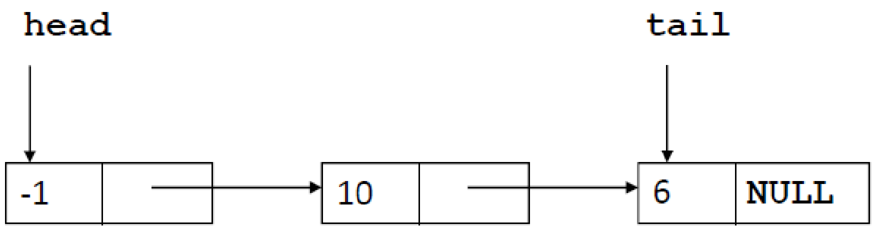
Singly Linked Lists, Pointers and Structures

B.1 Specification

Refer to the lecture nodes for the description of a singly linked list. In our implementation, the linked list stores non-negative integers. A dummy node with the data value -1 is used to simplify insertions and deletions. See the diagram below. `head` and `tail` are global variables.

Write a C program to implement the search, insertion and deletion operations.



B.2 Implementation

- The program to be submitted is named `list.c`. **Use the given template `list.c`** and fill in your code. **Submit only file `list.c`.**
- You are also given a file named `listMain.c` to test your code. Do not submit file `listMain.c`.
- Implement the following functions: `insertFirst()`, `insertLast()`, `removeFirst()`, and `search()`. See file `list.c` for their specifications.
- Function `insertFirst()`: The new element is to be inserted at the front of the list, right after the dummy node. If a new node cannot be created (e.g., insufficient memory), the function calls function `prtError()` to display an error message and returns NULL. Otherwise, it returns the pointer to the newly created node.
- Function `insertLast()`: The new element is to be inserted at the end of the list. If a new node cannot be created (e.g., insufficient memory), the function calls function `prtError()` to display an error message and returns NULL. Otherwise, it returns the pointer to the newly created node.
- Function `removeFirst()`: If the list is empty (i.e., no element other than the dummy node), the function calls function `prtError()` to display an error message and returns -1. Otherwise, it removes the first element (i.e., the node right behind the dummy node) and returns the data (integer) of the removed node.

- Function `search()`: If there is an element containing non-negative integer `k` then return the pointer to that element. Otherwise, return NULL. If there is more than one element containing `k`, return the pointer to the first encountered element.
- You may define your own variables inside the above functions.
- In file `list.c` you are given three utility functions: `init()`, `prtError()` and `prtList()`. DO NOT modify these functions.
- Do not modify the function and structure definitions in file `list.c`.

B.3 Sample Inputs/Outputs

See file `listMain_out.txt` for the output from running programs `list.c` and `listMain.c`.

Common Notes

- Complete the header in file `list.c` with your student and contact information.
- Assume that all inputs are non-negative integers.
- Do not use any C library function except malloc(), calloc(), free(), scanf() and printf().
- To compile both files `list.c` and `listMain.c`, use the following command:

  ```
  cc list.c listMain.c
  ```
- Submit only file `list.c`.
- Do not submit file `listMain.c` (the grader already has this file!).