《数据结构》课程实践报告

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 院、系 | 计算机学院 | | 年级专业 | 21计科 | 姓名 | 彭光 | 学号 | 2127405052 |
| 实验布置日期 | | 2022-09-27 | | 提交  日期 | 2022-10-23 | | 成绩 |  |

课程实践实验4：线性表的实现与运用

## 问题描述及要求

琳琳上大学后将她每天的消费支出项一行一项写在她的一个文本文件中，她的每一支出项目记录了支出日期、支出项目和金额。几个月过去了，她想统计一下她的消费行为信息，请使用自定义单链表类，设计相应程序，帮助她完成相应的查询和统计。

具体功可以自行设计，以下功能供参考：

（1）从文本文件读入所有n项支出项目；

（2）求出这n个支出项目中的最小、最大消费；每一天的平均消费；

（3）按照日期找出某一天的所有花费；

（4）按照项目找出该支出项目的所有花费。比如要求给出在“学习用品”这一项上总共花了多少钱；

（5）按照支出项花费递减的顺序输出每一项的对应总花费。

## 概要设计

对实验内容的理解：本次实验主要考察对单链表类的使用和如何利用单链表类来实现功能。

系统的功能列表：

0代表退出程序

1代表读取文件

2代表求最小花费

3代表求最大花费

4代表求每一天的平均花费

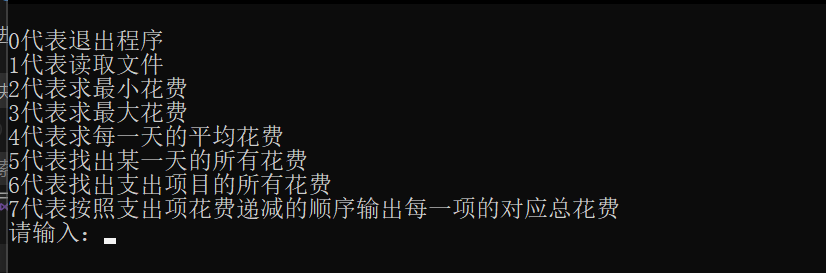
5代表找出某一天的所有花费

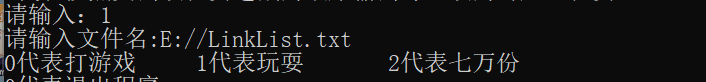
6代表找出支出项目的所有花费

7代表按照支出项花费递减的顺序输出每一项的对应总花费

**程序运行的界面设计：**

首先会出现屏幕提示，跳出一段功能列表以供选择：

如果输入1，那就请用户输入文件名，然后会跳出文件里的所有支付项目的名字和所对应的下标，如图：



在选择功能5的时候会要求输入日期。在选择功能6的时候，会要求选择下标（这里的下标代表含义在一开始出现的屏幕提示里）。

总体设计思路：

首先设计一个可以读取文档的readfile函数，利用fin.good()函数来进行文件是否可以读取。如果不可以读取，就输出文件打开失败，如果可以，就将文件里的数据按行读取，并且存成project结构体。就放在data数据域里。接下来我还设计了一个get\_all\_out函数，来得到所有的支付项目，来进行输出（避免中文乱码的情况）。

再根据功能列表里的2来设计get\_min函数，求支付项目里花费最少的金额。这里面我先设置头指针里为最小值，接下来用p->data.money < min\_double->data.money一直比较，来求得最小值。接下来的get\_max函数也是用了和get\_min函数一样的原理。

接下来利用get\_average()函数来实现计算每一天的平均花费这一功能4.因为是计算每一天的平均花费，所以应该把日期存在一个列表里，然后通过循环依次查找。我创建了一个arry向量，用来存储日期。我设计了一个not\_in\_vector的bool函数来判断日期是否在里面，如果不在，就push进向量中。然后我通过双重循环，利用count\_1,count\_2这两个来进行计数，算出对应的平均花费，把它输出出来。

接下来用get\_money\_by\_date函数实现功能5：输出某一天的具体花费，就是简单的循环和字符串匹配。

用get\_money\_by\_out函数来实现功能6：输出在某一支付项目上的所有花费，和前面的get\_money\_by\_date函数设计差不多。

用reverse\_allspend\_by\_out函数来实现功能7：按照支出项花费递减的顺序输出每一项的对应总花费。我创建了一个结构体向量，利用里面的结构体来进行排序，因为功能6就是求在某支付项目上的全部花费，所以我调用了功能6的函数来存在结构体中。

## 详细设计

主函数设计：

先创建了一个单链表l和一个string向量x。然后用while循环来进行输出，先给出功能列表，然后根据屏幕提示进行输入。如果输入0，就退出程序。

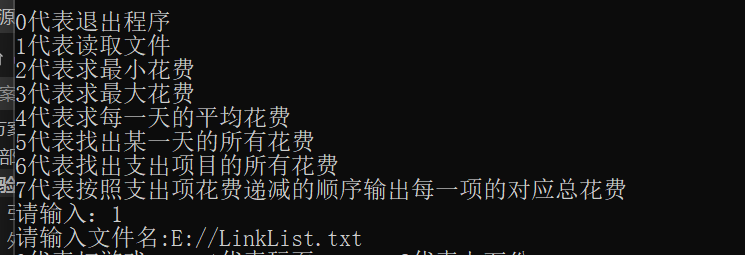
关键算法的设计：

最重要的部分是数据域里存结构体，如果默认了这样的操作，后面就不能再用DataType，因为这样会产生系统错乱，会不断地报错，所以我索性就都用具体的数据类型来进行存储。

用reverse\_allspend\_by\_out函数来实现功能7：按照支出项花费递减的顺序输出每一项的对应总花费。我创建了一个结构体向量，利用里面的结构体来进行排序，因为功能6就是求在某支付项目上的全部花费，所以我调用了功能6的函数来存在结构体中。

## 实验结果

测试输入：



测试目的：测试是否可以实现功能1

正确输出：

屏幕截图 2022-10-23 114638

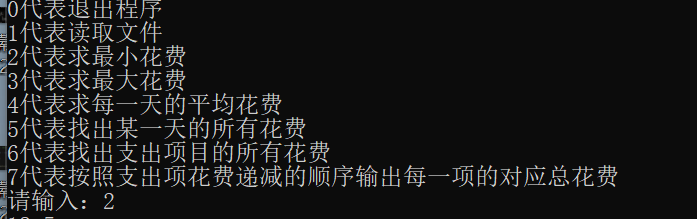
实际输出：

屏幕截图 2022-10-23 114638

错误原因：无

测试结论：通过

测试输入：



测试目的：测试是否可以通过功能2

正确输出：

屏幕截图 2022-10-23 114817

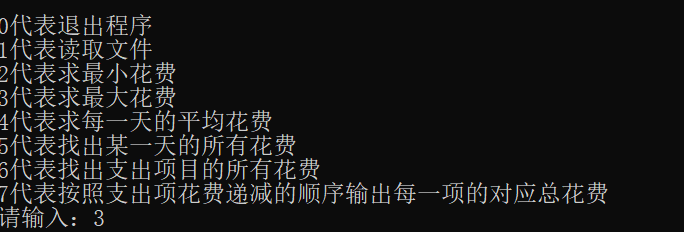
实际输出：

屏幕截图 2022-10-23 114817

错误原因：无

测试结论：通过

测试输入：



测试目的：测试是否可以实现功能3

正确输出：

屏幕截图 2022-10-23 114918

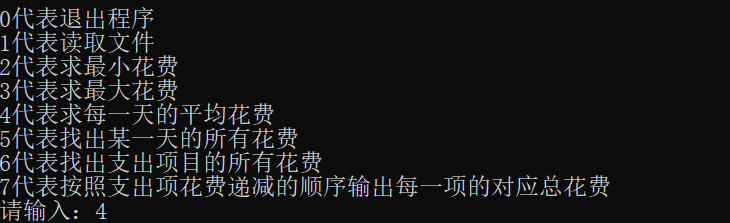
实际输出：

屏幕截图 2022-10-23 114918

错误原因：无

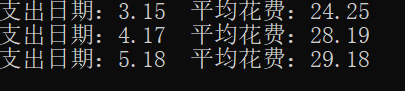
测试结论：通过

测试输入：

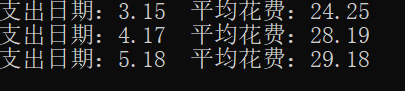


测试目的：测试是否可以实现功能4

正确输出：



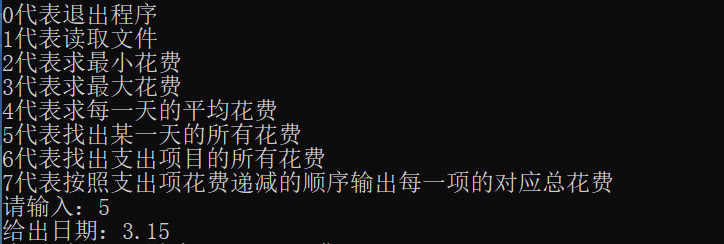
实际输出：



错误原因：无

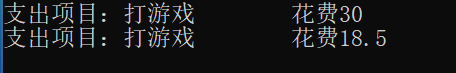
测试结论：通过

测试输入：

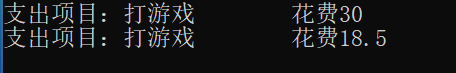


测试目的：测试是否可以实现功能5

正确输出：



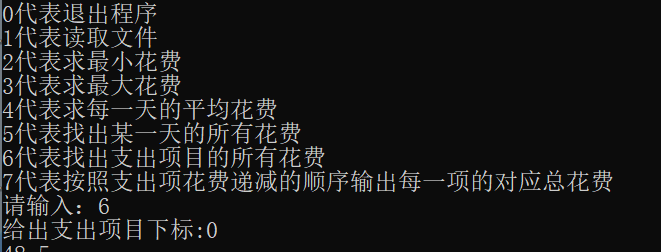
实际输出：



错误原因：无

测试结论：通过

测试输入：



测试目的：测试是否可以实现功能6

正确输出：

屏幕截图 2022-10-23 115336

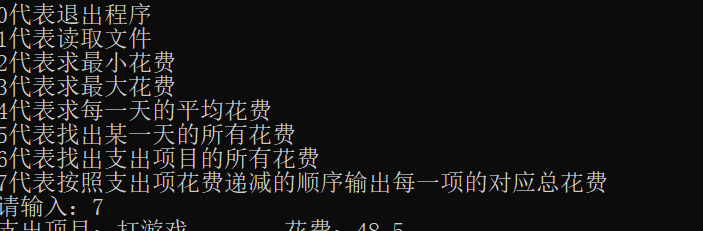
实际输出：

屏幕截图 2022-10-23 115336

错误原因：无

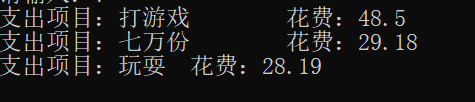
测试结论：通过

测试输入：

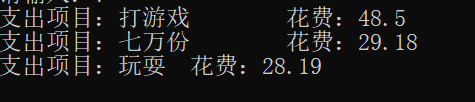


测试目的：测试是否可以实现功能7

正确输出：



实际输出：



错误原因：无

测试结论：通过

## 实验分析与探讨

测试结果的分析：

测试策略：测试每个功能是否可以正常的实现和运行。

时间复杂度O（n^2）

空间复杂度O（n）

结论：我觉得空间复杂度可以控制在O（n）

遇到的问题：

1. 中文乱码问题
2. DataType类型匹配系统报错

解决方法：

1. 将文件的编码格式设置为ANSI，就可以了
2. 将template<typename DataType>全部删除

## 小结

完善的方向：我觉得可以再继续增加一些功能，可以通过这个单链表来写成一个表格文档，这样看会更加直观。

完成的内容：要求上的所有内容

补充说明：文件编码格式必须是ANSI

## 附录：源代码

**1.实验环境：Visual Studio 2022**

2、

1. LinkList.h

#pragma once

#include<iostream>

#include<fstream>

#include<stdlib.h>

#include<algorithm>

#include<vector>

using namespace std;

struct project

{

string date;//支出日期

string out;//支出项目

double money;//支出金额

};

struct Node

{

struct project data;

Node\* next;

};

class LinkList

{

public:

void readfile(string &filename);

LinkList();

double get\_min();

double get\_max();

void get\_average();

void get\_money\_by\_date(string & s);

double get\_money\_by\_out(string& s);

void reverse\_allspend\_by\_out();

vector<string> get\_all\_out();

Node\* first;

};

bool not\_in\_vector(string& s, vector<string>& x)

{

for (int i = 0; i < x.size(); i++)

{

if (x[i] == s)

return false;

}

return true;

}

bool compare(project a, project b)

{

return a.money >= b.money;

}

LinkList::LinkList()

{

first = new Node;

first->next = nullptr;

}

void LinkList::readfile(string &filename)

{

Node\* p = first;

Node\* s;

struct project temp;

ifstream fin(filename);

if (fin.good() == false)

{

cout << "文件" << filename << "打开失败" << endl;

exit(0);

}

while (fin.eof() == false)

{

fin >> temp.date >> temp.out >> temp.money;

if (fin)

{

s = new Node;

s->data.date = temp.date;

s->data.out = temp.out;

s->data.money = temp.money;

p->next = s;

p = s;

p->next = nullptr;

}

}

fin.close();

}

double LinkList::get\_min()

{

Node\* min\_double = first->next;

auto p = first->next;

while (p != nullptr)

{

if (p->data.money < min\_double->data.money)

{

min\_double = p;

}

p = p->next;

}

return min\_double->data.money;

}

double LinkList::get\_max()

{

Node\* max\_double = first->next;

auto p = first->next;

while (p != nullptr)

{

if (p->data.money > max\_double->data.money)

{

max\_double = p;

}

p = p->next;

}

return max\_double->data.money;

}

void LinkList::get\_average()

{

Node\* p1 = first->next;

vector<string> arry;

arry.clear();

arry.push\_back(p1->data.date);

p1 = p1->next;

while (p1 != nullptr)

{

if (not\_in\_vector(p1->data.date, arry))

{

arry.push\_back(p1->data.date);

}

p1 = p1->next;

}

Node\* p ;

double count\_1,count\_2;

for (int i = 0; i < arry.size(); i++)

{

p = first->next;

count\_1 = 0, count\_2 = 0;

while (p != nullptr)

{

if (p->data.date == arry[i])

{

count\_1 += p->data.money;

count\_2++;

}

p = p->next;

}

cout << "支出日期：" << arry[i] << "\t";

cout << "平均花费：" << count\_1 / count\_2 << endl;

}

}

void LinkList::get\_money\_by\_date(string& s)

{

Node\* p = first->next;

while (p != nullptr)

{

if (p->data.date == s)

{

cout << "支出项目：" << p->data.out << "\t花费" << p->data.money << endl;

}

p = p->next;

}

}

double LinkList::get\_money\_by\_out(string& s)

{

Node\* p = first->next;

double count = 0;

while (p != nullptr)

{

if (p->data.out == s)

{

count += p->data.money;

}

p = p->next;

}

return count;

}

vector<string> LinkList::get\_all\_out()

{

Node\* p = first->next;

vector<string> arry;

arry.clear();

arry.push\_back(p->data.out);

p = p->next;

while (p != nullptr)

{

if (not\_in\_vector(p->data.out, arry))

{

arry.push\_back(p->data.out);

}

p = p->next;

}

return arry;

}

void LinkList::reverse\_allspend\_by\_out()

{

Node\* p = first->next;

vector<string> arry;

arry.clear();

arry.push\_back(p->data.out);

p = p->next;

while (p != nullptr)

{

if (not\_in\_vector(p->data.out, arry))

{

arry.push\_back(p->data.out);

}

p = p->next;

}

vector<project> x;

x.clear();

project temp;

for (int i = 0; i < arry.size(); i++)

{

temp.date = " ";

temp.out = arry[i];

temp.money = get\_money\_by\_out(arry[i]);

x.push\_back(temp);

}

sort(x.begin(), x.end(), compare);

for (int i = 0; i < x.size(); i++)

{

cout << "支出项目：" << x[i].out << "\t花费：" << x[i].money << endl;

}

}

（2）LinkList.cpp

#include "LinkList.h"

#include<iostream>

#include<string>

using namespace std;

int main()

{

LinkList l;

vector<string> x;

x.clear();

while (true)

{

int n;

cout << endl;

cout << "0代表退出程序" << endl;

cout << "1代表读取文件" << endl;

cout << "2代表求最小花费" << endl;

cout << "3代表求最大花费" << endl;

cout << "4代表求每一天的平均花费" << endl;

cout << "5代表找出某一天的所有花费" << endl;

cout << "6代表找出支出项目的所有花费" << endl;

cout << "7代表按照支出项花费递减的顺序输出每一项的对应总花费" << endl;

cout << "请输入：";

cin >> n;

if (n == 0)

break;

else if (n == 1)

{

string s;

cout << "请输入文件名:";

cin >> s;

l.readfile(s);

x = l.get\_all\_out();

for (int i = 0; i < x.size(); i++)

{

cout << i << "代表" << x[i] << "\t";

}

continue;

}

else if (n == 2)

{

cout << l.get\_min() << endl;

continue;

}

else if (n == 3)

{

cout << l.get\_max() << endl;

continue;

}

else if (n == 4)

{

l.get\_average();

cout << endl;

continue;

}

else if (n == 5)

{

string s1;

cout << "给出日期：";

cin >> s1;

l.get\_money\_by\_date(s1);

continue;

}

else if (n == 6)

{

int c;

cout << "给出支出项目下标:";

cin >> c;

cout << l.get\_money\_by\_out(x[c]) << endl;

continue;

}

else if (n == 7)

{

l.reverse\_allspend\_by\_out();

continue;

}

}

return 0;

}