ITP 20001-01 Data Structures, Spring 2023

Midterm Exam – Part 1. Programming.

Problem 1 (50 points). Let's think about writing an expression of string operations in a postfix notation. A string expression is one of the following three cases:

(1) a string,
(2) a string expression followed by an integer number between 1 and 10, or
(3) two string expression followed by the "+" operator.

A string expression followed by an integer number is evaluated to repeating the string of the given expression for the given number of times. Two string expressions followed by the "+" operator is evaluated to the concatenation of the strings of the two expressions. For examples:

- **`abc def +`** represents **`abcdef`**
- **`aa bb + cc +`** represents **`aabbcc`**
- **`a 3`** represents **`aaa`**
- **`a b 2 + 2`** represents **`abbabb`**

You are asked to complete function `char * eval (char **a, int length)` in `llist/strexpr.c`, which receives a string expression as the string array `a`, and then returns the resulting string. You must satisfy the following programming requirements in writing the `eval` function:

- Allocate a `llist` object and use it as a stack,
- Assume an input is always valid, and the length of an input string array `length` is no more than 16. Also, assume that element consists of at most 8 characters,
- The `eval` function must return a newly allocated string. The `eval` function must not result any memory leak.

Submission instruction: Upload `llist/strexpr.c` to the Problem 1 entry at the submission site.

Problem 2 (50 points). Implement `void llist_sort (llist * l, int (* elem_cmp)(void * e1, void * e2))` in `llist/llist.c`, which sorts the elements of a doubly linked list `l` in ascending order. A function given as `elem_cmp` is to determine the ordering of two objects `e1` and `e2`: `elem_cmp` returns -1 if `e1` precedes `e2`, return 0 if `e1` is equivalent with `e2`, or return 1 if `e2` precedes `e1`.

`llist_sort` must not invoke `llist_insert` or `llist_remove` functions. Also, `llist_sort` must not create a new `llist_node` object. You can assume that no invalid input is given.

Submission instruction: Upload `llist.c` to the Problem 2 entry at the submission site.

Problem 3 (25 points). You can find a circular list implementation in the `clist` directory. Function int `clist_resize (clist * l, size_t capacity)` is to change the capacity of the container array `arr` to `size` if the number of elements is less than `size`.

Complete the body of the `clist_resize` function to satisfy the following requirements:

- This function must use the `realloc` function to change the size of the container array `arr`,
- Return 1 if the function succeeds. Otherwise, return 0.

Submission instruction: Upload `clist.c` to the Problem 3 entry at the submission site.