COMP250 - Lecture 6

Java: Classes & Objects

Example: implementing Polynomials

Data, functions/methods processing it

Classes package these together

Object = instance of a class (example, value)

E.g. $2 + 3x + 15x^2$ - Polynomial

data

add / subtracting Poly, evaluate Poly at some value of x...

at index i → coefficient of x^i

```
public class Poly {
    // Data
    double[] coefficients;
    // Usually data is not public
    // Constructor / initializer & memory allocation methods
    // Same name as name of class
    public Poly() {
        coefficients = new double[1];
        coefficients[0] = 0;
    }
```

[0]

does not return anything!

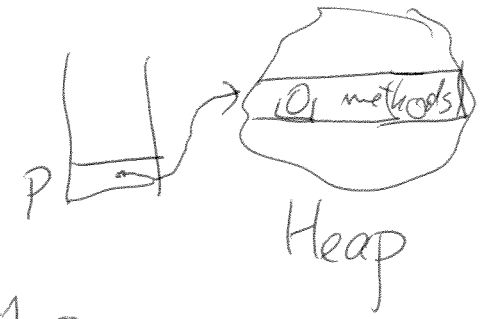
```
    // Method for printing
    public void displayPoly() {
        for (int i=0; i<coefficients.length; i++)
            System.out.print(" " + coefficients[i] + "*X^" + i);
        System.out.println();
    }
```

// displayPoly will work on coefficients array within the current polynomial.

```
public class TestPoly {
  public static void main(String[] args) {
    Poly p = new Poly();
    p.displayPoly();
  }
}
```

Poly p = new Poly();

- Poly: type of variable
- p: name
- new Poly(): construct p



p.displayPoly(); // 0*X^0

- p: object name
- displayPoly(): method

Poly q = new Poly(5);


```
double[] a = {1, 2, 3};
Poly r = new Poly(a);
```

```
public Poly (int degree) {
    coefficients = new double [degree];
    for(int i=0; i<coefficients.length; i++)
        coefficients[i]=0;
}
public Poly (double[] a) {
    coefficients = new double [a.length];
    // for loop to copy a into coefficients
    for (int i=0; i<a.length; i++)
        coefficients[i]=a[i];
}
public Poly (Poly p) {
    . . .
}
```

↑ what is passed is different type than in the other constructors

```
// Accessing data methods
// Accessors: returns some data
public double getCoefficient(int i) {
    return coefficients[i];
}
// return coefficients (gives address of array -> modify
```

i too big -> Error -> ArrayIndexOutOfBounds

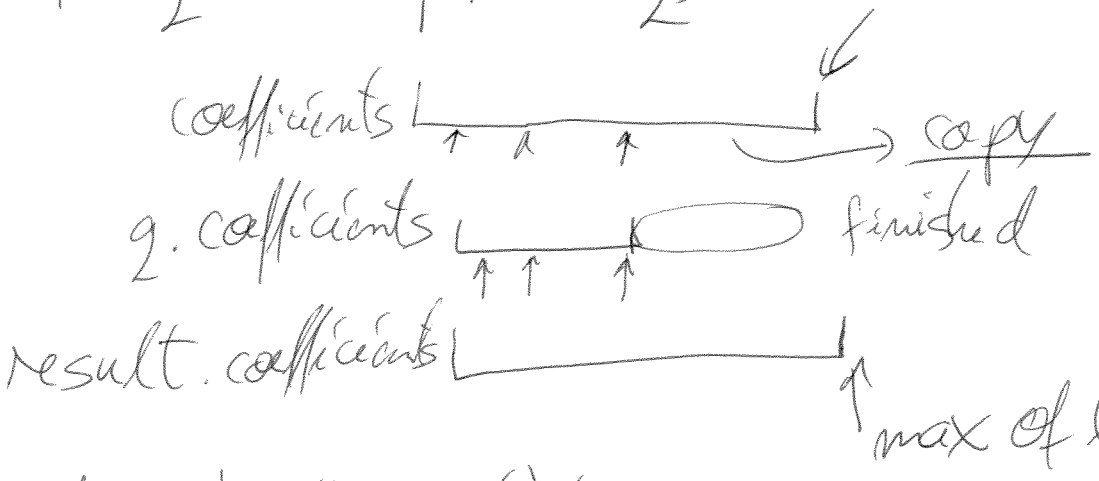
Errors = Exceptions (objects)

```
// More robust version
public double getCoefficient(int i) throws Exception {
                                          ↑ predefined Java class
    (part of "contract")
  // first check the input
  // if not ok -> generate error
  if (i >= coefficients.length)
      throw new Exception("should pass arg no bigger than" + coefficients.length);
  return coefficients[i];
}
```

constructs on the fly an Exception object

// Adding Polys

// p + q p.add(q)




```
public int getDegree() {
    return coefficients.length - 1;
} // Access data through methods not directly
//Mutator method: sets some data to a value
 public void setCoefficient (int i, double val) {
    coefficients[i] = val;
 } // could do this using Exception
```