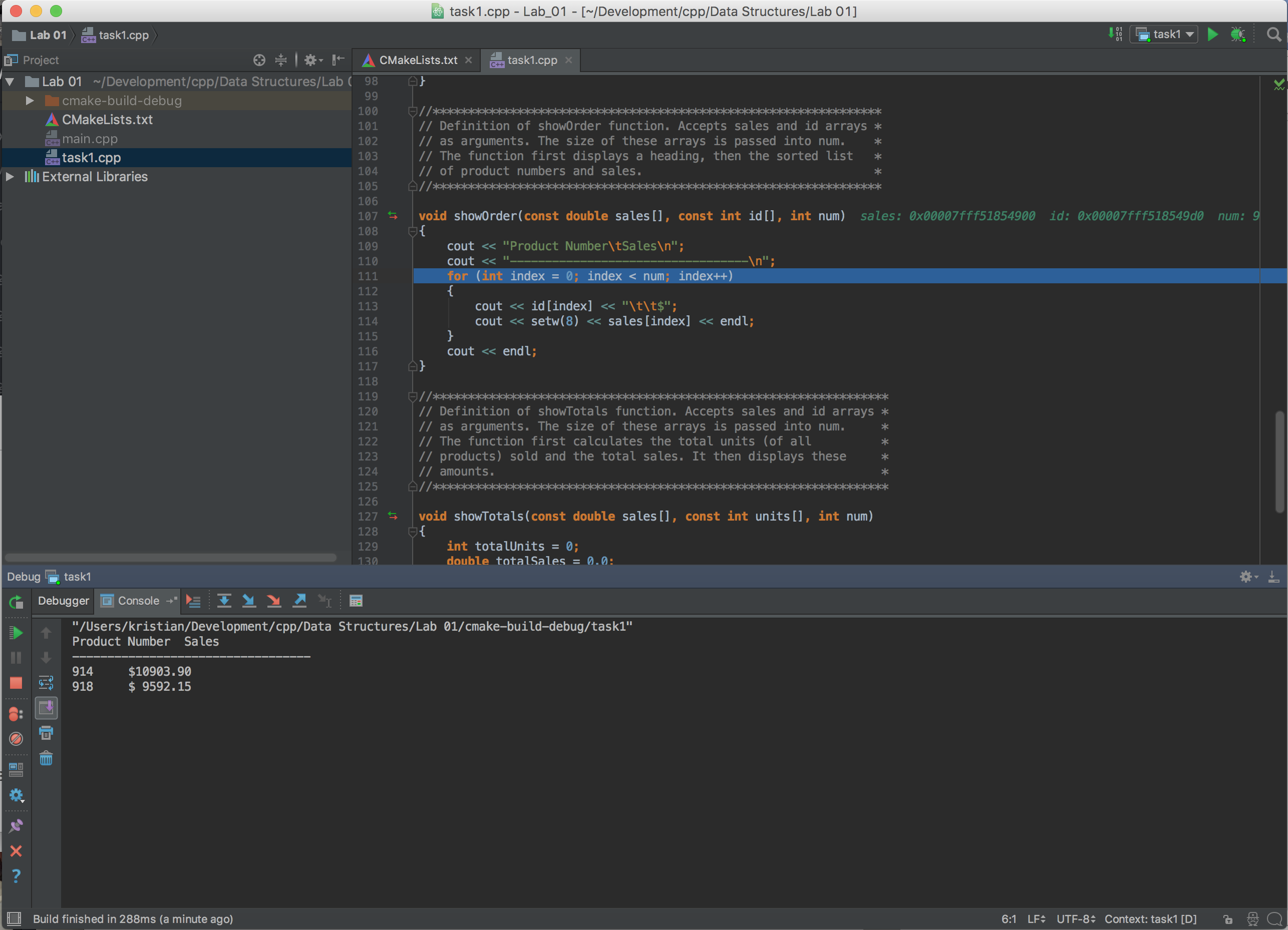
Kristian Snyder

Data Structures CS2021 Section 001

Prof. Charles Zimmer, Prof. Karthik Vadambacheri

Lab 01

**Central Concepts:**

The central concepts and methodologies in this lab included debugging, refactoring existing code, and understanding code others have written. These are central processes in all professions in computer science. Even while working alone, reading library documentation and debugging one’s own code are pervasive tasks. Having a competent knowledge of debugging is required to produce code at any typical scale, as increased structure and complexity usually introduces additional, more confusing errors. In this case, I was required in Task 1 to navigate the debugger in an IDE and step through the code I have run. This is the most basic form of it, but leads up to more complicated debugging concepts that could be covered in the future.

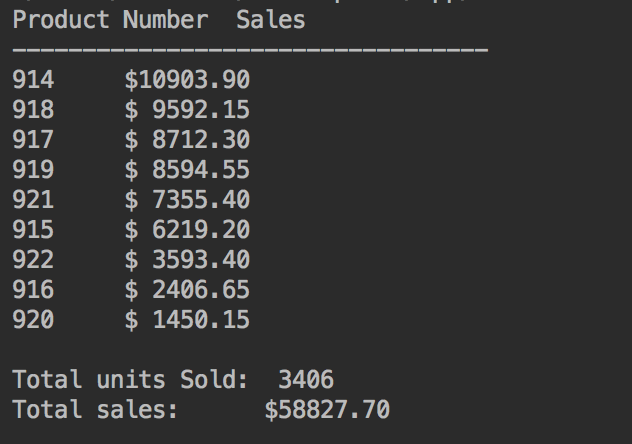
**Task 2:**

The first thing I checked after running the program—as instructed—was look at the output for any strange data, which I suspected included the division (leftmost column in the table) always being 4, no quarter 4 data except for one strange value, and no total sales.

In quarter 4 data, I looked at the display method and noticed that it wasn’t being displayed in the same indexing method that it was being stored in. This is an easy mistake to make when using 2D arrays; the programmer could have gotten the columns and rows mixed up when determining how to display them.

With respect to the division always being 4, I looked again at what statement was displaying the value and saw that it was the maximum value, which will be the same every time. I changed it to the loop counter + 1 to offset the 0 index. This was probably a mistake in parsing the variable names as the programmer should have recognized the NUM\_DIVS variable as a constant. However, excluding that mistake, the name div and NUM\_DIVS are similar enough that it is possible to miss.

Total sales displayed as 0 was simply an issue of total sales never being used. This should have been caught by the IDE as a warning or error because the variable was unused, but if the programmer wasn’t paying attention to the warnings then they could have cleared the code as correct.



**Task 3:**

The primary change in task 3 was to take the four arrays of data and merge them into an array of structures containing the same data. I initially considered rewriting the initializations of the arrays into one single statement for the entire array of Products (the struct name was Product), but instead I looped through the range of the arrays and added Products as I went to prevent mistyping. Most of the changes made to the individual functions revolved around changing the array access to selecting the Product at the same position and then getting the desired property. Initially, I was attempting to modify the Products passed to the functions by placing the new values into a separate array and returning it. I instead deferred to the initial implementation and instead modified the array in place. The only major bug I encountered was not updating my function definitions, which prevented the program from compiling on the first few attempts. I made sure to avoid changing as much code as possible. Most of the logical structure was already there so I normally only changed basic array accesses.