2) (10 pts) DSN (Linked Lists)

The function below is given two sorted, singly linked lists, where each node contains a 2D coordinate (x, y). The lists are sorted by the x-coordinate (increasing order) first and then by the y-coordinate (increasing order). Complete the following user defined function `merge2DCoordinates` so that it merges the two sorted linked lists into one sorted linked list, preserving the order of the coordinates. In particular, your code should NOT allocate or free any memory. Instead, your code should simply change where some of the existing pointers (`next`) are pointing and return a pointer to the front of the updated list. For full credit, **write your code recursively.** (Note: The recursive solution is also shorter to write.)

```
typedef struct node_s {
    int x;
    int y;
    struct node_s* next;
} node_t;

node_t * merge2DCoordinates (node_t* ptr1, node_t* ptr2) {

    if (ptr1 == NULL && ptr2 == NULL ) return NULL;    // 1 pt
    if (ptr1 == NULL) return ptr2;                     // 1 pt
    if (ptr2 == NULL) return ptr1;                     // 1 pt

    // 3 pts if statement condition to split into 2 cases.
    if (ptr1->x < ptr2->x || (ptr1->x == ptr2->x && ptr1->y < ptr2->y)) {

        // 2 pts for call, connect and return.
        ptr1->next = merge2DCoordinates(ptr1->next, ptr2);
        return ptr1;
    }

    // 2 pts for call, connect and return.
    ptr2->next = merge2DCoordinates(ptr1, ptr2->next);
    return ptr2;
}
```