2) (10 pts) DSN (Sorting)

Consider sorting students, where each student has a first name, last name and a unique ID number. Assume all names contain uppercase letters only, so that strcmp does an alphabetic comparison for the purposes of this problem. Complete the code below so that it sorts students by last name (A to Z), breaking ties by first name (A to Z), and finally breaking ties between students with identical first and last names by ID number (smallest to largest). For example, if we have the students (EMILIO SANCHEZ 17), (ANDREA SANCHEZ 22), and (EMILIO SANCHEZ 10), the correct ordering would be ANDREA SANCHEZ, followed by EMILIO SANCHEZ 10, with EMILIO SANCHEZ 17 last.

```
typedef struct student {
    char first[20];
    char last[20];
    int ID;
} student;

void sort(student** list, int len) {
    int i,j;
    for (i=len-1; i>0; i--) {
        for (j=0; j<i; j++) {
            if (cmp(list[j], list[j+1]) > 0) {
                student* tmp = list[j];
                list[j] = list[j+1];
                list[j+1] = tmp;
            }
        }
    }
}

int cmp(student* a, student* b) {

    if (strcmp(a->last, b->last) != 0)
        return strcmp(a->last, b->last);

    if (strcmp(a->first, b->first) != 0)
        return strcmp(a->first, b->first);

    return a->ID - b->ID;
}
```

Grading: 3 pts for breaking ties properly by last name, 3 pts for breaking ties properly by first name, 4 pts for breaking ties by ID. If no string functions are used but the logic is correct via regular relational operators, then take off 4 pts total for not properly calling the functions.