

Problem 3

Pseudocode

- let u, v as polynomials; goal u / v
- define **ZERO** as "0" polynomials
- define **Poly()** create polynomial(**Poly**) from **double[]**

```

Poly q = ZERO
Poly r = copy of u

// Precondition: v != 0 && u.degree >= v.degree
while (r != ZERO && r.degree > v.degree):
    // LI: (u = q * v + r) && r >= ZERO

    int scale = r.degree - v.degree
    double frac = r[r.degree] / v[v.degree]

    double[] rn = new double[scale + 1]
    Fill rn with 0
    rn[scale] = frac

    q += Poly(rn)

    r -= Poly(rn) * v
    // Exit Condition: r == ZERO || r.degree <= v.degree

// Postcondition: u / v == q + r

return q;

```

Prove

Base Case

$$q = 0 \Rightarrow u = r + 0 * v \Rightarrow u = r$$

If u is zero, then the loop won't run and postcondition still hold as $0 / v = 0 = u$

Induction

Assume LI holds for iteration k , prove it true at iteration $k + 1$

During iteration, r will be divided by v , and $r.degree - 1$

$$\text{If } k \text{ holds LI, then at } k + 1 \Rightarrow u = v * q + r$$

Since r must be larger than **ZERO** at iteration k (or loop exited), and r will decrease toward **ZERO**, so $r \geq \text{ZERO}$ is satisfied.

Exit the loop

r will finally go to ZERO or $r.\text{degree} \leq v.\text{degree}$, so $u / v == q + r$, at decreasing
function $D = r$