#include <stdlib.h>

#include <string.h>

#include "list.h"

#include "utils.h"

#include "option\_list.h"

list \*make\_list()

{

list\* l = (list\*)xmalloc(sizeof(list));

l->size = 0;

l->front = 0;

l->back = 0;

return l;

}

/\*

void transfer\_node(list \*s, list \*d, node \*n)

{

node \*prev, \*next;

prev = n->prev;

next = n->next;

if(prev) prev->next = next;

if(next) next->prev = prev;

--s->size;

if(s->front == n) s->front = next;

if(s->back == n) s->back = prev;

}

\*/

void \*list\_pop(list \*l){

if(!l->back) return 0;

node \*b = l->back;

void \*val = b->val;

l->back = b->prev;

if(l->back) l->back->next = 0;

free(b);

--l->size;

return val;

}

void list\_insert(list \*l, void \*val)

{

node\* newnode = (node\*)xmalloc(sizeof(node));

newnode->val = val;

newnode->next = 0;

if(!l->back){

l->front = newnode;

newnode->prev = 0;

}else{

l->back->next = newnode;

newnode->prev = l->back;

}

l->back = newnode;

++l->size;

}

void free\_node(node \*n)

{

node \*next;

while(n) {

next = n->next;

free(n);

n = next;

}

}

void free\_list\_val(list \*l)

{

node \*n = l->front;

node \*next;

while (n) {

next = n->next;

free(n->val);

n = next;

}

}

void free\_list(list \*l)

{

free\_node(l->front);

free(l);

}

void free\_list\_contents(list \*l)

{

node \*n = l->front;

while(n){

free(n->val);

n = n->next;

}

}

void free\_list\_contents\_kvp(list \*l)

{

node \*n = l->front;

while (n) {

kvp\* p = (kvp\*)n->val;

free(p->key);

free(n->val);

n = n->next;

}

}

void \*\*list\_to\_array(list \*l)

{

void\*\* a = (void\*\*)xcalloc(l->size, sizeof(void\*));

int count = 0;

node \*n = l->front;

while(n){

a[count++] = n->val;

n = n->next;

}

return a;

}