Assignment 6 - Functions

Goals: understand and use functions in C++, including prototypes, function calls, and return values.

Task:

Finish the starter code. Use the existing function prototypes and descriptions to implement your functions, and **use function calls for each case in the switch statemen**t.

You'll write the functions below main and put your implementation in each function. This is a stark difference from our past assignments as main will contain very little.

Use the comments in the starter code to inform yourself on how to implement each function.
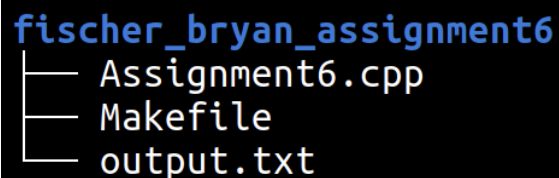
Submission:

Submit your completed Assignment6.cpp code, an output.txt file where you copy and paste the output of each case showing they all work correctly, along with the provided Makefile in a single zip folder named lastname_firstname_assignment6.zip to blackboard by the due date.

Your submission should look like this:

Inside a .zip folder named **lastname_firstname_assignment6.zip**
　　　Inside the .zip folder, another folder named **lastname_firstname_assignment6**
　　　　　Inside this folder: **Assignment6.cpp, output.txt,** and the provided **Makefile**

Provide at least 3 test cases for each function showing they work properly

<u>Rubric:</u>　　　**Total Points 75**

Code compiles - no errors or warnings	5 points (0 total points for non-compiling code)
Function calls in each switch statement - no implementation code inside the switch case blocks, only function calls	10 points
Function implementation -Print reverse: correctly prints the reversed number (10 points) -Odd Even Zero: correctly prints the number of even, odd, and zero numbers (10 points) -Print e: correctly approximates e depending on the value given to the function (15 points)	35 Points
Comments- sufficient comments provided for all implementations. This means roughly every 5-10 lines of code there should be at least one comment describing what is happening.	10 points
Correct folder names and folder hierarchy	6 points
output.txt file contains at least three tests for each function. Try to test "Edge cases" – think about what might break the program (cause it to crash), test it and show their program works.	9 points