Университет ИТМО, кафедра ВТ

Лабораторная работа №5 по Языкам Системного Программирования

Работу выполнил

студент группы P3200

Рогов Я. С.

Преподаватели:

Жирков И.О.

Балакшин П.В.

Санкт-Петербург, 2016

Задание: реализовать функции высшего порядка (map, foreach, foldl, iterate) для связного списка. Основная программа должна выполнить следующее:
1. Принять со входа числа и сохранить их в связном списке
2. Вывести исходный список с помощью foreach дважды, разделяя пробелами и переносами на новую строку соответственно
3. Вывести полученные с помощью map квадраты и кубы чисел
4. Вывести полученные с помощью foldl минимум, максимум, и сумму списка.
5. Вывести полученные с помощью map_mut модули чисел
6. Вывести полученные с помощью iterate список степеней числа 2 (первые 10)

inputnumber.c
```
#ifndef _INPUTNUMBER_C_
#define _INPUTNUMBER_C_ value

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

static const char* const urbad = "Correct number,
please\n";

static char inputnumber(int* num){
	static char buffer[128];
	char * endp = buffer;
	char endc;

	while(1){
		endc = scanf("%s", buffer);
		*num = strtol(buffer, &endp, 10);

		/* if buffer parsed and ends
correctly */
		if(endp!=buffer && *endp==0)
			break;
		else if(EOF==endc)
			return endc;

		printf(urbad);
	}
	return endc;
}

#endif
```

map.c
```
#include "llist.h"
#include "map.h"

void foreach(llist_t **phead, fptr_void__int_t
func){
	llist_t *lp = *phead;
	while(lp){
		(*func)(lp->value);
		lp=lp->next;
	}
}

llist_t* map(llist_t **phead, fptr_int__int_t
func){
	llist_t *result = llist_create((*func)
( (*phead)->value ));
	llist_t *lp = (*phead)->next;
	llist_t *lp_result = result;

	while(lp){
		lp_result->next =
llist_create( (*func)(lp->value) );
		lp_result = lp_result->next;
		lp = lp->next;
	}

	return result;
}
```

```
void map_mut(llist_t **phead, fptr_int__int_t
func){
	llist_t *lp = *phead;
	while(lp){
		lp->value = (*func)(lp->value);
		lp = lp->next;
	}
}

int foldl(llist_t **phead, fptr_int__int_int_t
func, int acc){
	llist_t *lp = *phead;
	while(lp){
		acc = (*func)(acc, lp->value);
		lp = lp->next;
	}
	return acc;
}

llist_t* iterate(int initv, size_t length,
fptr_int__int_t func){
	llist_t *result, *lp;
	if(length<=0)
		return NULL;
	lp = result = llist_create(initv);

	for(length-=1; length>0; length--){
		initv = (*func)(initv);
		lp->next = llist_create(initv);
		lp = lp->next;
	}
	return result;
}
```

program.c

```
#include <stdio.h>
#include <limits.h>

#include "map.h"
#include "llist.h"

#include "inputnumber.c"

int inc_int(int v){ return v+1; }

void print_int_space(int v){ printf("%d ", v); }
void print_int_newline(int v){ printf("%d\n", v);
}

int sqr_int(int v){ return v*v; }
int cube_int(int v){ return v*v*v; }

int sum_int(int a, int b){ return a+b; }
int max_int(int a, int b){ return a>b? a : b; }
int min_int(int a, int b){ return a<b? a : b; }

int abs_int(int a){ return a<0? -a: a; }
int mul2_int(int a){ return 2*a; }

static const char* const hello = "Your numbers,
please:\n";

static llist_t* input_list(){
        llist_t *llist;
        int num;

        printf(hello);

        if(EOF==inputnumber(&num)){
                printf("\nYou're laconic\n");
                exit(1);
        }
        llist = llist_create(num);

        while(EOF != inputnumber(&num))
                llist_add(num, &llist);
        return llist;
}

int main(){

        llist_t *list = input_list();
        llist_t *ret_list;

        printf("\n");

        /* foreach */

        foreach(&list, &print_int_space);
        printf("\n");
        foreach(&list, &print_int_newline);

        /* map */

        printf("\nElements' square:\n");
        ret_list = map(&list, &sqr_int);
        foreach(&ret_list , print_int_space);
        printf("\n");
        llist_free(&ret_list);

        printf("\nElements' cube:\n");
        ret_list = map(&list, &cube_int);
        foreach( &ret_list, print_int_space);
        printf("\n");
        llist_free(&ret_list);

        printf("\n");
        /* foldl */

        printf("Sum of the list: %d\n",
                        foldl(&list, &sum_int, 0));
        printf("Max element of the list: %d\n",
                        foldl(&list, &max_int,
list->value));
        printf("Min element of the list: %d\n",
                        foldl(&list, &min_int,
list->value));

        /* map_mut */

        printf("\nElements' absolute value:\n");
        map_mut(&list, &abs_int);
        foreach(&list, &print_int_space);
        printf("\n\n");

        /* iterate */

        printf("First 10 powers of 2:\n");
        ret_list = iterate(2, 10, &mul2_int);
        foreach(&ret_list, &print_int_newline);
        printf("\n");
        llist_free(&ret_list);

        llist_free(&list);

        return 0;
}
```

Вывод: в ходе выполнения данной лабораторной работы я ознакомился с использованием указателей на функции в C, а также ознакомился с функциями высшего порядка и написал их реализацию для связных списков.