Number Fun

In this assignment, we'll combine arithmetic operations, selection statements, and loops together to solve a larger problem. After completing this program you should also have a good handle on the mod operator (%). You are welcome to write your own functions for this assignment but they are not required.

Sample input and corresponding output

This program will be run interactively; the user should be prompted to enter an integer and then your program will produce another integer depending on whether the input was odd or even.

The following is an example of what you might see when you run the program; the input you type is shown in green (press Enter at the end of a line), and the output generated by the program is shown in black text.

```
Enter an integer: 123
Number is odd, doubling each digit in the integer...
Result: 112233
Would you like to enter another integer (y/n): y
Enter an integer: 456
Number is even, tripling each digit in the integer...
Result: 444555666
Would you like to enter another integer (y/n): y
Enter an integer: 124
Number is even, tripling each digit in the integer...
Result: 111222444
Would you like to enter another integer (y/n): y
Enter an integer: 123
Number is odd, doubling each digit in the integer...
Result: 112233
Would you like to enter another integer (y/n): n
```

When the entered number is odd, you double each digit in the integer, below are a few examples:

Value	Result
0	0
13	1133
1357	11335577

When the entered number is even, you triple each digit in the integer, below are a few examples:

Value	Result
0	0
12	111222
124	111222444

You may assume that all of the integers entered (odd or even) will be >= 0.

Also notice that your program should ask whether another integer should be entered. If the response to the prompt is "y" your program should ask for another integer and perform the appropriate computations, if the user inputs an "n" the program should end. At this point you don't have to worry about incorrect input, when asked "Would you like to enter another integer (y/n): ", the input will always be a "y" or an "n" character.

Your program should be able to handle **an arbitrary number of integers.** In the example above there were only 4 integers processed, but you should be able to handle 10, 20, 50, or more different integers during the course of running your program without issue.

The Mod (%) Operator

The mod operator produces the remainder of an integer division, so 1 % 2 = 1, 2 % 2 = 0, and 3 % 2 = 1. One common use of the mod operator is determining whether a number is odd or even. If you mod **with 2** and the result is 1, the number is odd; if the result is 0, the number is even:

```
123 % 2 = 1   // Non-zero remainder, not even.
1234 % 2 = 0  // Zero remainder, even.
```

Further, let's say I want to get each of the digits in the integer in 1234, I can do that with the mod operator and integer division:

```
// Get the least significant digit.
1234 % 10 = 4
// Shave off one digit using division.
1234 / 10 = 123

// Get the next least significant digit.
123  % 10 = 3
// Shave off one digit using division.
123 / 10 = 12

// Get the next least significant digit.
12 % 10 = 2
// Shave off one digit using division.
12 / 10 = 1

// Get the next least significant digit.
1 % 10 = 1
// Shave off one digit using division.
1 / 10 = 0
```

Your program should able to handle an integer with any number of digits, so you should generalize this example with some sort of loop.

Plan of Attack
If you're relatively new to programming, don't try to write the entire program at once. Come up with a plan of attack, break the task down into smaller sub-problems, and test your work after each stage. Here's an example plan of attack:

1. Create your project (referring back to Homework 1's instructions if necessary), and remember **not to use spaces in the name.**
2. You will need several loops for this assignment. One loop will continue executing as long the user enters a `"y"` when asked whether they want to enter more integers. The loop will stop when the user enters `"n"` to the prompt.
3. After that, use a selection (`if`) statement to determine whether the integer entered is even or odd. Try printing out a different message for each possibility like in the sample program execution.
4. Determine how to double or triple each digit in the integer. This is another place that you will need a loop (nested loops actually). Take a look at the examples above, you will need the integer division and the mod operator.

Testing Your Program
You are expected to thoroughly test your program before you submit it for grading. Make sure you try both odd and even integers, also try small and large integers (no more than 4 digits). Make sure you test any "edge cases" you can think of and try processing multiple integers in a row.

Documentation and Style
Project 1 is worth **100 points**. When you submit Project 1, Web-CAT will test that your program works correctly and creates the proper output, if your program works correctly you will receive **80 points** from the automated testing provided by Web-CAT. The remaining **20** points are for style and documentation, which will be **manually graded by the TAs**, using the rubric posted on the course website.

What to Submit
For this assignment you should submit your `main.cpp` file from inside of your project directory.

This assignment will be graded automatically. Test your programs thoroughly before submitting them. Make sure that your programs produce correct results for every logically valid test case you can think of. Do not waste submissions on untested code, or on code that does not compile with the supplied code from the course website.

Web-CAT will assign a score based on runtime testing of your submission; your best score will be counted; the TAs will later verify that your best submission meets the stated restrictions, and assess penalties if not.

To submit this assignment:
1. Visit http://web-cat.cs.vt.edu in your web browser.

2. Enter your Virginia Tech PID and password in the appropriate fields on the log-in screen, and make sure that **Virginia Tech** is selected as the institution. Click **Login**.
3. The Web-CAT home screen will display useful announcements and assignments that are currently accepting submissions. Find the assignment that you want to submit in the table, and click the "Submit" button next to it.
4. Click the **Browse...** button and select the file you want to upload. The homework assignments and programming projects for this course should be self-contained in a single **main.cpp** file, so you can simply select that one file.
5. Click the **Upload Submission** button. The next page will ask you to review your selection to ensure that you have chosen the right file. If everything looks correct, click **Confirm**.

The next page will show that your assignment is currently queued for grading, with an estimated wait time. This page will refresh itself automatically, and when grading is complete you will be taken to a page with your results.

Pledge

Each of your program submissions must be pledged to conform to the Honor Code requirements for this course. Specifically, you **must** include the following pledge statement in the submitted file:

```
//    On my honor:
//
//    - I have not discussed the C++ language code in my program with
//      anyone other than my instructor or the teaching assistants
//      assigned to this course.
//
//    - I have not used C++ language code obtained from another student,
//      or any other unauthorized source, either modified or unmodified.
//
//    - If any C++ language code or documentation used in my program
//      was obtained from an allowed source, such as a text book or course
//      notes, that has been clearly noted with a proper citation in
//      the comments of my program.
//
//    <Student Name>
```

Failure to include this pledge in a submission will result in the submission being disallowed during code review.