Quiz: Sorting Algorithms

Example

Given the partial bubble sort function below, fill in the blank (a) to complete the sorting algorithm. Use a.1 for sorting by the book's title and a.2 for sorting by the number of pages.

```
typedef struct {
    char title[100];
    int pageNumbers;
} Book;
void bubbleSort(Book array[], int n) {
    for (————————————————(b)) {
        for (int j = 0; j < n - i - 1; j++) {
            if (————————————————————————(a)) {
                Book temp = array[j];
                array[j] = array[j + 1];
                array[j + 1] = temp;
            }
        }
    }
}
```

Answer

- a.1:`strcmp`[1]`(array[j].title, array[j + 1].title) >0`,
- a.2: `array[j].pageNumbers >array[j + 1].pageNumbers`.
- `b.  int i = 0; i < n - 1; i++`

Exercises

Using the example of the Bubble Sort algorithm as a guide, perform similar modifications to the following sorting algorithms:

1. Quick Sort
2. Merge Sort

```
void merge(Book array[], int l, int m, int r) {
    int n1 = m - l + 1;
    int n2 = r - m;
    Book L[n1], R[n2];
    for (int i = 0; i < n1; i++)
        L[i] = array[l + i];
    for (————————————————(c))
        R[j] = array[m + 1 + j];
    int i = 0, j = 0, k = l;
    while (i < n1 && j < n2) {
        if (————————————————(d)) {
            array[k] = L[i];
            i++;
        } else {
            array[k] = R[j];
            j++;
        }
        k++;
    }
    while (i < n1) {
        array[k] = L[i];
        i++;
        k++;
    }
    while (————————(e)) {
        array[k] = R[j];
        j++;
        k++;
    }
}
\rest of merge sort algorithm is here
int partition(Book array[], int low, int high)
{
    Book pivot = array[high];
    int i = low - 1;
    for (————————————————————(f)) {
        if (————————————————————————(g) {
            i++;
            Book temp = array[i];
            array[i] = array[j];
            array[j] = temp;
        }
    }
    Book temp = array[———(h)];
    array[i + 1] = array[———(k)];
    array[high] = temp;
    return (i + 1);
}
\rest of quick sort algorithm
```

Coding

Please give me the algoirthm of swap that suits this example

[1]The `strcmp` function in C is used to compare two strings. It returns an integer less than, equal to, or greater than zero if the first string is found to be less than, to match, or be greater than the second string, respectively. For example, strcmp("Ab","Ac") is -1, strcmp("Ac","Ab") is 1, strcmp("Aa","Aa") is 0.

solution

Merge Sort Solution

- c: `int j = 0; j < n2; j++`
- d.1: `strcmp(L[i].title, R[j].title) < 0`
- d.2: `L[i].pageNumbers < R[j].pageNumbers`
- e: `j < n2`

Quick Sort Solution

- f: `int j = low; j < high; j++`
- g.1: `strcmp(array[j].title, pivot.title) < 0`
- g.2: `array[j].pageNumbers < pivot.pageNumbers`
- h: `i + 1`
- k: `high`

Swap Code

```
void swap(Book *a, Book *b) {
    Book temp = *a;
    *a = *b;
    *b = temp;
}
```