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 $i$ $\rightarrow$ 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;
}
}

// does not return anything!

// Method for printing

public void displayPoly() {
    for (int i = 0; i < coefficients.length; i++)
        System.out.print("a^" + coefficients[i] + "x^" + i);
    System.out.println();
}

// displayPoly will work on coefficients away within
// the current polynomial.
public class TestPoly {
    public static void main (String[] args) {
        Poly p = new Poly();
        p. displayPoly();
        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
// Accesses: returns some data
public double getCoefficient (int i) {
    return coefficients[i];
}

// Return coefficients (gives address of array to modify)
// i too big -> Error: ArrayIndexOutOfBoundsException
Errors = Exceptions (objects)

// More robust version
public double getCoefficient (int i) throws Exception {  // part of "contract"
    // first check the input
    // if not ok -> generate error
    if (i >= coefficients.length) {
        throw new Exception("Should pass an index bigger than " + coefficients.length);
    }
    return coefficients[i];  // constructs an Exception object
// Adding polys
// p + q   p.add(q)

coefficients += a + b -> copy
q.coefficients = finished
result.coefficients = max of lengths

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