x^3 + 5x^2 \quad - 25 \end{array}$$

POLYNOMIAL MULTIPLICATION

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

$$\begin{array}{lr} & 4x^3 - x^2 + 5 \\ * & x - 5 \\ \hline & -20x^3 + 5x^2 \quad - 25 \\ & 4x^4 \quad -x^3 \quad + 5x \end{array}$$

POLYNOMIAL MULTIPLICATION

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

$$\begin{array}{lr} & 4x^3 - x^2 + 5 \\ * & x - 5 \\ \hline & -20x^3 + 5x^2 \quad - 25 \\ + & 4x^4 \quad -x^3 \quad + 5x \\ \hline & \mathbf{4x^4 \; -21x^3 + 5x^2 + 5x - 25} \end{array}$$

POLYNOMIAL DIVISION

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

POLYNOMIAL DIVISION

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

$x^3 - 2x - 5$ | $5x^6 + 4x^4 - x^3 + 5$

POLYNOMIAL DIVISION

1 0 -2 -5 | 5 0 4 -1 0 0 5

POLYNOMIAL DIVISION

5

1 0 -2 -5 | 5 0 4 -1 0 0 5

POLYNOMIAL DIVISION

5

1 0 -2 -5 | 5 0 4 -1 0 0 5

5 0 -10 -25

POLYNOMIAL DIVISION

5

1 0 -2 -5 | 5 0 4 -1 0 0 5

5 0 -10 -25

0 0 14 24

POLYNOMIAL DIVISION

5

1 0 -2 -5 | 5 0 4 -1 0 0 5

5 0 -10 -25

0 0 14 24

14 24 0

POLYNOMIAL DIVISION

```
                  5  0
1  0 -2 -5  | 5  0  4  -1  0  0  5
              5  0-10 -25
              0  0 14  24
                   14  24 0
```

POLYNOMIAL DIVISION

```
                  5  0
1  0 -2 -5  | 5  0  4  -1  0  0  5
              5  0-10 -25
              0  0 14  24
                   14  24 0
                   14  24 0  0
```

POLYNOMIAL DIVISION

```
                  5  0 14
1  0 -2 -5  | 5  0  4  -1  0  0  5
              5  0-10 -25
              0  0 14  24
                   14  24 0
                   14  24 0  0
```

POLYNOMIAL DIVISION

```
                  5  0 14
1  0 -2 -5  | 5  0  4  -1  0  0  5
              5  0-10 -25
              0  0 14  24
                   14  24 0
                   14  24 0  0
                   14  0 -28 -70
```

POLYNOMIAL DIVISION

```
                  5  0 14
1  0 -2 -5  | 5  0  4  -1  0  0  5
              5  0-10 -25
              0  0 14  24
                   14  24 0
                   14  24 0  0
                   14  0 -28 -70
                   0  24  28  70
```

POLYNOMIAL DIVISION

```
                  5  0 14
1  0 -2 -5  | 5  0  4  -1  0  0  5
              5  0-10 -25
              0  0 14  24
                   14  24 0
                   14  24 0  0
                   14  0 -28 -70
                   0  24  28  70
                      24  28  70 5
```

POLYNOMIAL DIVISION

```
                     5  0 14 24
 1  0 -2 -5    5  0  4  -1  0  0  5
               5  0-10 -25
               0  0 14  24
                    14  24 0
                    14  24 0  0
                    14  0 -28 -70
                    0  24  28  70
                       24  28  70 5
                       24  0  -48 -120
```

POLYNOMIAL DIVISION

```
                     5  0 14 24
 1  0 -2 -5    5  0  4  -1  0  0  5
               5  0-10 -25
               0  0 14  24
                    14  24 0
                    14  24 0  0
                    14  0 -28 -70
                    0  24  28  70
                       24  28  70 5
                       24  0  -48 -120
                       0     28 118 125
```

POLYNOMIAL DIVISION

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

$5x^3 + 14x + 24$

POLYNOMIAL DIVISION

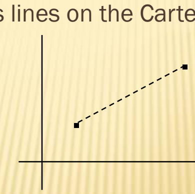
$(5x^6 + 4x^4 - x^3 + 5) / (x^3 - 2x - 5)$

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

CALCULATORFRAME DEMO

ADT EXAMPLE: LINE

Suppose we want to make a `Line` class that represents lines on the Cartesian plane



See http://courses.cs.washington.edu/courses/cse331/15sp/concepts/specifications.html for more

ADT EXAMPLE: LINE

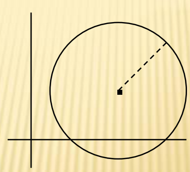
```
/**
 * This class represents the mathematical concept of a line segment.
 *
 * A line is an immutable line segment on the 2D plane that has endpoints p1
 * and p2
 */
public class Line {
…
}
```

REPRESENTATION INVARIANTS

- Constrains an object's internal state
- Maps concrete representation of object to a boolean
- If representation invariant is false/violated, the object is "broken" – doesn't map to any abstract value

ADT EXAMPLE: CIRCLE

- Circle on the Cartesian coordinate plane

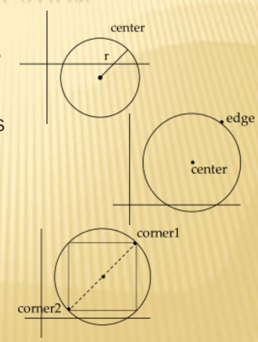


CIRCLE: CLASS SPECIFICATION

What represents the abstract state of a Circle?
- Center
- Radius

What are some properties of a circle we can determine?
- Circumference
- Area

How can we implement this?
- #1: Center, radius
- #2: Center, edge
- #3: Corners of diameter



CIRCLE IMPLEMENTATION 1

```
public class Circle1 {
     private Point center;
     private double rad;

     // Rep invariant:
     //

     //  ...
}
```

CIRCLE IMPLEMENTATION 1

```
public class Circle1 {
     private Point center;
     private double rad;

     // Rep invariant:
     // center != null && rad > 0

     //  ...
}
```

CIRCLE IMPLEMENTATION 2

```
public class Circle2 {
    private Point center;
    private Point edge;

    // Rep invariant:
    //

    //    ...
}
```

CIRCLE IMPLEMENTATION 2

```
public class Circle2 {
    private Point center;
    private Point edge;

    // Rep invariant:
    // center != null &&
    // edge != null &&
    // !center.equals(edge)
    //    ...
}
```

CIRCLE IMPLEMENTATION 3

```
public class Circle3 {
    private Point corner1, corner2;

    // Rep invariant:
    //


    //    ...
}
```

CIRCLE IMPLEMENTATION 3

```
public class Circle3 {
    private Point corner1, corner2;

    // Rep invariant:
    // corner1 != null &&
    // corner2 != null &&
    // !corner1.equals(corner2)
    //    ...
}
```

CHECKING REP INVARIANTS

- Representation invariant should hold before and after every public method
- Write and use checkRep()
  - Call before and after public methods
  - Make use of Java's assert syntax!
  - OK that it adds extra code
    - Asserts won't be included on release builds
    - Important for finding bugs

CHECKREP() EXAMPLE WITH EXCEPTIONS

```
public class Circle1 {
        private Point center;
        private double rad;

        private void checkRep() throws RuntimeException {
                if (center == null) {
                        throw new RuntimeException(“This does
                                not have a center");
                }

                if (radius <= 0) {
                        throw new RuntimeException(“This
                        circle has a negative radius”);
                }
        }
}
```

CHECKREP() EXAMPLE WITH ASSERTS

```
public class Circle1 {
        private Point center;
        private double rad;

        private void checkRep() throws RuntimeException {
                assert center != null : "This does not have a
                                          center";
                assert radius > 0 : "This circle has a negative
                                     radius";
        }
}
```

A lot neater!

USING ASSERTS

- To enable asserts: Go to Run->Run Configurations...->Arguments tab-> input –**ea** in VM arguments section
  - Do this for every test file
  - Demo!

ABSTRACTION FUNCTION

- Abstraction function: a **mapping** from **internal state** to **abstract value**
- Abstract fields may not map directly to representation fields
  - Circle has **radius** but not necessarily `private int radius;`
- Internal representation can be anything as long as it somehow encodes the abstract value
- Representation Invariant excludes values for which the abstraction function has no meaning

CIRCLE IMPLEMENTATION 1

```
public class Circle1 {
        private Point center;
        private double rad;

        // Abstraction function:
        // AF(this) = a circle c such that
        //     c.center =
        //     c.radius =

        // Rep invariant:
        // center != null && rad > 0

        //  ...
}
```

CIRCLE IMPLEMENTATION 1

```
public class Circle1 {
        private Point center;
        private double rad;

        // Abstraction function:
        // AF(this) = a circle c such that
        //     c.center = this.center
        //     c.radius = this.rad

        // Rep invariant:
        // center != null && rad > 0

        //  ...
}
```

CIRCLE IMPLEMENTATION 2

```
public class Circle2 {
        private Point center;
        private Point edge;

        // Abstraction function:
        // AF(this) = a circle c such that
        //     c.center =
        //     c.radius =

        // Rep invariant:
        // center != null && edge ! null &&
           !center.equals(edge)

        //    ...
}
```

CIRCLE IMPLEMENTATION 2

```
public class Circle2 {
	private Point center;
	private Point edge;

	// Abstraction function:
	// AF(this) = a circle c such that
	//     c.center = this.center
	//     c.radius = sqrt((center.x-edge.x)^2 +
	                (center.y-edge.y)^2)

	// Rep invariant:
	// center != null && edge ! null &&
	   !center.equals(edge)

	//    ...
}
```

CIRCLE IMPLEMENTATION 3

```
public class Circle3 {
	private Point corner1, corner2;

	// Abstraction function:
	// AF(this) = a circle c such that
	//     c.center =
	//     c.radius =

	// Rep invariant:
	// corner1 != null && corner2 != null &&
	   !corner1.equals(corner2)

	//    ...
}
```

CIRCLE IMPLEMENTATION 3

```
public class Circle3 {
	private Point corner1, corner2;

	// Abstraction function:
	// AF(this) = a circle c such that
	//     c.center = <(corner1.x + corner2.x) / 2,
	                (corner.y + corner2.y) / 2>

	//     c.radius = (1/2)*sqrt((corner1.x-
	                corner2.x)^2 + (corner1.y-
	                corner2.y)^2)

	// Rep invariant:
	// corner1 != null && corner2 != null &&
	   !corner1.equals(corner2)

	//    ...
}
```

ADT EXAMPLE: NONNULLSTRINGLIST

```
public class NonNullStringList  {
	// Abstraction function:
	// ??

	// Rep invariant:
	// ??

	public void add(String s) { ... }
	public boolean remove(String s) { ... }
	public String get(int i) { ... }
}
```

NONNULLSTRINGLIST IMPLEMENTATION 1

```
public class NonNullStringList  {
	// Abstraction function:
	// Index i in arr contains the ith element in the
	   list

	// Rep invariant:
	// RI = [0,count-1] != null

	private String[] arr;
	private int count;

	public void add(String s) { ... }
	public boolean remove(String s) { ... }
	public String get(int i) { ... }
}
```

Problems?

NONNULLSTRINGLIST IMPLEMENTATION 2

```
public class NonNullStringList  {
	// Abstraction function:
	// Value in the nth node after head contains the
	   nth item in the list

	// Rep invariant:
	// RI = Head has size nodes after it, each whose
	   value is non-null, no cycle in ListNodes

	public int size;
	public ListNode head;

	public void add(String s) { ... }
	public boolean remove(String s) { ... }
	public String get(int i) { ... }
}
```