**江西师范大学计算机信息工程学院学生实验报告（6）**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **专业：** | **数据科学与大数据技术2班** | **姓名：** | **赖丽婷** | **学号：** |  | **日期：** | **2012-3-16** |

|  |  |  |  |
| --- | --- | --- | --- |
| 课程名称 | 数据结构 | 实验室名称 | 计算机综合实验室 |
| 实验名称 | 函数 | | |
| 指导教师 |  | 成绩 |  |

1. **实验目的**（结出本次实验所涉及并要求掌握的知识点）
2. 学会squlist.h头文件的调用，利用前后对应的数组下标将a数组进行倒置
3. 通过%2判断奇偶将数组奇偶分离
4. 当不知道循环何时结束时，用while循环会方便很多

**2.实验内容**（结出实验内容具体描述）

**3.算法描述及实验步骤**（用适当的形式表达算法设计思想与算法实现步骤）

**6.**（程序源代码等）

实验内容[4]程序源代码：

1

#include <stdio.h>

#include <stdlib.h>

#include "test.h"

void reverse(sequence\_list \*L){

int i, j, mid;

j = L->size - 1;

for(i = 0; i <= (int) L->size/2 - 1;){

if(i != j){

mid = L->a[i];

L->a[i++] = L->a[j];

L->a[j--] = mid;

}

}

}

int main()

{

sequence\_list L;

input(&L);

reverse(&L);

print(&L);

return 0;

}

2

void sprit(sequence\_list \*L1, sequence\_list \*L2, sequence\_list \*L3){

int i, j, k;

j = 0;

k = 0;

for(i=0; i < L1->size; i++){

if((L1->a[i])%2 == 0){

L3->a[k++] = L1->a[i];

}else{

L2->a[j++] = L1->a[i];

}

}

L2->size = j;

L3->size = k;

/\*这一步很关键，除了初始化没有地方会对size进行处理\*/

}

int main()

{

sequence\_list L1, L2, L3;

input(&L1);

//input(&L2);

//merge(&L1,&L2,&L3);

sprit(&L1, &L2, &L3);

print(&L1);

print(&L2);

print(&L3);

printf("%d",L3.a[L3.size-1]);

return 0;

}

3

void merge(sequence\_list \*L1, sequence\_list \*L2, sequence\_list \*L3){

int i, j, k;

i=j=k=0;

L3->size = 0;

//当某个链表数据遍历完后，就跳出循环

while(L1->a[i]&&L2->a[j]){

if(L1->a[i]<L2->a[j]){

L3->a[k++]=L1->a[i++];

}else{

L3->a[k++]=L2->a[j++];

}

}

//不知道是那个先遍历完所以要进行条件判断

while(L1->a[i]){

L3->a[k++]=L1->a[i++];

}

while(L2->a[j]){

L3->a[k++]=L2->a[j++];

}

L3->size = L1->size +L2->size;

}

int main()

{

sequence\_list L1, L2, L3;

input(&L1);

input(&L2);

merge(&L1,&L2,&L3);

//input(&L1);

//sprit(&L1, &L2, &L3);

print(&L1);

print(&L2);

print(&L3);

printf("%d",L3.a[L3.size-1]);

return 0;

}

4

#include <stdio.h>

#include <stdlib.h>

#include "sequlist.h"

void inter(sequence\_list \*la, sequence\_list \*lb, sequence\_list \*lc){

int i, j, k;

i=j=k=0;

lc->size = 0;

for(i = 0; i < la->size; i++){

for(j = 0; j < lb->size; j++){

if(la->a[i]==lb->a[j]){

lc->a[k++]=la->a[i];

break;

}

}

}

lc->size=k;

}

int main()

{

sequence\_list la, lb, lc;

input(&la);

input(&lb);

inter(&la, &lb, &lc);

print(&lc);

return 0;

}

5

void partion(sequence\_list \*L){

int i, j, k, n;

i=j=k=0;

for(i=0; i < L->size; i++){

if(L->a[i]%2==0){

j=i;

for(k = i+1; k < L->size; k++){

if(L->a[k]%2!=0){

n = L->a[j];

L->a[j] = L->a[k];

L->a[k] = n;

break;

}

}

}

}

}

int main()

{

sequence\_list L;

input(&L);

partion(&L);

print(&L);

return 0;

}