#include <stdio.h>

#include <stdlib.h>

/\* self-referential structure \*/

struct Node {

int item;

struct Node \*next;

};

struct List {

struct Node \*head;

struct Node \*tail;

};

struct List SLL\_new() {

/\* construct an empty list \*/

struct List list;

list.head = NULL;

list.tail = NULL;

return list;

}

int SLL\_length(struct List \*list) {

/\* return the number of items in the list \*/

struct Node \*p;

int n = 0;

for (p = list->head; p != NULL; p = p->next) {

++n;

}

return n;

}

int SLL\_empty(struct List \*list) {

/\* return true if the list contains no items \*/

return list->head == NULL;

}

int SLL\_pop(struct List \*list) {

/\* remove and return the first item of the list \*/

struct Node \*node = list->head;

int item = node->item;

list->head = node->next;

if (SLL\_empty(list)) {

list->tail = NULL;

}

free(node);

return item;

}

void SLL\_clear(struct List \*list) {

/\* remove all the items from the list \*/

while (!SLL\_empty(list)) {

SLL\_pop(list);

}

}

void SLL\_push(struct List \*list, int item) {

/\* insert the item at the front of the list \*/

struct Node \*node = malloc(sizeof(struct Node));

node->item = item;

node->next = list->head;

if (SLL\_empty(list)) {

list->tail = node;

}

list->head = node;

}

void SLL\_append(struct List \*list, int item) {

/\* append the item to the end of the list \*/

if (SLL\_empty(list)) {

SLL\_push(list, item);

}

else {

struct Node \*node = malloc(sizeof(struct Node));

node->item = item;

node->next = NULL;

list->tail->next = node;

list->tail = node;

}

}

int main() {

int i;

struct List list = SLL\_new();

for (i = 0; i < 5; ++i) {

SLL\_push(&list, i);

SLL\_append(&list, i);

}

while (!SLL\_empty(&list)) {

printf("%d\n", SLL\_pop(&list));

}

return 0;

}