Problem 3

1. Write a pseudocode algorithm for polynomial division. Write your answer in the file answers/problem3.pdf

   When writing pseudocode use symbols +, -, *, and / to express rational number and polynomial arithmetic. You may also use u[i] to retrieve the coefficient at power i of polynomial u, as well as c * x^i to denote the single-term polynomial of degree i and coefficient c.

   pseudocode:

```
RatPoly div(RatNum[] u, RatNum[] v)
{Precondition: u.degree >= v.degree > 0; !(v.equals(ZERO))}
{
        RatNum[] p = u;
        RatNum[] q = {0};

        while (p.degree >= v.degree && (!(p.equals(ZERO))) {
                RatNum i = p[p.degree - 1] / v[v.degree - 1];
                q = q + i;
                p = p - (i * v);
        }

        return q, p;
}
{Postcondition: u = q * v + p}
```

2. State the loop invariant for the main loop and prove partial correctness. Write your answer in the file answers/problem3.pdf.

   For the proof question, you do not need to handle division by zero; however, you will need to do so in the Java program.

Important: write your pseudocode, invariants, and proofs first, then write the Java code. Going backwards will be harder.

Loop invariant: u = q * v + p

Proof:

Base Case: To prove the base case, we need to show that the expression q times v plus p is equal to u. By substituting the values of q and r, we get the expression 0 times v plus u, which simplifies to u. This holds true for any value of v, therefore the base case is proven.

Induction Step: For the induction step, we assume that the loop invariant is true for the k'th iteration and aim to prove it for the (k+1)'th iteration. We identify the values that undergo change, namely new_q and new_p. We rearrange the expressions for the (k+1)'th iteration and simplify to show that it is equivalent to the k'th iteration. This demonstrates that the loop invariant holds true for the (k+1)'th iteration and completes the induction step.

Partial Correctness: In terms of partial correctness, the loop invariant remains the same throughout the loop and does not violate the exit condition, which states that either p is equal to zero or p's degree is less than v's degree. Therefore, the loop satisfies the principle of partial correctness