```
/// <summary>
/// Класс Rational определяет новый тип данных - рациональные числа и основные операции над
/// ними - сложение, умножение, /// вычитание и деление. Рациональное число задается парой
/// целых чисел (m,n) и изображается обычно в виде дроби m/n. Число m называется числителем,
/// n - знаменателем. Для каждого рационального числа существует множество его представлений,
/// например, 1/2, 2/4, 3/6, 6/12 – задают одно и тоже рациональное число. Среди всех представлений
/// можно выделить то, в котором числитель и знаменатель взаимно несократимы. Такой
/// представитель будет храниться в полях класса. Операции над рациональными числами
/// определяются естественным для математики образом
/// </summary>
public class Rational
{
// Описание тела класса Rational
}//Rational


//Поля класса. Числитель и знаменатель рационального числа.
int m,n;


/// <summary>
/// Конструктор класса. Создает рациональное число  m/n, эквивалентное a/b, но со взаимно-
/// несократимыми числителем и знаменателем. Если b=0, то результатом  является рациональное
/// число 0 -пара (0,1).
/// </summary>
/// <param name="a">числитель</param>
/// <param name="b">знаменатель</param>

public Rational(int a, int b)
{
if(b==0) {m=0; n=1;}
else
{
//приведение знака
if( b<0) {b=-b; a=-a;}
//приведение к несократимой дроби
int d = nod(a,b);
m=a/d; n=b/d;
}
}

/// <summary>
/// Закрытый метод класса. Возвращает наибольший общий делитель чисел a,b
/// </summary>
/// <param name="a">первое число</param>
/// <param name="b">второе число, положительное</param>
/// <returns>НОД(a,b)</returns>
int nod(int m, int n) {
int p=0;
m=Math.Abs(m); n =Math.Abs(n);
if(n>m){p=m; m=n; n=p;}
do {
p = m%n; m=n; n=p;
}while (n!=0);
return(m); }//nod
```

```
public void PrintRational(string name)
{
Console.WriteLine(" {0} = {1}/{2}",name,m,n);
}


public Rational Plus(Rational a)
{
int u,v;
u = m*a.n +n*a.m; v= n*a.n;
return( new Rational(u, v));
}//Plus
public static Rational operator +(Rational r1, Rational r2)
{return (r1.Plus(r2));
}


public Rational Minus(Rational a)
{
int u,v;
u = m*a.n - n*a.m; v= n*a.n;
return( new Rational(u, v));
}//Minus
public static Rational operator -(Rational r1, Rational r2)
{return (r1.Minus(r2));
}
public Rational Mult(Rational a)
{
int u,v;
u = m*a.m; v= n*a.n;
return( new Rational(u, v));
}//Mult
public static Rational operator *(Rational r1, Rational r2)
{
return (r1.Mult(r2));
}
public Rational Divide(Rational a)
{
int u,v;
u = m*a.n; v= n*a.m;
return( new Rational(u, v));
}//Divide
public static Rational operator /(Rational r1, Rational r2)
{
return (r1.Divide(r2));
}
```