Precondition: this.degree >= p.degree, this and p are not null nor NAN

```
Copy = this
Index = 0
new_size = this.degree()-p.degree+1
degree_diff = this.degree – tmp.degree
While (degree_diff >= 0)
{
        answer = this / p
        i = new_size – index-1
        while (i < this.length – index)
        {
                tmp = copy – tmp
                i ++
        }
        index ++
        degree_diff--
}
```

Postcondition: this = answer * p + tmp

Inner LI: i < this.length +1

Base Case:
index = 0, i = new_size -1 < this.length < this.length+1, TRUE

Induction:
i_new = i_old + 1
i_new <= this.length – index +1
i_old +1 <= this.length – index +1
i_old <= this.length - index + 1

Exit Condition:
i <= this.length – index +1 && i < this.length +1
i <= this.length – index +1

Outer LI: this = answer * p + tmp && diff >= 0

Base Case:

temp is always the remainder. The base case will be this = answer * divisor + remainder.

Which is TRUE

Induction:

index_new = index_old + 1

tmp_new = copy – tmp_old

answer_new = this / p

this = answer * p + tmp

= answer_new * p + tmp_new

= this/p * p +copy – tmp_old

= this + copy – tmp_old

= answer * p + tmp + copy – tmp_old

= answer * p + tmp

Exit Condition:

degree_diff == 0 && (this = answer * p + tmp) && (degree_diff >= 0)

(this = answer * p + tmp) && (degree_diff >= 0)