Quiz 5
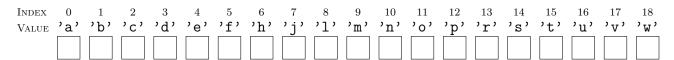
Name: _____

Question 1 (20 points). Write a Python function which performs a *linear* search on a list. Your function should have two parameters: the list and the value to find in the list. It should return the index of the element or `None` if no such element is found. (You may not call the `index` method on the list.)

Question 2 (15 points). In the boxes below, indicate the order in which a *binary* search would examine the following list of characters when looking for the character `'i'`. That is, locate the first value that the binary search would compare to `'i'` and place a **1** under that value. Then, place a **2** under the next value it would consider, and so on.

Index	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Value	'a'	'b'	'c'	'd'	'e'	'f'	'h'	'j'	'l'	'm'	'n'	'o'	'p'	'r'	's'	't'	'u'	'v'	'w'

Question 3 (15 points). Linear search has a complexity of $O(n)$ while binary search has a complexity of $O(\log n)$. If binary search is faster than linear search, why don't we just use binary search all the time?

Question 4 (30 points). The following is a flawed implementation of the Selection Sort algorithm. Recall that this algorithm performs a sort by finding the appropriate value for each index of the list and then swapping it into position. This implementation is broken into different functions for readability. There are four things wrong with the following implementation. Identify at least three of them and explain how they should be corrected.

```python
1  def swap(lst, i, j):
2    lst[i] = lst[j]
3    lst[j] = lst[i]
4
5  def find_smallest(lst):
6    smallest_idx = 0
7    for i in range(len(lst)):
8      if lst[smallest_idx] < lst[i]:
9        smallest_idx = i
10     return smallest_idx
11
12 def selection_sort(lst):
13   for i in range(len(lst)):
14     smallest_idx = find_smallest(lst)
15     swap(lst,i,smallest_idx)
```

Question 5 (20 points). Write a function which counts the number of words in the first four lines of a file. The function should accept the name of the file as a parameter and return an integer indicating the number of words. Here, a "word" is any sequence of non-space characters. You may assume that the file contains at least four lines.