```
#ifndef STACK_H
#define STACK_H

/* This module uses the GC of Boehm.  Thus, there is no need to
   specify memory allocation and deallocation.  Memory is freed
   when it is no longer referenced.  The push and pop operations
   are strictly imperative in that the alter the stack.  We use
   uniform reference semantics, so all references to the same
   stack are affected by imperative operations. */

struct stack;
typedef struct stack *stack;

extern stack stack_create(void);

extern void stack_push(stack s, void *element);

extern int stack_empty(stack s);

/* Precondition: the stack must not be emtpy,
   as reported by stack_empty() */

extern void stack_pop(stack s);

/* Precondition: the stack must not be emtpy,
   as reported by stack_empty() */
extern void *stack_top(stack s);

/* Return a new, independent stack with the
   same elements as s */
extern stack stack_copy(stack s);


#endif
```