《汇编语言程序设计实验》

实验三任务

实验名称：实验三 模块化程序设计

（本实验使用4+4两次课内上机学时完成，实验报告在第二次上机完成后的次日提交）

1. 实验目的和要求
2. 掌握子程序设计的方法与技巧，熟悉子程序的参数传递方法和调用原理；
3. 掌握宏指令、模块化程序的设计方法;
4. 掌握较大规模程序的合作开发与调试方法；
5. 掌握汇编语言程序与C语言程序混合编程的方法；
6. 熟悉C编译器的基本优化方法;
7. 了解C语言编译器的命名方法，主、子程序之间参数传递的机制。
8. 实验内容

任务1 宏与子程序设计 （尽量在第一次4个课内学时阶段完成，上机前实验报告应完成该任务中除实验记录与分析、总结与体会之外的内容）

进一步修改与增强实验一任务四的**网店商品信息管理程序**的功能，主要调整功能三。

**1.调整后的功能三的描述**

（1）首先显示一个功能菜单（格式自行定义。若是未登录状态，只显示菜单“1”和“6”）:

1=查询商品信息，2=修改商品信息，3=计算平均利润率，

4=计算利润率排名，5=输出全部商品信息，6=程序退出。

输入1-6的数字进入对应的功能。

（2）查询商品信息

提示用户输入要查询的商品名称。若未能在第一个网店中找到该商品，重新提示输入商品名称。若只输入回车，则回到功能三（1）。

找到该商品之后，按照：“SHOP1，商品名称，销售价，进货总数，已售数量”顺序显示该商品的信息，同时还要将“SHOP2”中该商品的信息也显示出来。显示之后回到功能三（1）。

（3）修改商品信息

提示用户输入要修改信息的商品名称（先指定网店名称）。[若把接下来的处理步骤写成子程序，则网店名称和商品名称（或其偏移地址）就是子程序的入口参数，是否找到、是否是回车或者修改成功的信息是出口参数]。若未能在指定网店中找到该商品，重新提示输入网店名称和商品名称。若只输入回车，则回到功能三（1）。

找到该商品之后，按照：进货价，销售价，进货总数的次序，逐一先显示原来的数值，然后输入新的数值（若输入有错，则重新对该项信息进行显示与修改。若直接回车，则不修改该项信息）。

如：进货价：25》24 //符号“》”仅作为分隔符，也可以选择其他分隔符号

销售价：46》5A6 //输入了非法数值，下一行重新显示和输入

销售价：46》56

进货总数：30》 //直接回车时，对这项信息不做修改

当对三项信息都处理完毕后，回到功能三（1）。

（4）计算平均利润率

首先计算SHOP1中第一个商品的利润率PR1，然后在SHOP2网店中寻找到该商品，也计算其利润率PR2。最后求出该商品的平均利润率APR=(PR1+PR2)/2，并保存到SHOP1的利润率字段中。重复上述步骤，依次将每个商品的平均利润率计算出来。回到功能三（1）。

（5）计算利润率排名

对SHOP2中的每个商品按照平均利润率的大小排名，排名信息存放到SHOP2中商品的利润率字段中。回到功能三（1）。

（6）输出全部商品信息

将SHOP1和SHOP2中的所有商品信息显示到屏幕上，包括平均利润率和排名（替代了商品原有的利润率字段）。具体的显示格式自行定义（可以分网店显示，也可以按照商品排名显示，等等，显示方式可以作为子程序的入口参数）。回到功能三（1）。

**2.其他要求**

（1）**两人一组**，一人负责包括菜单显示、程序退出在内的主程序，以及菜单中的功能（1）和（2）；另一人负责菜单中的功能（3）、（4）和（5）。各自汇编自己的模块，设计测试方法，测试通过；然后把自己的模块交给对方，各自把对方的程序整合到自己的程序中，连接生成一个程序，再进行整体调试。

**注意，**在每个模块的开始，注明编写者的名字以及同组同学的名字。整合到一起时，要注意删掉自己测试时额外加的代码，若有重复的模块（如：两个人都会使用进制转换子程序，各自模块中可能都有相同的进制转换程序），也需要去掉重复的部分。

**建议分组方法：**按照学号顺序依次两人一组，若班级人数为奇数，则最后三人一组（其中两人的分工是相同的，第三人只需要选择其中一个同学的模块与自己模块整合即可）。

（2）排名的基本要求是按照平均利润率从高到低计算名次，也可以考虑按照指定字段（比如已售数量等）排名。相同平均利润率时排名相同，下一个相邻平均利润率的名次应该是排名在前的所有商品种类“和”的下一个数值。

（3）将9号和10号DOS系统功能调用定义成宏指令并调用。功能（1）-（5）应尽量采用子程序方式实现。需要借鉴书上的进制转换程序：十进制转二进制的子程序F10T2和二进制转十进制的子程序F2T10。

**上述任务1中，值得思考或者写到操作步骤中的问题提示：**

1. 在TD中跟踪到子程序内部有几种方法？在TD中观察子程序调用和返回时堆栈的变化。
2. 注意观察FAR、NEAR类型子程序的RET指令的机器码有何不同？观察FAR类型子程序被调用时堆栈的变化情况。
3. 通过把一个模块拆成多个模块或反之，体会子程序和模块化程序设计的方法，体会模块调用关系图、子程序功能说明、输入/输出说明在程序设计中的作用。
4. 观察不同模块的可合并段合并后变量偏移地址的变化情况。观察不同段在内存里的放置次序。体会模块间段的定义及其对应的装配方法。
5. 在编程中使用不同的子程序参数传递方法来编写子程序。
6. 观察模块间的参数的传递方法，包括公共符号的定义和外部符号的引用，若符号名不一致或类型不一致会有什么现象发生？
7. 通过TD观察宏指令在执行程序中的替换和扩展，解释宏和子程序的调用有何不同。
8. 如何使菜单等显示信息显示得更漂亮一点？
9. EXTRN说明语句放在.386之前或者之后有什么区别？
10. EXTRN说明的变量的段与段寄存器的关联关系（ASSUME伪指令所表达的信息）是否能带入到本模块中？如果不能带入，是否可以通过加段前缀的方法来解决？
11. 如何利用宏功能使汇编语言的程序变得更加直观易读？

例：下面是一个利用宏功能直观化后的完整代码段程序，请写出对应的宏定义，并模仿该方式对自己编写的某段程序进行类似的改写。

StartProgram code，data，stack，start

Initial\_ds

GetStringTo BUF

DisplayStringFrom BUF

ExitToDOS

EndProgram code，start

任务2：在C语言程序中调用 汇编语言实现的函数

（主要在第二次4个课内学时阶段完成，上机前实验报告应完成该任务中除实验记录与分析、总结与体会之外的内容）

对于任务1的程序进行改造，主控程序、以及输入输出较多的某一个功能（如功能（1）、（2）、（5）中的某一个）用C语言实现，其他功能用独立的汇编语言子程序的方式实现； 在C语言程序中调用汇编语言子程序。

**要求与提示：**

(1)在不同的C语言开发环境中实现与汇编语言程序的混合编程，其操作方法有可能是不同的。请大家选择自己熟悉的C语言开发环境并查找相关的资料完成本实验（建议用BC31，其功能与操作方法相对比较简单）。

1. 在实验报告中，比较详细的给出你的开发环境及其实现方法。
2. 观察C语言编译器中对各种符号的命名规则（指编译器内部可以识别的命名规则，比如，符号名前面是否加下划线“\_”，等），主、子程序之间参数传递的机制，通过堆栈传递参数后堆栈空间回收的方法（要设计一个有多个参数需要传递的C函数）。
3. 对混合编程形成的执行程序，用调试工具观察由C语言形成的程序代码与由汇编语言形成的程序代码之间的相互关系，包括段、偏移的值，汇编指令访问C的变量时是如何翻译的，等。
4. 请尝试在C语言源程序中不合理地嵌入汇编语言的指令语句，达到破坏C语言程序的正确性的目的。比如，在连续的几条C语言语句中间加入一条修改AX寄存器（或DS等其他寄存器）的汇编指令语句，而AX的内容在此处本不该被修改，这样就可观察到破坏C语言程序正确性的效果（该项实验表明：在C语言程序中，若不考虑上下语句翻译成怎样的机器码而随意嵌入汇编指令语句时，有可能存在出错的风险）。
5. 观察C编译器的优化策略对代码的影响。通过实际观察与分析，记录本实验中汇编语言程序的效率会优于C语言程序的实例（至少给出一处的观察结果）。
6. 通过调试混合编程的程序，体会与纯粹汇编语言编写的程序的调试过程的差异。

（8）通过本次实验，希望大家明白：不同的编程语言是可以协同解决一个问题的，而且可以利用不同语言的特点来更好地解决问题；利用汇编语言的知识，能够更好地理解高级语言的内部处理原理与策略，为编写更好的C语言程序、用好C编译器提供支持。

**参考文献：**

[1]许向阳，《80X86汇编语言程序设计上机指南》“第十一章 汇编语言程序与C程序的连接”。

资料说明：

1. 第十一章 汇编语言程序与C程序的连接.doc（从教学网站的实验指导中下载）
2. C 程序调用 汇编语言函数 操作说明.doc（从教学网站的实验指导中下载）

操作步骤与多模块编程时建立一个项目（或工程，PROJECT）相同，只是选择模块时除了C语言的模块，还有汇编语言的模块（源程序或OBJ文件）。

1. BC3.1 或 TC3.0下混合编程的方法与上述“【1】【2】”中介绍的类似，但还需要注意一些区别。首先是C编译器需要外挂一个汇编程序以帮助处理汇编语言的语句，一般会使用TASM，它与MASM基本一致，语法规则只有少许差异。**教学网站中可以下载一个带DOS虚拟机的完整BC31软件，包括C编译器，TASM，TD等**。其次，命名规则有点不同，这里不需要在ASM关键词前加下划线。另外，若想C编译器对嵌入的汇编语言语句不报错，还需要在C语言程序开始之处加上说明语句#pragma inline，例如：

**#include <stdio.h>**

**#pragma inline**

**int main()**

**{ int count;**

**asm mov count,5**

**printf("count=%d",count);**

**return 0;**

**}**