InClass4 - Pizza Slice Calculator

This Project is from Chapter 3 of the textbook. You will calculate how many slices a pizza should be cut into and how many it will serve.

Objectives

- experience with writing whole programs
- experience translating “word problems” into algorithms
- experience prompting and reading data using an input stream using the >> operator
- experience writing expressions in C++

The Web-CAT assignment for submission will not be available until Wednesday afternoon sometime.

Description

Joe’s Pizza Palace needs a program to calculate the number of slices a pizza of any size can be divided into. The program should perform the following steps:

- Ask the user for the diameter of the pizza in inches and read the value (double)
- Calculate the number of slices to cut that size of pizza.
- Display a message telling the number of slices.
- Assumming Bill always eats 4 slizes, Joe eats 2 slices, and Sue eats 3 slices. Display messages saying how many Bill’s that pizza could serve, how many Joe’s that pizza could serve, and how many Sue’s that pizza could serve.

To calculate the number of slices that may be taken from the pizza, you must know the following facts:

Each slice must have an area of at least *19.625* square inches. To calculate the number of slices, simply divide the area of the pizza by *19.625*.The area of the pizza is calculated as "pi r squared" where *pi* is approximately *3.14* and r is the radius (half the the diameter). Use M_PI for *pi* .

Sample Input && Output

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

```
Enter diameter of the pizza: 10

This pizza is cut into 4 slices.
It would serve 1 Bills or 2 Joes or 1 Sues.
```

Run your code with several different inputs and hand check the values to make sure your code is doing the calculations right.

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.