

```

/*
 * Polynomial.h
 *
 * Created on: Dec 28, 2009
 * Author: koscum
 */

#ifndef POLYNOMIAL_H_
#define POLYNOMIAL_H_

#include "Error.h"
#include <iostream>
#include <vector>

namespace GF
{
    class Error;
    class GF;

    class Polynomial
    {
    public:
        Polynomial();
        Polynomial(const std::vector<int>&);
        Polynomial(const int*, int);
        Polynomial(int);
        Polynomial(int, int);
        Polynomial(const Polynomial&);
        virtual ~Polynomial();

        Polynomial& Add(const Polynomial&) const;
        Polynomial& Sub(const Polynomial&) const;
        Polynomial& Mul(const Polynomial&) const;
        Polynomial& Div(const Polynomial&) const;
        Polynomial& Mod(const Polynomial&) const;
        Polynomial& Pow(const Polynomial&) const;

        Polynomial& Add(const Polynomial&, const int) const;
        Polynomial& Sub(const Polynomial&, const int) const;
        Polynomial& Mul(const Polynomial&, const int) const;
        Polynomial& Div(const Polynomial&, const int) const;
        Polynomial& Mod(const Polynomial&, const int) const;
        Polynomial& Pow(const Polynomial&, const int) const;

        friend Polynomial& operator +(const Polynomial& polynomialA, const Polynomial&
polynomialB);
        friend Polynomial& operator -(const Polynomial& polynomialA, const Polynomial&
polynomialB);
        friend Polynomial& operator *(const Polynomial& polynomialA, const Polynomial&
polynomialB);
        friend Polynomial& operator /(const Polynomial& polynomialA, const Polynomial&
polynomialB);
        friend Polynomial& operator %(const Polynomial& polynomialA, const Polynomial&
polynomialB);
        friend Polynomial& operator ^(const Polynomial& polynomialA, const Polynomial&
polynomialB);

```

```

Polynomial& operator =(const Polynomial&);

Polynomial& operator +=(const Polynomial&);
Polynomial& operator -=(const Polynomial&);
Polynomial& operator *=(const Polynomial&);
Polynomial& operator /=(const Polynomial&);
Polynomial& operator %=(const Polynomial&);
Polynomial& operator ^=(const Polynomial&);

friend bool operator ==(const Polynomial& polynomialA, const Polynomial& polynomialB);
friend bool operator !=(const Polynomial& polynomialA, const Polynomial& polynomialB);
friend bool operator <(const Polynomial& polynomialA, const Polynomial& polynomialB);
friend bool operator >(const Polynomial& polynomialA, const Polynomial& polynomialB);
friend bool operator <=(const Polynomial& polynomialA, const Polynomial& polynomialB);
friend bool operator >=(const Polynomial& polynomialA, const Polynomial& polynomialB);

Polynomial& Normalise(int base) const;

int GetOrder() const;
int GetCoefficient(int) const;
int operator [] (int) const;

protected:
    void Copy(const Polynomial&);
    void Out(std::ostream&);
    void Free();

private:
    static Polynomial& AddSub(const Polynomial& polynomialA, const Polynomial&
polynomialB, const int base, const bool mode);
    static Polynomial& MulPow(const Polynomial& polynomialA, const Polynomial&
polynomialB, const int base, const bool mode);
    static Polynomial& DivMod(const Polynomial& polynomialA, const Polynomial&
polynomialB, const int base, const bool mode);
    static int ModuoDiv(const int a, const int b, const int base);

    void Truncate();

    friend std::ostream& operator <<(std::ostream&, const Polynomial&);

public:
    static const Polynomial ZERO;
    static const Polynomial ONE;

private:
    int order;
    std::vector<int> coefficients;
};
}

#endif /* POLYNOMIAL_H_ */

```