```c
#include <stdio.h>
#include <stdlib.h>

int syntax() { printf("syntax: lister arg+\n") ; return 1 ; }

/* singly linked list data structure */
typedef struct List_s {
    const char*     data ;
    struct List_s*  next ;
} List_t, *List_p,**List_h;

List_p List_reverse(List_h h)
{
    List_p p = *h;
    if   ( p ) {
        List_p prev = NULL;
        while (p)
        {
            List_p temp = p->next;
            p->next     = prev   ;
            prev        = p      ;
            p           = temp   ;
        }
        *h = prev;
    }
}

void List_print(const char* msg,List_p p)
{
    if ( p ) {
        printf("%s: ",msg) ;
        List_p node = p ;
        while (node) {
            printf("%s ",node->data) ;
            node = node->next ;
        }
        printf("\n") ;
    }
}

void List_free(List_h h)
{
    List_p p = *h ;
    if  (  p ) {
        List_p node = p ;
        while (node) {
            void* temp = node ;
            node = node->next ;
            free(temp) ;
        }
        *h = NULL ;
    }
}

int main(int argc, const char* argv[])
{
    int i ;
    List_p list = NULL ;
    /* create a list from argv */
    for ( i = 1 ; i < argc ; i++ ) {
        List_p node = (List_p) malloc(sizeof(List_t)) ;
        if ( node ) {
            node->data  = argv[argc-i] ;
            node->next  = NULL ;
            if ( list ) {
                node->next = list ;
            }
            list = node ;
        }
    }

    List_print  ("forward: " ,list) ;
    List_reverse(&list) ;
    List_print  ("reverse: " ,list) ;
    List_free   (&list) ;

    return argc < 2 ? syntax() : 0 ;
}
```