1. The algorithm does work as intended and multiplies two integer values together. It does so by looping the process of adding the first integer value (*a*) to the variable *product*. The while loop is looped the number of times of the value of the second integer value (*b*). First, the values of *a* and *b* are checked to see if they are 0. If they are, the product is automatically default. If neither of the two integer values are 0, the variables *count* and *product* are set to 0 and a while loop begins. Each instance the condition of the loop is true, the value of *a* is added to the variable *product* and the value of 1 is added to the variable *count*. The variable *count* is used as a counter to check with the value of *b*. Once the value of *count* equals or exceeds the value of *b*, the condition of the loop is now false and the algorithm exits the loop. The final value of *product* is printed.
2. The error is that the algorithm does not take into consideration the input of negative numbers. This is especially impactful when the second value, *b*, is negative. The algorithm will first check that neither *a* or *b*, is 0 but not if any are negative. then, it proceeds to add the value of *a* to *product*, the number of times required for *count* to reach *b*. Except for the case of a negative *b*, there will be additional iterations until *b* reaches 0. For example, if *b* is -3, there will be three additional times where *a* is added to *product*.