CS110: Computer Programming Lab Department of CSE IIT, Guwahati

Module 04 Stage 02 Exercise 32

Assessment exercises are designed to help us check if the student has learned the basics of the topics included in the drill instructions. However, a drill assessment is not a comprehensive assessment. A fuller and complete assessment aimed at determining the course grades will be done through CS110 examinations.

It is expected that the student will attempt and solve many more exercises from the drill assessment sets to improve their programming skills and for an excellent performance at the examinations.

Exercise

In this exercise, you are required to implement methods listed in interface `sortedList`. The list containing the entries is always sorted in ascending (increasing) order of key values.

If a new entry is added with the same key as an entry already in the sorted list, the new entry replaces the previous entry.

1. Write test-first approach based function `main()` to verify the correctness of the implementation.
2. Implement methods in interface `sortedList` .
3. Verify the correctness of your implementation using the previously written test suite.

Since every method in this exercise requires a search over the list of `list_elem` from its start, you need only hold a single start link in the main header node of object `sortedList`. This link will point to the first (smallest key) `list_elem`. Also, you will need to hold a member (attribute) in `struct list_elem` to hold the object's `key`.

To avoid a need for special actions when the first or the last entry (`list_elem`) is added or removed, you must implement the class such that method `makList()` creates the empty list with two special `list_elem` in the list. Their keys will be `LONG_MAX` and other `LONG_MIN`. Obviously, the list users will not be permitted to use these two values as keys in their applications.

Interface sortedList

```
#ifndef LIST_H_INCLUDED
#define LIST_H_INCLUDED

/* Returns a reference to a new list */
void * mkList(void);

/* Place a new entry */*/
void insert (void * listP, void * objP, long key);

/* Returns/removes reference to object with key */
void * remove (void * listP, long key);

/* Returns reference/removes first object in list */
void * first (void *);

#endif // LIST_H_INCLUDED
```