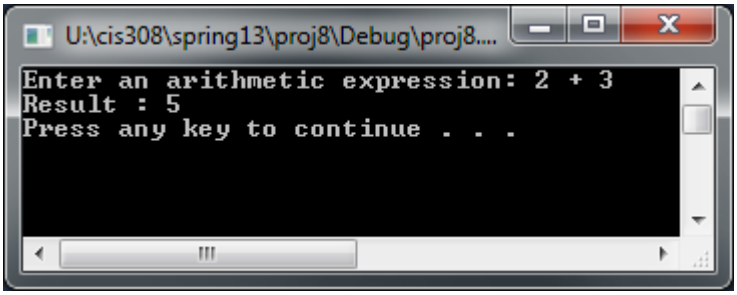
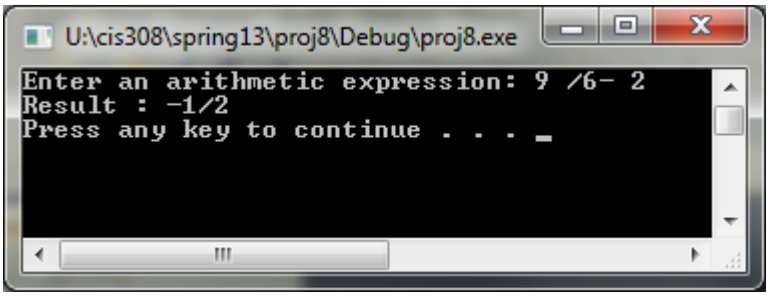
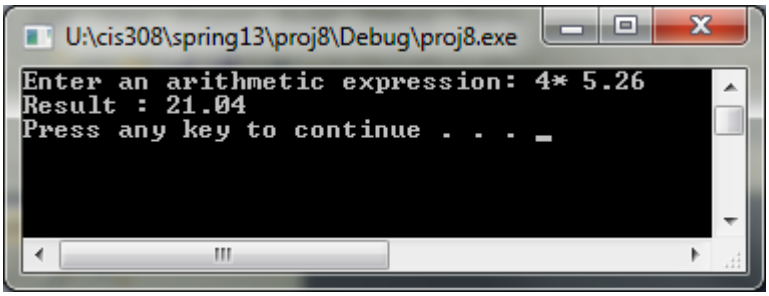
Programming Project 8 (50 points)
Due: Thursday, May 9 by midnight

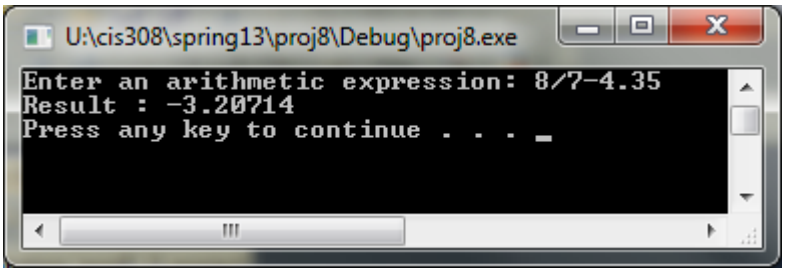
Assignment Description:

You are to write a program in C++ that will evaluate simple arithmetic expressions involving integers, decimals, and fractions. When your program runs, the user will input some kind of number, an operation (+, -, or *), and another number. You will output the result of the operation. Below is a sample of FOUR different runs of the program:









The input will meet the following specifications:

- It will be of the form Number OP Number, where each Number is either an integer, improper fraction, or decimal value. OP is one of +, -, or *.
- There may be some or no whitespace between input values
- You are NOT required to handle input that does not match these specifications, but you are certainly welcome to add error-handling

Your output must meet the following specifications:

- If two integers are given as input, the result must be an integer
- If two rational numbers are given as input, the result must be a reduced rational number (a reduced improper fraction)
- If two decimal numbers are given as input, the result must be a decimal number
- If an integer and a rational are given as input, the result must be a reduced rational
- If an integer and a decimal number are given as input, the result must be a decimal number
- If a rational and a decimal number are given as input, the result must be a decimal number

Implementation Requirements:
Your program must include the following files:

- `number.h`
- `real.h and real.cpp`
- `rational.h and rational.cpp`
- `integer.h and integer.cpp`
- `proj8.cpp`

`number.h` defines an abstract class that represents a number. It should include the following pure virtual functions:

```
double value()
Number* plus(Number *n)
Number* minus(Number *n)
Number* times(Number *n)
void print(void)
```

The `value` function should return the value of this `Number`, and the `print` function should display this `Number`'s value. The `plus`, `minus`, and `times` functions should apply the operation between this `Number` and the argument (`n`), and return a pointer to a `Number` that represents the result.

`real.h` and **`real.cpp`** define the class `Real` that represents real numbers (with decimal values). `Real` should extend `Number`, and should thus implement all its pure virtual functions. You may define any other variables or functions that you wish.

`rational.h` and **`rational.cpp`** define the class `Rational` that represents rational numbers (improper fractions that are not necessarily reduced). `Rational` should extend `Real`, and should also implement all `Number`'s pure virtual functions. You may define any other variables or functions that you wish. Notice that your `print` function in `Rational` should display the number as a reduced improper fraction.

`integer.h` and **`integer.cpp`** define the class `Integer` that represents integer numbers. `Integer` should extend `Rational`, and should also implement all `Number`'s pure virtual functions. You may define any other variables or functions that you wish.

Each class must also include a **constructor** to initialize the instance variables.

`proj8.cpp` must contain the `main` function for your program. It should ask the user to enter an arithmetic expression, and should create two `Number` pointer variables – one for each number in the expression. If the number entered was an integer, you should create an instance of `Integer`; if it was a fraction, you should create an instance of `Rational`; if it was a decimal number, you should create an instance of `Real`. Then, call the appropriate operation function two apply the operation to the two `Number`s, and display the result.

I recommend using `cin.getline` to read the entire input line. Then, use `strtok` to tokenize the string. You might also want to use `strcspn` to get the index of an element within the input string.

Documentation:

Your program must include a comment block at the top of every file, as well as at the top of each function. The function comments should include a brief description of what the function does, and explain any function arguments and return values. You may use the comment block below as a template:

```
/*********************************************
* Name: (YOUR NAME)                          *
* Date: (THE DUE DATE)                       *
* Assignment: Project 8: Representing Numbers *
*********************************************
* (WRITE A DESCRIPTION OF THE PROGRAM)       *
*********************************************/
```

Submission:

Your project must be submitted as a zip file using the Project Submission Link on K-State Online.

Grading:

Programs that do not build in Visual Studio will receive a grade of 0. A grading breakdown for programs that do compile appears below:

Correctly reads input	10
Correctly makes instance of `Integer`, `Rational`, or `Real`	3
`Number` class	5
`Real` class	6
`Rational` class	6
`Integer` class	6
Correct output (same number type)	5
Correct output (mixed number types)	5
Constructors	2
Documentation	2
Total	**50**