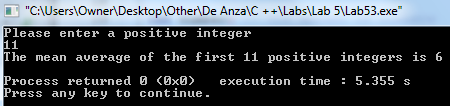
**Lab 5.3**

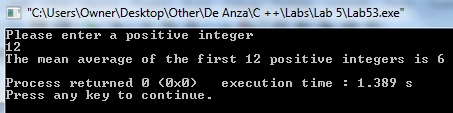
***Why is the typecast operator needed to compute the mean in the statement mean = static\_cast(float)(total)/value;?***

The typecast operator is needed to convert the integer total to a float. Otherwise the division would take place between two integers which would always get you an integer. But the program must be created to accept float type results in case of the mean having a fractional result.

***What do you think will happen if it is removed? Modify the code and try it. Record what happens. Make sure that you try both even and odd cases. Now put static\_cast<float> total back in the program. Include your comments and answers from above.***

By removing the typecasting code, the variable "mean" only accepts whole or integer values, not floating-point numbers. Therefore the value of mean will always be an integer (even when it divides even numbers). Here were the outputs with an odd and an even number when the typecasting was removed:

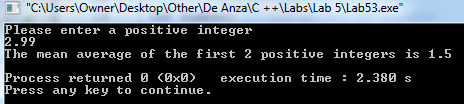




With typecasting, the mean (with an input like the number 12) would be 6.5, not 6.

***What happens if you enter a float such as 2.99 instead of an integer for value? Try it and record the results. Take a screen shot of the output.***

The output or result displayed is 1.5. The program is designed to have an integer as an input, not a float-point value. So a value such as 2.99 has its fractional part truncated and the program calculates the mean of 1 and 2 (eliminating the fractional part of 0.99). This was the output:



***Modify the code so that it computes the mean of the consecutive positive integers n, n+1, n+2, . . . , m, where the user chooses n and m. For example, if the user picks 3 and 9, then the program should find the mean of 3, 4, 5, 6, 7, 8, and 9, which is 6.***

I modified it a lot because I didn't want to leave room for invalid input. It does the job of obtaining the mean of two positive integers. Here's examples of two final output displays:

