```
1 //日期: 2018/ 时间:
 2 #include <stdio.h>
 3 #include <stdlib.h>
 4 #include <algorithm>
 5 using namespace std;
 6 //制定以下总则
 7 //down为非负数,如果分数为负,令分子up为负即可;
 8 //如果该分数为0,那么规定其分子为0,分母为1
 9 //分子和分母没有除了1以外的公约数
10 struct Fraction{
       int up,down;
11
12 };
13
14 int gcd(int a,int b){
15
       return !b ? a : gcd(b,a%b);
16 }
17
18 //分数化简, 化简步骤有三:
19 //如果分母down为负数,那么分子分母取相反数
20 //如果分子up为0,那么令分母down为1
21 //找出分子分母绝对值的最大公约数,分子分母分别除以这个数
22 Fraction reduction(Fraction result){
23
       if(result.down < 0){</pre>
24
          result.up = -result.up;
25
          result.down = -result.down;
26
27
       if(result.up == 0)
28
          result.down = 1;
29
       else{
           int d = gcd(abs(result.down),abs(result.up));
30
31
           result.down /= d;
32
           result.up /= d;
33
       }
34
35
       return result;
36 }
37
38 //分数的输出
39 void showResult(Fraction r){
40
      r = reduction(r);
       if(r.down == 1) printf("%d",r.up);
41
42
       else if(abs(r.up) > r.down){
43
           printf("%d %d/%d",r.up/r.down,abs(r.up)%r.down,r.down);
44
       }else{
           printf("%d/%d",r.up,r.down);
45
46
47 }
48
49 //以下的加减乘除都是一样的。
50 Fraction add(Fraction f1,Fraction f2){
51
       Fraction result;
       result.up = f1.up * f2.down + f2.up * f1.down;
52
       result.down = f1.down * f2.down;
54
       return reduction(result);
55 }
56
```

```
57 Fraction minu(Fraction f1,Fraction f2){
       Fraction result;
       result.up = f1.up * f2.down - f2.up * f1.down;
59
60
       result.down = f1.down * f2.down;
61
       return reduction(result);
62 }
63
64 Fraction multi(Fraction f1,Fraction f2){
       Fraction result;
66
       result.up = f1.up * f2.up;
67
       result.down = f1.down * f2.down;
68
       return reduction(result);
69 }
70
71 Fraction divide(Fraction f1, Fraction f2){
72
       Fraction result;
73
       result.up = f1.up * f2.down;
74
       result.down = f1.down * f2.up;
75
       return reduction(result);
76 }
77
78 int main(){
79
80
81
       return 0;
82 }
83
84
```