In-Class 3 - Opening Files && Expressions

This exercise has you opening a file to read input from reading input, implementing expressions, and doing some formatting. You will read 2 ints and 2 doubles. You should

- Learn how to prompt for a file name and open the file for reading
- Read data from the opened file
- Learn how to create a data file in CodeBlocks
- Gain more experience implementing more expression, including some expressions shown with integer division (as they usually are), but actually needs to be written in C++ not using int constants
- Perform some formatting using fixed, setprecision(), and setw().

Steps for this assignment

1. Opening and reading from a file
   a. Prompt and read an input file name
   b. Open the file
   c. Read 2 integers, into variable like **int1** and **int2**
   d. Read 2 doubles, like **dbl1** and **dbl2**
   e. Use the variable below. Declare variables for each of the following expressions, and print each variable so you see what the expression does. To do this you can declare result1, result2, and result3, OR you can use just result, print the answer, reuse result, print the anser, reuse result and finally print the answer.

```
i.   result1 = int1 % int2;
ii.  result2 = int2 % int1;
iii. result3 = int1 / int2;
```

2. Constants are usually made global, by declaring them ***above main()*** or in a header file.

```
const double INCLASS3_PI = 3.14;
```

3. Now, use this definition of **INCLASS3_PI**, and add lines of code to calculate and to output the value for the area of a semi-circle,
   a. assign radius = **dbl1** (from the file)
   b. semiCircleArea = ½ INCLASS3_PI radius^2

```
radius 11.10 ==> area 193.44
```

4. For the following, use fixed, setprecision(4), setw(12) and the constants for the cmath constants Π and e from the math library as we have done previously. to print the following formatted lines:

```
123456789012123456789012  ( I am showing this line only to show spacing for the next lines.)
           p           e
      3.1416      2.7183
```

cout the values of `M_PI` and `M_E`

5. Encode fahrenheit = 9.0 / 5.0 celsius + 32
   - assign celsius = **dbl2** (from the file, e.g. -11.1)

```
celsius -11.10 = fahrenheit 12.02
```

6. Encode the following:

```
fah1  = (double) (9 / 5) * celsius + 32
fah2  = (double)  9 / 5 * celsius + 32
fah1 number ?==? fah2 number
```

What is the difference between how these expressions? (What is being made a double?)

7. Encode `double dblVar =` (1/4) * numOfSides *lengthOfSides^2 * cot(Π/numOfSides)
   a. This is the formula for the area of a Regular n-polygon.
   b. cot(x) = 1/tan(x)
   c. Assign test values: numOfSides=6 and lengthOfSides=5
   d. Result: **`dblVar = 64.99`**

8. Encode `dblVar =` (1/3) * PI * radiusOfCircle^2 * heightOfCone
   a. This is the formula for the volume of a cone.
   b. Assign test values: radiusOfCircle = 5 and heightOfCone =10
   c. Result: **`dblVar = 261.67`**

9. Encode `dblVar = 15 % 4 + 11 / 5`
   a. Can you change the 4 to 4.0? What happens and why?
   b. Change back the 4 and change the 11 to 11.0. What happens and why?
   c. Finally change the + in the expression to * and leave the 11 as 11.0, how does the expression evaluate and why?
   d. output **`dblVar = 6.60`**

10. Encode `dblVar = 11 / 5 + 11 / 5`
    a. Why is the answer not 11 or 11.0?
    b. Change the first 11 to 11.0. Why does this not still make the answer 4.4?
    c. Submit this last version that has only the first 11 as 11.0.
    d. output **`dblVar = 4.20`**

Input

Make an input file named infile1.in and put this in it:

```
43 11 11.1 -11.1
```

Here is the Input and Output!

Your output should look like this. (Please copy and paste the text from the prompts and text output into your code! You are graded on this on Web-CAT.)

```
Name of input file: inclass3input.txt
result = 10
result = 11
result = 3
radius 11.10 ==> area 193.44

            p           e
       3.1416      2.7183

celsius -11.10 = fahrenheit 12.02

fah1 20.90 ?==? fah2 12.02

dblVar = 64.99
dblVar = 261.67
dblVar = 6.60
dblVar = 4.20
```

Programming Style

An important part of programming is using proper programming style, formatting, and comments. At this point, the most important items are:

- Use descriptive variable names
- Vertical (proper use of blank line mostly) and horizontal spacing (putting spaces around operators, indentation, etc.)
    - Indent properly
    - Skip a line between each small section of your program, so that each section is separated by a blank line,

- Have a beginning comment that describes what the program is doing (just “Homework 2” is not good enough). See the main Canvas page for Coding Guidelines about what every program header needs to contain.
- Each section (i.e. logical chunk of code) begins with a comment describing what that section does, so that the comments alone would provide an outline of the program (It’s NOT good to follow every line of code with a comment.)
- Continue typing your code on the next line if it is too long. Too long is over 80 characters. In Code::Blocks, there is a column indicator at the bottom-middle-ish of the screen that tells you which column the cursor is at.

Submit Your Work

To submit your work to Web-CAT to be graded.