```
#ifndef LINKED_STACK_H_
#define LINKED_STACK_H_

#include <cstddef>    // includes defintions for size_t and NULL

namespace KW {

  template<typename Item_Type>
  class stack {

  public:
    stack() : top_of_stack(NULL), num_items(0) { }

    size_t size() const {
      return num_items;
    }

    bool empty() const {
      return num_items == 0;
    }

    void push(const Item_Type& item) {
      top_of_stack = new Node(item, top_of_stack);
      num_items++;
    }

    Item_Type& top() {
      return top_of_stack->data;
    }

    const Item_Type& top() const {
      return top_of_stack->data;
    }

    void pop() {
      Node* old_top = top_of_stack;
      top_of_stack = top_of_stack->next;
      delete old_top;
      num_items--;
    }

    // MISSING HOUSEKEEING FUNCTIONS

    private:

      struct Node {
        Item_Type data; /** The data */
        Node* next;     /** The pointer to the next node. */

        Node(const Item_Type& data_item, Node* next_ptr = NULL) :
          data(data_item), next(next_ptr) {}
      };


      // data members
      Node* top_of_stack;  /** A pointer to the top of the stack */
      size_t num_items;

  }; // End class stack

} // End namespace KW
#endif
```