ENGRECE 12: Assignment-6

February 19, 2004

Due Wednesday 3/3/2004 12:00pm"Two-week assignment"

1 Class RationalNumber [80 points]

The goal of this program is to get familiar with object oriented programing. Object oriented programming is a powerful technique in programming that lets your program be more robust and easy to be modified. Please read exercise 7.4 at page 258 to get familiar to this assignment.

Initial values of numerator and denominator are 0 and 1 if not mentioned by the user when you call the class constructor. So $0/1$ is the initial format of any rational number if the numerator and denominator are not set by the user.

There are a few sample of rational numbers in the following line:

$$A = (\ 2/7\ )\quad B = (\ 3/5\ )\quad C = (\ 12/36\ )$$

A and B are reduced however, C could be shown as $1/3$ as well. Your program should always print a rational number in its reduced format(print 1/3 instead of 12/36) This can be easily achieved by calling a `reduce` function *before* and *after* any computation.

Your task is to create a class `RationalNumber` with the following funtions as the class member functions:

- `Sum     To add two rational numbers (See 7.4 (a))`
- `Sub   To subtract two rational numbers (See 7.4 (b))`
- `Mul   To multiply two rational numbers (See 7.4 (c))`
- `Div   To devide two rational numbers (See 7.4 (d))`
- `Reduce   To reduce the numerator and denominator of a rational number`
- `Print   To print the rational number in the form a/b (See 7.4 (e))`

Your class should contain the above six methods (in addition to any other methods you my need). Using your class, it should be possible to define rational number objects, compute with them, and print them in a readable format.

`Hint`: For your class method named $reduce$ you can use the following pseudo code:

```
reduce(n,d)
  p=2
  while(p <= n AND p <= d)
    if ((n%p == 0) AND (d%p == 0))
        n = n/p
        d = d/p
    else
        p = p+1
  return(n,d)
```

Use the exact following test cases to test your program. You are welcome to add additional test cases at the end of the following test script to show that your program works fine with different numbers as well.

```
Sum:       1/41 + 37/173
Sub:       3/8 - 19/57
Mul:       9/17 * 113/209
Div:       71/237 / 3/1
Reduce:    12/36, 244/420, 27/81
```

2 Extra Credit [20 points]

For a maximum of 20 extra credits, improve your program in the folowing areas:

_Provide *Operator Overloading* so that rational numbers can be computed by the usual +, -, *, / operators

_allow negative rational numbers and improve your methods accordingly; note that the pseudo code for the above `reduce` reduce does not support any negative values and needs modification

_handle error conditions safely: for example, division by zero errors should be avoided; make sure the denominator never equals zero

_provide conversion functions from your rational numbers to floating point numbers and integers(use truncation for rounding)

3 What to turn in

Use the command

```
% python ~ece12/tools/submit.pyc
```

to turn in your homework. Your files should be a level above the `hw6` directory. You should have the following files:

- `RatinalNumber.py    RationalNumber.script`