*/\* apple stocks - solution 1, brute force \*/*

*// declare profit, init to 0*

*// if the argument passed in is not an array, we return the profit*

*// otherwise, we loop through the apple stock.*

*// we declare a "buy stock" at the current el*

*// and if the el is not a number, we return 0.*

*// and we set an inner loop, starting at the 2nd el of the array*

*// and we set these elements to sellStock*

*// and we get the Math.max of either the profit or the buyStock - sellStock, assigning this value to the profit.*

*/\*\* solution 2, single iteration \*/*

*// if what's passed in is not an array, we return 0.*

*// otherwise, we initialize the profit to 0.*

*// and looping from the 2nd el from the array,*

*// the sellStock is initialized to this 2nd element*

*// and if it's not a number, we'll return 0.*

*// otherwise, if the buy price is greater than the sell stock*

*// if it's greater, we assign the buyStock to the sellStock - so we're trying to buy at the lowest price.*

*// and we assign the profit to the max of either the profit or the sell price less buy price.*

*/\*\* solution 3, single iteration \*/*

*// we set minIndex to 0 - where the minimum price occurrs.*

*// run array check, returning 0 ifit's not an array*

*// and reducing the array,*

*// if the 2nd el is less than the element at the price at the minIndex, we reset the minINdex to this current index.*

*// and we define sellStock at the current index, buyStock at the minIndex.*

*// and we define profit the be the difference between those.*

*// and if the profit is greater than the accumulator, we reassign the accumulator to the profit,*

*// reuturning the accumulator.*