Name: _____ Date: _____

Pledge: _____

Consider the implementation of quickselect below:

```
int lomuto_partition(int array[], int left, int right) {
   int p = array[left], s = left;
   for (int i = left + 1; i <= right; ++i) {
       if (array[i] < p) {
           ++s;
           swap(array[s], array[i]);
       }
   }
   swap(array[left], array[s]);
   return s;
}

int quick_select(int array[], size_t left, size_t right, size_t k) {
    size_t s = lomuto_partition(array, left, right);

    if (s == k - 1) {
        return _____(a)_____;
    }
    if (s > k - 1) {
        return quick_select(array, left, _____(b)_____, k);
    }
    return quick_select(array, _____(c)_____, right, _____(d)_____);
}

int quick_select(int array[], const size_t length, size_t k) {
    return quick_select(array, 0, length - 1, k);
}
```

1) Fill in the 4 blanks in the quickselect algorithm. (4 points)
   a) _____
   b) _____
   c) _____
   d) _____

2) Show the array [4, 5, 6, 4, 0] after running **`lomuto_partition`**. (3 points) _____

3) Suppose we are sorting an array of eight integers using **`quicksort`** with **`lomuto_partition`** and have just finished the first call to **`lomuto_partition`**. The array now looks as follows:

   11 4 20 45 32 60 98 70

   Which value or values could have been the pivot? (1 point) _____

4) Suppose mergesort were to cut the array into 3 evenly sized subarrays (instead of 2) and did a 3-way merge after making the recursive calls.
   a) Write the recurrence relation for this modified version of mergesort. (1 point)

   $$T(n) = \_\_\_\_\_$$

   b) Use the Master Theorem to determine its complexity. (1 point, answer depends on correct 4a, indicate base for logarithm)

   $$T(n) \in \theta(\_\_\_\_\_)$$