COMP9024 Assignment One

Objectives

- Give you experience with doubly linked lists
- Understand how to solve set union, set intersection and longest sublist problems
- Give you practice with time complexity analysis
- Give you further practice with C and data structures

Admin

Marks 12 marks. Marking is based on the correctness and efficiency of your code. Your code must be well commented.

Group? This assignment is completed individually

Due Time 10:00 pm Sunday, March 14, 2021.

Late Submissions Late submissions will not be accepted!

Aims

In this assignment, you will implement a set of functions based on the doubly linked list defined in MyDLList.c.

The functions you need to implement are shown as follows:

1. DLList *CreateDLListFromFileDlist(const char *filename). This function creates a doubly linked list of integers by reading all integers from a text file named filename, and returns a pointer to the doubly linked list where each node stores one integer. Assume that adjacent integers in the file filename and the standard input are separated by one or more white space characters or a new line character.

   If filename is "stdin", CreateDLListFromFileDlist ("stdin") creates a doubly linked list by reading all integers from the standard input and the word end denotes the end of input.

   This function must check for invalid input. In case of invalid input, it prints an error message "Invalid input!" and returns NULL. (**2 marks**)

2. void printDLList(DLList *u). This function prints all the elements (integers) of a doubly linked list pointed by u in the order from the first node to the last node in the list on the standard output, one element per line. (**1 mark**)

3. DLList *cloneList(DLList *u). This function creates an identical copy of a doubly linked list u and returns a pointer to the list cloned. (**1 mark**)

4. DLList *longestSublist(DLList *u). This function returns a doubly linked list that is the longest sublist of u. The longest sublist v of a list u is defined as follows:

a. The greatest common divisor of all the elements (integers) of v is not 1 (we consider positive divisors only).
b. No other sublist of u contains more elements than v.

Assume that all the integers in the list pointed by u are distinct.

Consider a list u = {2, -6, 8, 15, 11,23, 20}. The longest sublist v of u is {2, -6, 8, 20}, and the greatest common divisor of all the elements of v is 2.

Note that the longest sublist of a list may not be unique. Consider a list u = {2, 6, 4, 3, 9}. There are two longest sublists: v1={2, 6, 4} and v2={6, 3, 9}. If there are multiple longest sublists, this function returns any one of them.

When analysing the time complexity of your algorithm, you may assume that the maximum integer of the input list is a constant. Under this assumption, it takes constant time O(1) to check if two integers are mutually prime (i.e., their only divisor is 1). (**3 marks**)

5. DLList *setUnion(DLList *u, DLList *v). This function computes the union of the two sets of integers that are stored in the doubly linked lists pointed by u and v, respectively, and returns a pointer to the doubly linked list that stores the union. Each element (int) of a set is stored in a node of the corresponding doubly linked list.

   Given two sets A and B, the union of A and B is a set that contains all the distinct element of A and B. For example, assuming that A = {2, 8, 5, 7} and B = {5, 9, 6, 7}, A U B = {2, 8, 5, 7, 9, 6}. Note that in a set, all the integers are not necessarily sorted. (**2 marks**)

6. DLList *setIntersection(DLList *u, DLList *v). This function computes the intersection of the two sets of integers that are stored in the doubly linked lists pointed by u and v, respectively, and returns a pointer to the doubly linked list that stores the intersection. Each element (int) of a set is stored in a node of the corresponding doubly linked list. (**2 marks**)

   For simplicity, you may assume that all the elements of each input set are distinct for both set union and set intersection. Therefore, you do not need to check if a set contains duplicates.

   Given two sets A and B, the intersection of A and B is a set that contains all the elements of A that are also in B. For example, assuming that A = {2, 8, 5, 7} and B = {5, 9, 6, 7}, A ∩ B = {5, 7}.

7. void freeDLList(DLList *u). This function frees the space occupied by all the nodes of the doubly linked list pointed by u. (**1 marks**)

Time complexity analysis: For each function, you need to analyze its time complexity in terms of big-O notation and put your analysis as comments immediately before the code of the function. You may assume that each built-in I/O function such as printf(), malloc() and free(), takes O(1) time.

How to submit your code?

1. Go to the assignment one page

2. Click on Assignment Specfication
3. Click on Make Submission
4. Submit your MyDLList.c file that contains all the code.

Plagiarism

This is an individual assignment. Each student must work out his/her own solution without help from other people. In particular, it is not permitted to exchange code or pseudocode. You are not allowed to use code developed by persons other than yourself. All work submitted for assessment must be entirely your own work. We regard unacknowledged copying of material, in whole or part, as an extremely serious offence. For further information, see the Course Information.