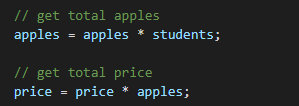
Lab 02: Integer Overflows

# Discovery

This program has several flaws including some which could result in integer overflows. Identify changes you would make and why.

1. Review the source code for the application of interest ( apples.c).
   1. Provide screenshots of vulnerable or otherwise flawed code segments.

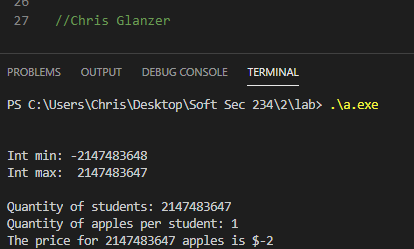




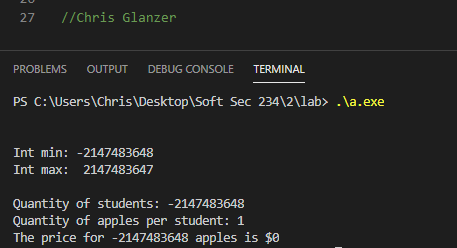
* 1. Explain each issue in your own words.

1. The variables should be declared as unsigned int, since you can’t sensibly have negative apples or students, and allowing either would result in a negative charge.
2. There are no checks against the multiplication operation meaning that we can overflow and wraparound the sign.
3. Compile and run the program
   1. Provide a screenshot demonstrating each of the vulnerabilities using a method of your choice

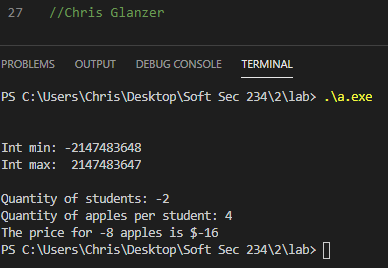
First is a wraparound:



Then an overflow



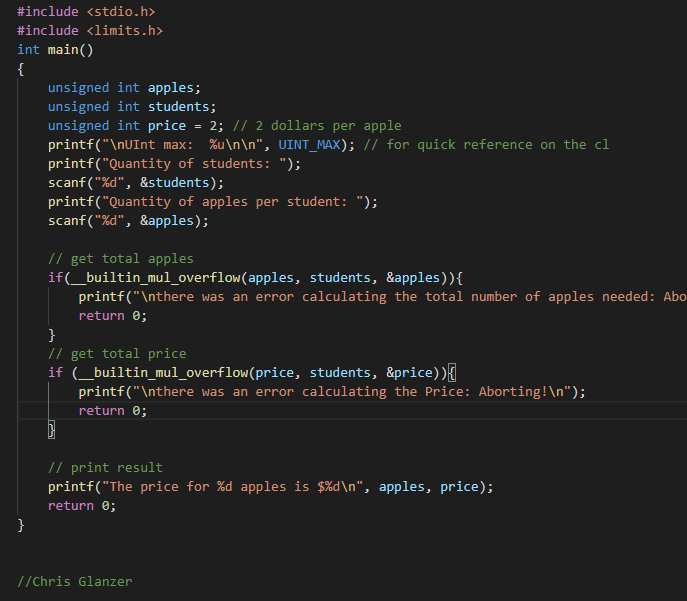
And finally, just a logic error resulting from signed input variables



# Remediation

After confirming the vulnerability, use your knowledge and available resources to modify the source to follow best practices and avoid integer overflows.

1. Modify the source code of apples.cto gracefully handle overflows.
   1. Document your resulting source code with a screenshot

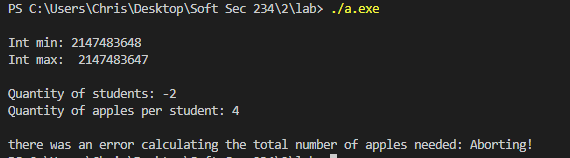


1. Test your changes
   1. Are your attempts at entering further malicious input successful? Why or why not?

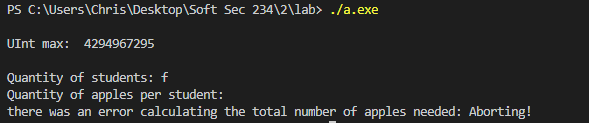
my attempts at entering further malicious input are unsuccessful, wraparounds are being managed by the unsigned int datatype, and overflows are being guarded against by a builtin function.

* 1. Provide screenshots documenting proper handling of invalid / malicious input.

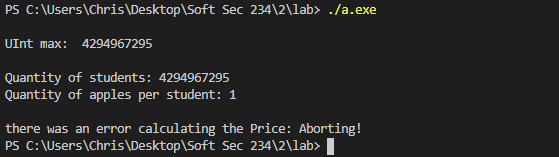
Negative values input (I didn’t update the code to show the UINT min/max until after this screenshot)



Non-numerical data: (this one actually surprised me. I thought that f and \n would be typecasted into integers - 13 and 70 and then multiplied. The resulting 910 is well within the range of an unsigned int but it still caught it.)



Overflow of UINT:



# Scoring

The rubric will be published at scoring time. Each portion of the assignment has the following points assigned.

|  |  |
| --- | --- |
| **Section Points** | |
| **1** | 6 |
| **2** | 2 |
| **3** | 6 |
| **4** | 6 |
| **Total** | **20** |