Department of Information Systems and Technologies

CTIS 152 – Data Structures and Algorithms
FALL 2024 - 2025

Lab Guide #21 – Week 14-1

OBJECTIVE : Linked List Exercises

Instructor : Burcu LIMAN
Assistant : Berk ÖNDER, Engin Zafer KIRAÇBEDEL

Q1. The purpose of the program is to obtain a **sorted linked list** consisting of words.

Implement the necessary functions for a linked list in the **linkedList_str.h** header file.

Write the following functions;

- **createList** gets an input text file pointer as a parameter, reads the words from the text file and inserts each word to the correct position in the linked list. After each insertion, the content of the list will be displayed. The function returns the initial address of the linked list. **Note:** Write a search function to find the correct position in the linked list.
- **removeWord**: gets the initial address of the linked list and the word to be deleted as parameters, deletes all words which start with the given string in the linked list. The function also returns **1** or **0** depending on the success of the delete operation.

Write a C program that reads the words from "**words.txt**", create and display the content of the sorted linked list as in the example run. The program also reads the word to be deleted, from the user, deletes the nodes containing this word from the list. The updated list will be displayed at the end of the program.

Project Name: LG21_Q1
File Name: Q1.cpp

words.txt

```
acceptance
breeze
apple
backhand
accept
zoo
backward
bicycle
back
playback
```

Example Run #1:

```
acceptance->NULL
acceptance->breeze->NULL
acceptance->apple->breeze->NULL
acceptance->apple->backhand->breeze->NULL
accept->acceptance->apple->backhand->breeze->NULL
accept->acceptance->apple->backhand->breeze->zoo->NULL
accept->acceptance->apple->backhand->backward->breeze->zoo->NULL
accept->acceptance->apple->backhand->backward->bicycle->breeze->zoo->NULL
accept->acceptance->apple->back->backhand->backward->bicycle->breeze->zoo->NULL
accept->acceptance->apple->back->backhand->backward->bicycle->breeze->playback->zoo ->NULL

Enter a string to delete: back
accept->acceptance->apple->bicycle->breeze->playback->zoo->NULL
```

Example Run #2:

```
acceptance->NULL
acceptance->breeze->NULL
acceptance->apple->breeze->NULL
acceptance->apple->backhand->breeze->NULL
accept->acceptance->apple->backhand->breeze->NULL
accept->acceptance->apple->backhand->breeze->zoo->NULL
accept->acceptance->apple->backhand->backward->breeze->zoo->NULL
accept->acceptance->apple->backhand->backward->bicycle->breeze->zoo->NULL
accept->acceptance->apple->back->backhand->backward->bicycle->breeze->zoo->NULL
accept->acceptance->apple->back->backhand->backward->bicycle->breeze->playback->zoo ->NULL

Enter a string to delete: accept
```

```
apple->back->backhand->backward->bicycle->breeze->playback->zoo->NULL
```

Example Run #3:

```
acceptance->NULL
acceptance->breeze->NULL
acceptance->apple->breeze->NULL
acceptance->apple->backhand->breeze->NULL
accept->acceptance->apple->backhand->breeze->NULL
accept->acceptance->apple->backhand->breeze->zoo->NULL
accept->acceptance->apple->backhand->backward->breeze->zoo->NULL
accept->acceptance->apple->backhand->backward->bicycle->breeze->zoo->NULL
accept->acceptance->apple->back->backhand->backward->bicycle->breeze->zoo->NULL
accept->acceptance->apple->back->backhand->backward->bicycle->breeze->playback->zoo ->NULL

Enter a string to delete: book
There is NO word which starts with the string <book>
```

Q2. Write a C program that creates the linked list, reads several characters from the user until '**!**' is entered, stores them in the linked list and displays the content of the list. Then, it deletes the first occurrence of the given character from the list. You are supposed to create a **linkedList_char.h** file by writing the following functions;

getNode, addAfter, addBeginning, deleteFirst, deleteAfter

Write the following functions;

- **createList:** that gets the initial address of the linked list, reads several characters from the user until '**!**' is entered and stores them in the linked list. The function returns the initial address of the list.
- **removeChar:** that gets the initial address of the linked list and the character to be deleted as parameters and deletes the **first** occurrence of the character from the linked list. The function returns the initial address of the list. Also returns the found info, if function can delete the character returns 1 otherwise return 0.
- **displayRec:** that displays all the characters recursively in the linked list.

Project Name: LG21_Q2
File Name: Q2.cpp

Example Run #1:

```
Enter a character (or ! to stop): w
Enter a character (or ! to stop): e
Enter a character (or ! to stop): c
Enter a character (or ! to stop): w
Enter a character (or ! to stop): a
Enter a character (or ! to stop): n
Enter a character (or ! to stop): w
Enter a character (or ! to stop): !

w -> e -> c -> w -> a -> n -> w ->  NULL


Enter a char to delete : w

e -> c -> w -> a -> n -> w ->  NULL
```

Q3. There are 3 types of courses in the gym. They are listed below according to the their ids;

1 -> Aerobics
2 -> Zumba
3 -> Pilates

Write a C program that reads a text file named **"courses.txt"** into an array of structure linked list. The program should create a linked list for every type of courses and fill these linked lists according to their ids. Then, it will displays the content of the linked list array which includes the courses' linked lists in the correct order.

Note: Initial address (i.e, head pointer) of each linked list must be stored in an array.

Write the following functions;

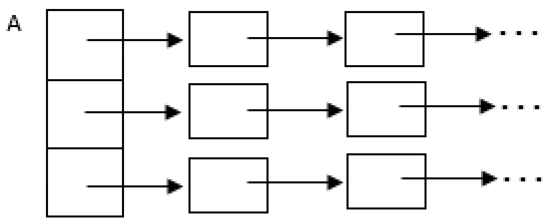
- **`createList:`** gets the file and a linked list array as parameters for reading the text file and fills the linked lists according to the courseIDs. The linked list will keep the courses in reverse order (each node will be added to the beginning of the linked list).
- **`displayList:`** gets a linked list as a parameter to display its' content.

Project Name: LG21_Q3
File Name: Q3.cpp

Courses.txt

```
1 Abby Kohn : 3 months
3 Michael Pearce : 1 year
2 Dante Tomaselli : 1 month
2 Jeremy Dyson : 1 week
3 Alexandros Avranas : 3 months
1 Kay Cannon : 1 month
3 Timothy McNeil : 1 month
2 John Krasinski : 1 year
1 Holderman : 3 months
1 Raja Gosnell : 1 year
3 Dominic Cooke : 3 months
2 Leigh Whannell : 3 months
2 Ari Aster : 3 months
3 Paul Schrader : 1 year
1 Tim Kirby : 1 week
```

HINT: `node_t * A[3];`



Example Run:

```
Aerobics
****************************************
   1  Tim Kirby          1 week     ->
   1  Raja Gosnell       1 year     ->
   1  Holderman          3 months   ->
   1  Kay Cannon         1 month    ->
   1  Abby Kohn          3 months   ->
NULL

Zumba
****************************************
   2  Ari Aster          3 months   ->
   2  Leigh Whannell     3 months   ->
   2  John Krasinski     1 year     ->
   2  Jeremy Dyson       1 week     ->
   2  Dante Tomaselli    1 month    ->
NULL

Pilates
****************************************
   3  Paul Schrader      1 year     ->
   3  Dominic Cooke      3 months   ->
   3  Timothy McNeil     1 month    ->
   3  Alexandros Avranas 3 months   ->
   3  Michael Pearce     1 year     ->
NULL
```