Exercise 1

Write a function that performs exactly one pass of selection sort on any given list of integers.

Example:

```
selection_sort_one_pass([5, 2, 3, 4, 0, 1])
>>> [0, 2, 3, 4, 5, 1]
```

Exercise 2

The file large_list.py contains a list of $10^{4.6}$ tuples each containing 3 numbers. Write an algorithm that iterates through each tuple and creates a new list containing each tuple where the sum of index 0 and 1 equal index 2.

Sort the new list containing only valid tuples so that the last elements of the tuples are in ascending order. Use a sorting algorithm of your choice.

Example:

```
>>> tuples = [(0,0,1), (0,1,1), (0,1,2), (1,1,2), (1,2,3)]
>>> tuples = filter_tuples(tuples)
>>> selection_sort(tuples)
> [(0,1,1), (1,1,2), (1,2,3)]
```

Exercise 3

Write a function that can take in two strings as function arguments and determine if they are anagrams or not, and print the result in the console. An anagram is a word or phrase formed by rearranging the letters of a different word or phrase. Two words are anagrams of each other if they contain the same number of characters and the same amount of each character.

Examples of anagrams are as follows: abc = cba, ABBA = baba, Cat = Act, Astronomer = Moon starer, The Detectives = Detect Thieves, Funeral = Real fun, Listen = Silent.

You need to sort the characters in lexicographic order (alphabetical order), and determine if all the characters in one string are equal to and in the same order as all the characters in the other string. To allow for strings with upper case letters and phrases with spaces, you should convert the string values into lowercase and remove spaces before you sort the characters.

You are not allowed to do this the easy way and use inbuilt functions such as the sorted() function, you must utilize another sorting algorithm of your choice (Like bubble sort, selection sort, insertion sort etc).