**第1次课 课程任务的布置和讲解**

**两种学习方式**

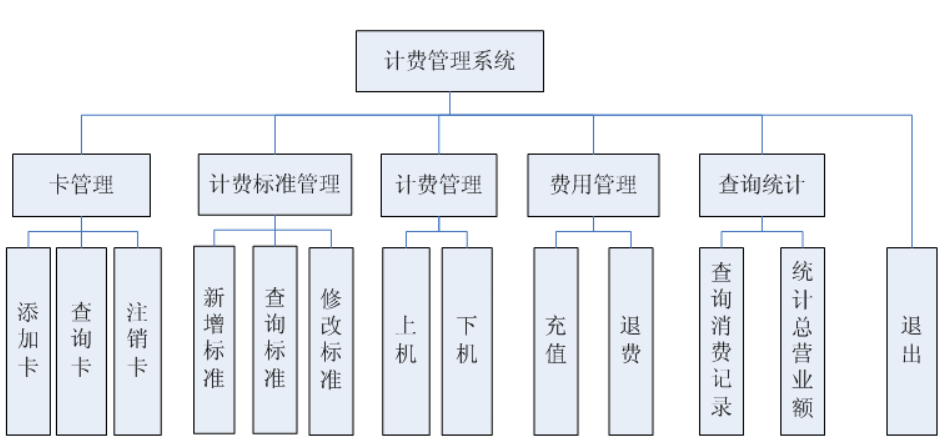
1. 通过网站视频：<http://zy1.cslab.whut.edu.cn/>
2. 通过实验指导书（推荐）

**技术方案（C/C++）**

1. 结构化编程：非类（C/C++）
2. 面向对象的编程：类（C++）

**系统需求分析**

**基本功能结构**



**卡管理：新增卡、查询卡、注销卡**

（1）新增卡：根据卡号、密码、开卡金额，新增一张新卡。卡号必须唯一。可以增加一些特色功能，如卡号必须都是数字，实际中卡号的长度都是固定的；可以对密码的复杂度做一些规定，如必须是数字和字母的组合等。

（2）查询卡：根据卡号，查询卡信息，包括卡号，卡状态，余额，累计使用，使用次数，最后使用时间。可以增加一些特色功能，如查询某段时间内开的卡。

（3）注销卡：将一张不再使用而且未注销的卡注销，注销后的卡不能在进行任何操作，记载注销的时间。（在实现上，一般是做一个删除标志，注销的 卡不能使用，但是可以查询。以前有些同学实现了恢复注销卡的功能，让注销的卡可以重新使用，这都可以，同学们可以自由发挥，实现特色功能）

**计费标准管理：新增标准、查询标准、删除标准、修改标准。**

在实现中，可以将用户进行分类，不同类别的用户使用不同的标准。注意：计费标准可以修改保存的，修改后就意味着应该按新的标准计费，这在技术实现上应该有一个文件专门存储计费标准。

**计费管理：上机、下机。**

（1）上机：根据输入卡号和密码，进行上机，并开始计费。在实现中要注意，已经上机的卡不能重复上机，卡中余额不够的也不能上机。

（2）下机：根据输入的卡号和密码，判断该卡是否正在上机。并对上机卡进行下机操作，计算消费金额。注意，消费金额要符合实际，如0.1234元就不符合实际；如按秒计费等。

**费用管理：充值、退费。**

（1）充值：可以对卡进行充值，记载每次充值的时间和金额，充值后余额累加。注意，不能充值负数，如 -10元，也可以控制充值上限。

（2）退费：不再进行消费时，可以根据卡内余额申请退费。注意：退费不能超过余额。

**查询统计：查询消费记录、统计总营业额、统计月营业额。**

（1）消费记录查询：根据卡号和时间段，查询该卡在该时间段内的消费记录。

（2）统计总营业额：根据时间段统计这段时间内的总营业额。

（3）统计月营业额：统计一年中每个月的营业额。

可以增加一些特色统计：如统计充值和退费信息，得到某个时间段的现金流等。

可以将统计信息显示在显示器上的同时，输出到文本文件中。

**权限管理：**管理使用这个系统的用户，添加用户、删除用户、设置权限等。

**登录和退出**

根据输入的账号和密码，确定用户是否有权限登录系统。

**检查实验的任务节点**

1. 界面
2. Vector（数组）方式
3. 链表方式
4. 扩展功能

第一次实验：界面的实现。

第二次实验：用数组（或Vector）实现开卡、查询卡及注销卡的功能。

第三次实验：在第2次实验的基础上，实现卡信息的更改能永久保存。

前面3次实验内容作为一个版本保存下来。

文件 <-> 数组（Vector）<-> 系统的操作

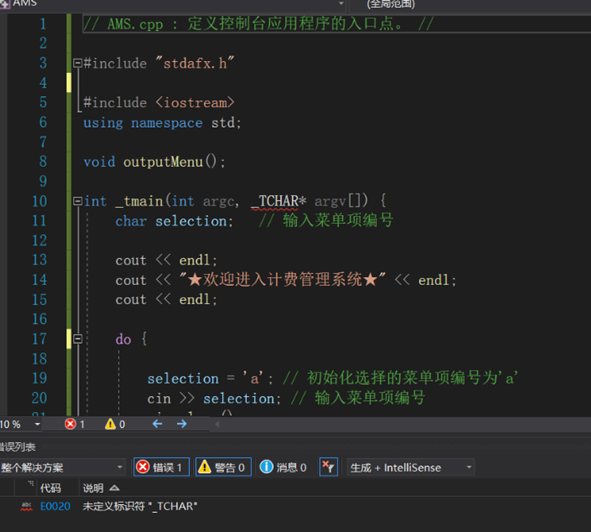
第4次：改写第3次实验内容

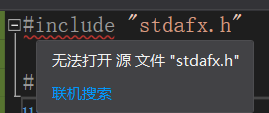
文件 <-> 链表<-> 系统的操作

第5-7次：在第4次实验基础上，完成全部功能，是基于链表的

**第2次课 1，2次实验涉及到的问题和知识点**

VS的版本问题



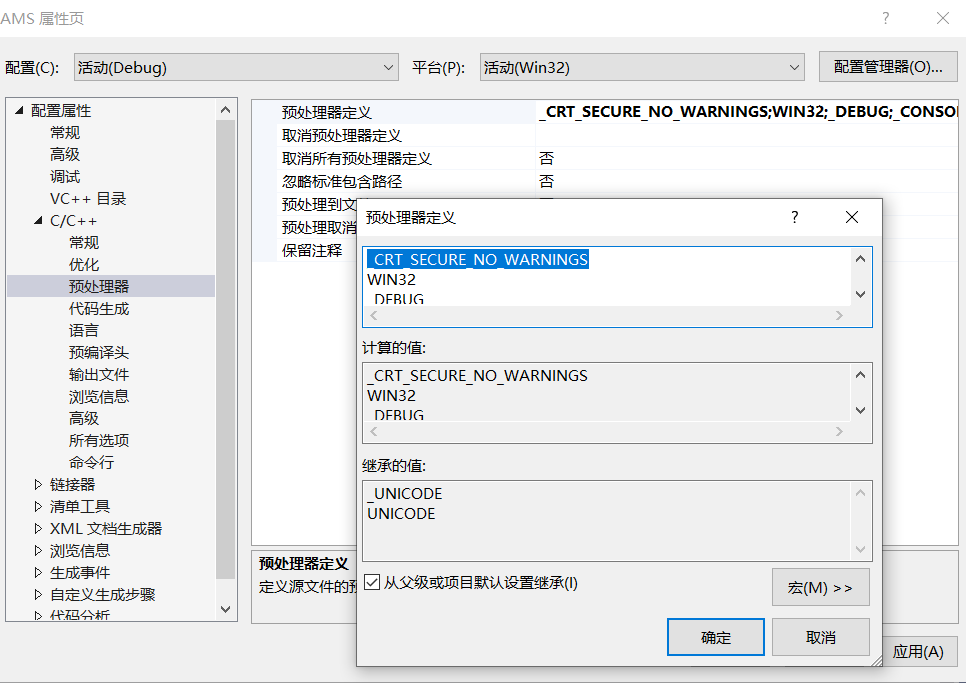


VS 2019版不需要加 #include “stdafx.h”

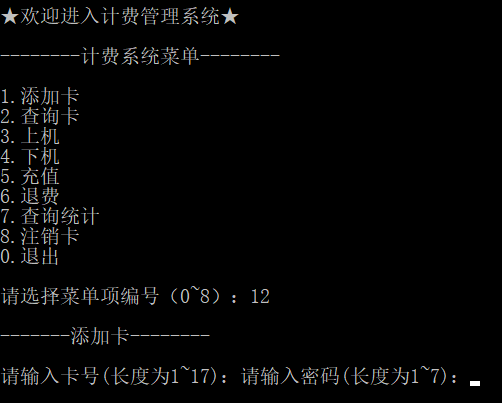
Visual Studio 2019 编写C语言程序<https://blog.csdn.net/dingyc_ee/article/details/98075146>

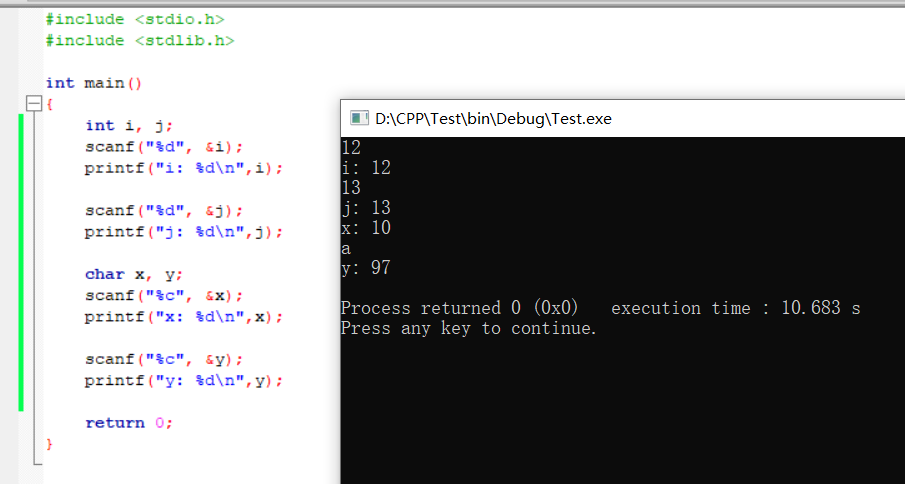
VS开发环境中，使用scanf()函数引发的警告：

<https://blog.csdn.net/nannanzhang121/article/details/96830680?depth_1-utm_source=distribute.pc_relevant.none-task&utm_source=distribute.pc_relevant.none-task>



键盘缓冲区问题的讲解





过程：键盘上输入的数据会以ASCII形式存储到键盘缓冲区，程序需要接收数据时，从键盘缓冲区读取。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 键盘 | 缓冲区 | 程序 | 说明 |
| 1 |  | 空 | scanf("%d", &i); | 缓冲区为空，等待键盘录入 |
| 2 | 12回车 | 49，50，10 |  |  |
| 3 | 取数据给i  （49，50） | 10 |  | 从缓冲区取整数，直到遇到非数字ASCII码 |
| 4 |  | 10 | scanf("%d", &j); | 从缓冲区取整数，首次遇到10或32，就丢弃掉 |
| 5 |  | 空 |  | 缓冲区为空，等待键盘录入 |
| 6 | 13回车 | 49，51，10 |  |  |
| 7 | 取数据给j  （49，51） | 10 |  | 从缓冲区取整数，直到遇到非数字ASCII码 |
| 8 | 取数据给x  （10） | 空 | scanf("%c", &x); | 缓冲区不为空，不需等待键盘录入 |
| 9 |  | 空 | scanf("%c", &y); | 缓冲区为空，等待键盘录入 |
| 10 | a回车 | 97，10 |  |  |
| 11 | 取数据给y  （97） | 10 |  | 从缓冲区取字符 |

**清除键盘缓冲区的方法**

C：fflush(stdin);

C++：

cin.clear();

cin.sync();

cin.ignore(100, '\n');

丢弃字符直到遇到'\n'（包含'\n'），如果丢弃了100个字符还没遇到字符'\n'，那也停止丢弃字符，所以100是丢弃的最大字符数。

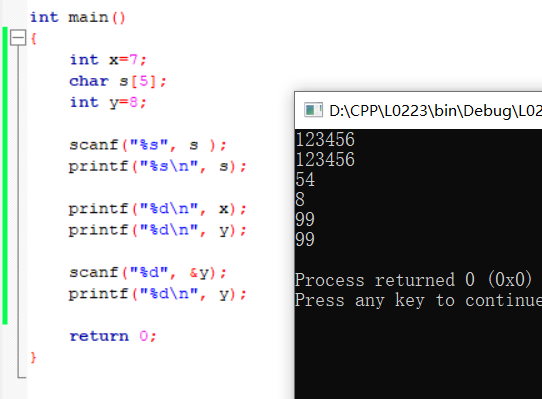
while (getchar() != '\n');

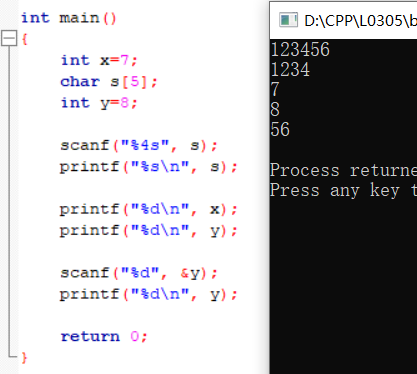
思考：

如何输入“带空格的字符串”？

如果希望卡号是由0-9之间的数字组成，如何保证？

**防止字符串越界**





**条件编译**

作用1：防止头文件的重复include

1）为什么.cpp和.h要分开？

函数只能定义一次，可以声明多次

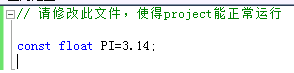
2）在一个project中，全局变量、常量一般用单独的头文件统一定义，大家都可以include，为了防止多次include，导致变量重复定义，使用条件编译

3）条件编译中，为了唯一性，一般用“文件名”做标识符。

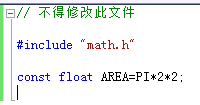
思考题：

已知project由文件math.h，circle.h和Base1.cpp组成

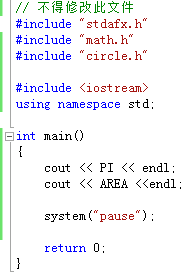
math.h代码如下



circle.h代码如下

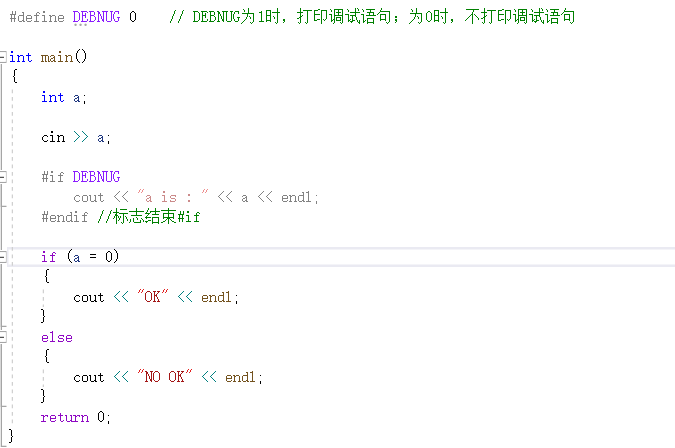


Base1.cpp代码如下



请只改动math.h，使得project能正常运行

作用2：关闭调试语句



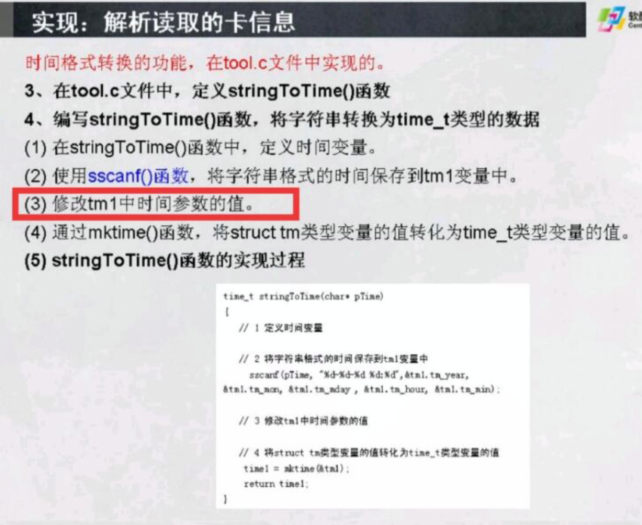
在这个例子中，把a==0 误写成 a=0，在调试中，需要先输出a观察a的值，

cout << "a is : " << a << endl;

但是实际要求中不能有输出a的值，正常我们是调试完后，需要删除这条语句。如果程序很复杂，可能会增加很多这样的语句，这样后续的删除有可能会漏掉一些语句或者错删语句，为了减少这样的错误，可以使用条件编译，它就像一个开关，通过修改DEBUG值为1或0，就可以控制这些语句是否执行。

|  |  |
| --- | --- |
| #define DEBNUG 1 | #define DEBNUG 0 |
|  |  |

**时间处理**



修改时间参数的原因：

struct tm

{

int tm\_sec; /\* Seconds: 0-59 (K&R says 0-61?) \*/

int tm\_min; /\* Minutes: 0-59 \*/

int tm\_hour; /\* Hours since midnight: 0-23 \*/

int tm\_mday; /\* Day of the month: 1-31 \*/

int tm\_mon; /\* Months \*since\* january: 0-11 \*/

int tm\_year; /\* Years since 1900 \*/

int tm\_wday; /\* Days since Sunday (0-6) \*/

int tm\_yday; /\* Days since Jan. 1: 0-365 \*/

int tm\_isdst; /\* +1 Daylight Savings Time, 0 No DST,

\* -1 don't know \*/

};

在这个结构中我们用到了年、月、日等，因为

年份为从1900年开始，所以存储时，需将你输入的年份-1900

月份为0~11，所以存储时，需将你输入的月份-1

如：你输入2020年3月，则：

tm\_year -= 1900;

tm1.tm\_mon -= 1;

如果涉及到时分秒等，也要做相关的处理。

**C++ 中的string**

注意：如果使用C++编程，由于我们需要以结构体或者对象为单元写入到文件中，而string在结构体或者类中存储的是指针，所以在结构体或者类中，不能使用string去定义name和password等字符串数据。应该使用字符数组，将string转换成字符数组。

讲解见例子演示。