CS 101 LAB #9 DATA STRUCTURES

Mythili Vutukuru

IIT Bombay

Reference: "C How to Program", Deitel and Deitel, 8th Edition, Chapter 10,12

GOAL OF LAB #9

- In this lab, you will write code to understand the following concepts:
  - Performing basic operations on linked lists
  - Using linked lists with recursion
- Write and submit code for any two programs
- Please write your code neatly, using proper indentations and comments.

LINKED LISTS: INSTRUCTIONS

- For all problems in this lab, you have to work with a linked list of integers or characters.
- Outside of the main function, you can define a structure that holds one integer/character data item, and contains a pointer to the next structure element. This structure can be used to solve all problems in this lab.
- Declare any pointers to linked lists (start pointer, or anything else) as global variables outside main
- Inside main, you can use randomly generated data or ask data for input on what data to insert/delete into the linked list
- Use separate functions to insert/traverse or perform any other operations on your linked lists, and invoke these functions from main
- Use malloc and free to allocate/release memory for linked list nodes

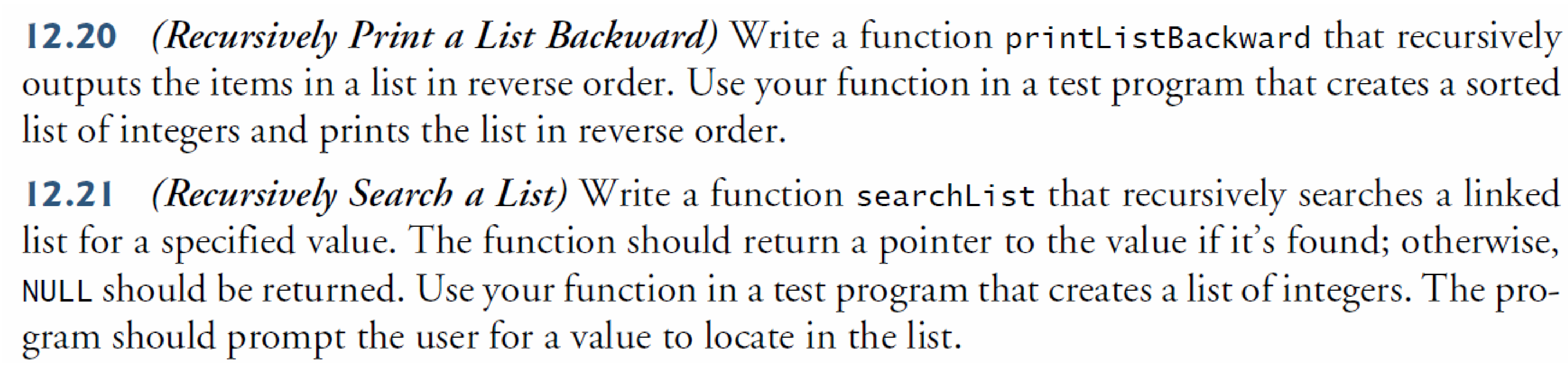
LINKED LISTS: BASIC OPERATIONS

- Write a program that inserts 25 random integers between 1 and 100 (both inclusive) into a linked list in **unsorted** order. Use a separate function to perform the insert operation and invoke it from main. Next, write a function to traverse the linked list and return its maximum item. Invoke this function from main and print the returned value in main.
- Write a program that inserts 25 random integers between 1 and 100 (both inclusive) into a linked list in **sorted** order. Use a separate function to perform the insert operation and invoke it from main. Next, write a function to traverse the linked list and return the average value of all elements. Invoke this function from main and print the returned value in main.

LINKED LISTS: HANDLING MULTIPLE LISTS

- Write a program that has two linked lists of 10 integers each. You can add any integers of your choice to the list. Note that the program will have two separate start pointers to the two lists. First, print both lists one after the other. Next, use a function to concatenate the two lists, where the items of the second list are inserted at the end of the first list. After this operation, print the first list again.
- Write a program that has two sorted linked lists of 10 integers each. You can add any integers of your choice to the list in sorted order. First, print both lists one after the other. Next, use a function to merge the two lists in sorted order. This function should merge elements from both lists into a new third linked list. After this operation, print the final merged linked list.

LINKED LISTS AND RECURSION



12.20 *(Recursively Print a List Backward)* Write a function `printListBackward` that recursively outputs the items in a list in reverse order. Use your function in a test program that creates a sorted list of integers and prints the list in reverse order.

12.21 *(Recursively Search a List)* Write a function `searchList` that recursively searches a linked list for a specified value. The function should return a pointer to the value if it's found; otherwise, `NULL` should be returned. Use your function in a test program that creates a list of integers. The program should prompt the user for a value to locate in the list.