
Exercise Sheet $n^{\circ}3$: Linked Lists

In exercises 1 to 8, a singly linked list is defined by:

```
typedef struct Node {  
    int data;  
    struct Node *next;  
} List;
```

Exercise 1. Write a function to compute the sum of the elements in a linked list. Implement both an iterative and a recursive version. The recursive function should follow the prototype:
`void sumNodes(List *head, int *sum);`

Exercise 2. Implement a recursive function to find the maximum value in a linked list. Provide two versions with the following prototypes:

- `int maxList(List *head);`
- `void maxList(List *head, int *max);`

Exercise 3. Write the iterative and recursive versions of a function that searches for an element X in a linked list. The function should return a pointer to the node containing X if found, or NULL if X is not present in the list.

Exercise 4. Write a recursive function that reverses a linked list.

Exercise 5. Given a list of integers sorted in ascending order, write a function to insert an element X so that the list remains sorted in ascending order.

Exercise 6. Write a function to remove all occurrences of an element X from a list.

Exercise 7. Given two sorted linked lists, L1 and L2, write a function to merge them into a single sorted list.

Exercise 8. Write a function that splits a linked list into two separate lists: one containing all the odd elements and the other containing all the even elements.

Exercise 9. Given a circular linked list of integers, write a function to remove the first element of the list.

Exercise 10. Write a function to append an element to the end of a doubly linked list.