****

**汇编语言与接口技术上机实验报告**

**Experimental Report of Assembly Language and Interface Technology Course of Beijing Institute of Technology**

**学 院：** 徐特立学院

**专 业：** 计算机科学与技术

**班 级：** 30141902

**姓 名：** 陈照欣

**指导教师：** 张华平

**原创性声明**

本人郑重声明：所呈交的毕业设计（论文），是本人在指导老师 的指导下独立进行研究所取得的成果。除文中已经注明引用的内容 外，本文不包含任何其他个人或集体已经发表或撰写过的研究成果。 对本文的研究做出重要贡献的个人和集体，均已在文中以明确方式 标明。

特此申明。

本人签名： 日 期： 2022 年 6 月 1 日

**关于使用授权的声明**

本人完全了解北京理工大学有关保管、使用毕业设计（论文） 的规定，其中包括：①学校有权保管、并向有关部门送交本毕业设 计（论文）的原件与复印件；②学校可以采用影印、缩印或其它复 制手段复制并保存本毕业设计（论文）；③学校可允许本毕业设计（论 文）被查阅或借阅；④学校可以学术交流为目的,复制赠送和交换本 毕业设计（论文）；⑤学校可以公布本毕业设计（论文）的全部或部 分内容。

本人签名： 日 期： 2021 年 5 月 20 日

指导老师签名： 日 期： 年 月 日

**北京理工大学汇编语言与接口技术课程实验报告**

摘要

汇编语言是用于电子计算机，微处理器等其他编程器件的低级语言，也被称为是符号语言。在汇编语言中，实际的机器指令被助记符替换，用地址符号或其他标号来代替操作数的地址，在不同的设备中对应着不同的机器语言指令集，通过汇编过程转化成机器指令。

尽管现在汇编语言已不像其他大多数的程序设计语言一样被广泛用于程序设计，在今天的实际应用中，它通常被应用在底层硬件操作和高要求的程序优化的场合。驱动程序、嵌入式操作系统和实时运行程序中都会需要汇编语言。

本报告为2021-2022年第二学期北京理工大学计算机学院汇编语言与接口技术课程的实验报告。本报告完成了大数乘法、计算器、文本比较三个实验，对程序设计的过程进行了具体的说明和展示，对编写的代码进行测试以及对实验运行结果的评估。

本报告一共分为四章，前三章依次对每个实验的具体完成过程进行了详细的分析，最后一章为对本学期汇编语言与接口技术课程的总结与收获。

**关键词 汇编语言，反汇编，课程心得**

**Experimental Report of Assembly Language and Interface Technology Course of Beijinginstitute of Technology**

Abstract

Assembly language is a low-level language used for programming devices such as computers and microprocessors, and is also called a symbolic language. In assembly language, the actual machine instructions are replaced by mnemonics, and the address symbols or other labels are used to replace the addresses of the operands. Different equipment corresponds to different machine language instruction sets, which are converted into machine instructions through the assembly process.

Although assembly language has not been widely used in programming like most other programming languages, in today's practical applications, it is usually used in low-level hardware operations and high-demand program optimization occasions. Assembly language is needed in drivers, embedded operating systems, and real-time running programs.

This report is an experimental report of the assembly language and interface technology course of the School of Computer Science and Technology, Beijing Institute of Technology in the second semester of 2021-2022. This report completes three experiments: large number multiplication, calculator, and text comparison, and specifically explains and demonstrates the process of program design, tests the written code, and evaluates the results of the experiment.

This report is divided into four chapters. The first three chapters give a detailed analysis of the specific completion process of each experiment in turn. The last chapter is a summary of assembly language and interface technology courses this semester.

Key words assembly languag， lesson experience

1. **大数乘法**
   1. **实验要求**

采用x86汇编语言实现大数相乘，要求实现两个十进制大整数的相乘（100位以上），输出乘法运算的结果。

* 1. **实验环境**

本实验进行的环境如表1-1，如果没有特殊说明，本实验后续部分的运行环境仍保持一致

表 1-1

|  |  |
| --- | --- |
| **名称** | **信息** |
| 操作系统 | Windows 10 |
| 汇编环境 | MASM 32 SDK |
| IDE | Visual Studio 2019 |
| 汇编语言 | x86 |

* 1. **实验原理**

大数相乘就是指相乘的数字比较大，本身就已经超过了基本类型的表示范围，所以这样的数据是不能直接使用指令相乘计算的。大数乘法算法有很多种，如下

·小学模拟乘法：最简单的乘法竖式手算累加形

·分治乘法：最简单的是Karatsuba乘法，一般化以后有Toom-Cook乘法；

·快速傅里叶变换FFT：（为了避免精度问题，可以改用快速数论变换FNTT）；

本实验采用的是模拟小学乘法算法，也就是最简单的乘法竖式手算的累加形。实验的原理就是，将乘数B的每一位都与A进行相乘，将所有中间结果累加，最后得到乘数的结果。

* 1. **实验步骤**

由于普通没有任何已有数据类型可以存放超过一百位数据，故需要设置字符数组数据结构，将其转为整形数组后进行模拟手算。关键数据结构为：

CharA byte 100 dup(0)

CharB byte 100 dup(0)

resultChar byte 200 dup(0)

IntA dword 100 dup(0)

IntB dword 100 dup(0)

resultInt dword 200 dup(0)

lengthA dword 0

lengthB dword 0

lengthC dword 0

整体功能分解为如下模块：

1. 输入输出大整数字符串
2. 字符数组与整形数字的转换
3. 大数乘法
   * 1. **输入输出大整数字符串**

该部分在主函数中实现，通过调用scanf和strlen等库函数获取两个字符及其长度，保存在全局变量中。调用转换函数和乘法函数后，将结果数组转成字符串保存到全局变量，使用printf输出。

* + 1. **字符数组与整形数组的转换**

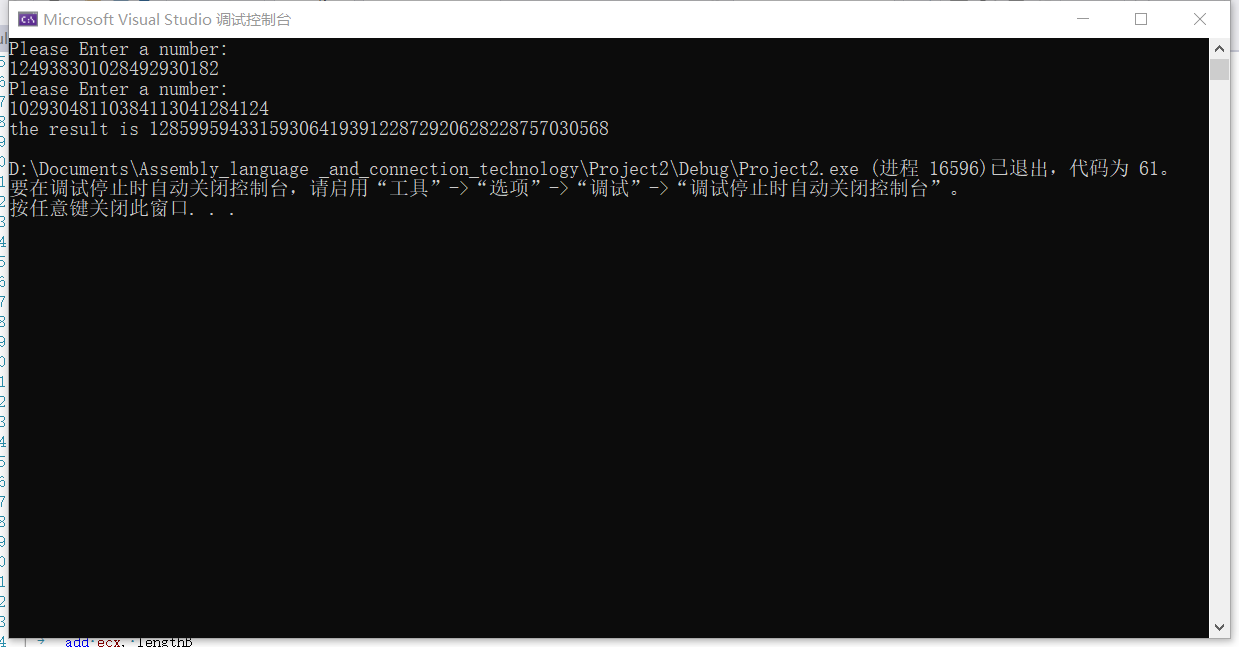
该功能由char2int\_reverse与int2char\_reverse函数组成。输入字符串、数组结构、长度，即可实现互相转换。由于计算过程是从低位到高位，因此在字符数组转整型数组时，将低位保存在前部，实现两种结构的逆序转换，方便之后的计算。

* + 1. **大数乘法**

该功能主要由big\_mul实现，主要为双循环结构。这一过程从数字的低位遍历到高位，在计算完毕后，需要对每一位进行进位处理。将除10的余数保留在当前位，商作为进位仅需传递。

在最后需要计算出大整数的长度，此时要注意处理高位为0的情形，需要在计算位数的时候减去相应长度。

* 1. **实验结果**



* 1. **实验小节**

在本次编写汇编程序的过程中，对汇编程序的结构有了更深刻的了解，加深了对课堂中学习的汇编指令理论部分的理解。在编程过程中也遇到了很多困难，但是通过上网查询，查看汇编指令的说明，解决了这些语法错误。在编写过程中尽量使用模块化的思想，加快了调试和修改错误的速度，带来了很多便利。

大数乘法的程序也曾经在c语言中实现过。这一次在汇编中再一次实现，了解到了两个语言程序的不同之处。汇编语言虽然不及C语言或是Python等语言更加流行，学习起来也更加的困难，但是汇编语言编译后运行速度更快，在现在很多程序的核心部分仍被使用。

1. **计算器**
   1. **实验要求**

结合Windows界面编程和浮点数编程，实现完善的计算器功能，支持浮点运算和三角函数等功能。

* 1. **运行环境**

表 2-1

|  |  |
| --- | --- |
| 名称 | 信息 |
| 操作系统 | Windows 10 |
| 汇编环境 | MASM 32 SDK |
| IDE | Visual Studio 2019 |
| 汇编语言 | x86 |

* 1. **实验原理**

与之前不同的是，这次需要实现Windows界面风格的程序，因此除了文件比较逻辑之外，还需要实现带有输入输出的Windows窗口。

* + 1. **Windows窗口程序**

窗口主要元素应该包括

1. 两个用于显示输入输出的编辑框
2. 多个功能按钮

如此，首先需要在主函数中实现窗口的初始化并将其显示，然后就可以进入对消息获取和处理的无限循环。

窗口初始化在\_ProcWinMain函数中实现，具体需要完成的工作为

1. 创建两个文本框
2. 创建按钮，并设置对应的响应信号
3. 设置对响应信号的操作
   * 1. **响应信号处理**
4. 如果输入数字，若此前未输入符号，则将数字填入第一个操作数；若符号不为空，则为第二个操作数。
5. 如果输入符号，若不为等号且符号位不为空，则先将之前的式子进行运算，将结果存入第一个操作数中；如果输入等号，则直接进行计算。
6. 在判断符号是否为空时，三角函数另行判断，一般不作为符号判断依据。
   * 1. **计算过程**
7. 加法

invoke StrToFloat,addr BufferA,addr tempA

invoke StrToFloat,addr BufferB,addr tempB

fld tempA

fld tempB

fadd st(0), st(1)

fst tempAns

invoke FloatToStr,tempAns,addr BufferAns

1. 减法

invoke StrToFloat,addr BufferA,addr tempA

invoke StrToFloat,addr BufferB,addr tempB

fld tempB

fld tempA

fsub st(0), st(1)

fst tempAns

invoke FloatToStr,tempAns,addr BufferAns

1. cos/sin

invoke StrToFloat,addr BufferA,addr tempA

fld tempA

fldpi

fmul st(0), st(1)

fld temp

fdiv st(1), st(0)

fstp temp

fcos/fsin

fst tempAns

invoke FloatToStr,tempAns,addr BufferAns

1. mod

invoke StrToFloat,addr BufferA,addr tempA

invoke StrToFloat,addr BufferB,addr tempB

fld tempB

fld tempA

fprem

fst tempAns

invoke FloatToStr,tempAns,addr BufferAns

* 1. **实验结果**
     1. **计算器界面**

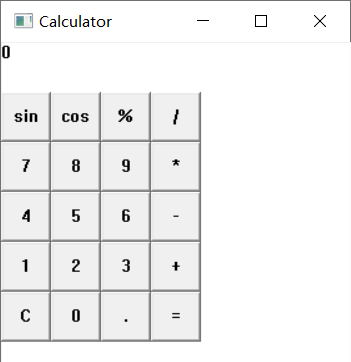
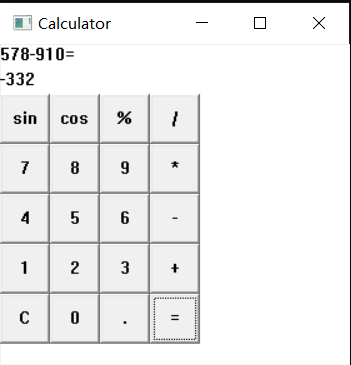
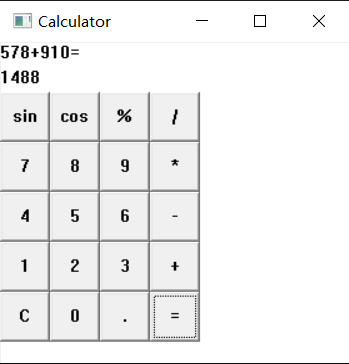
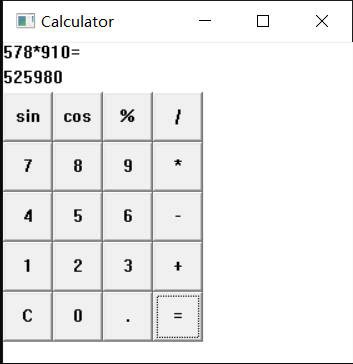
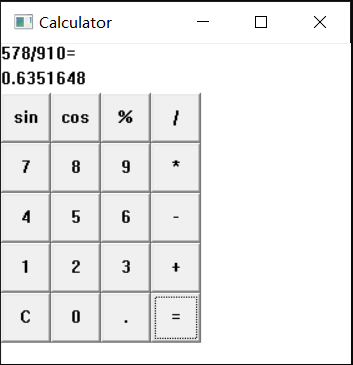


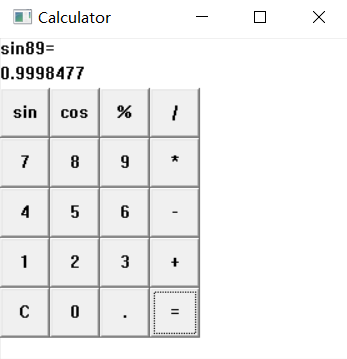
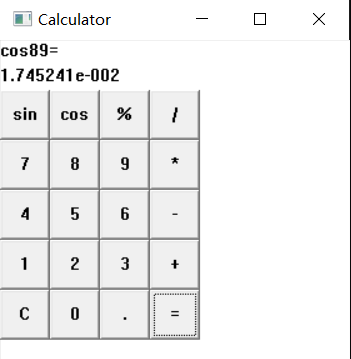
图2-1 计算器界面

* + 1. **基本的加减乘除**

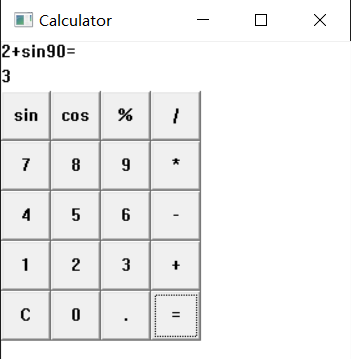


* + 1. **三角函数**

* + 1. **复合运算**



* 1. **实验小节**

这是第一次使用汇编语言进行界面编程，总的来说，在windows下进行界面编程的难点主要在以下几个方面。

* 如何进行界面开发？
* 如何处理鼠标事件？
* 如何进行数据计算？

进行界面开发当然是使用系统自带的API了。通过系统API创建窗口并增加通用控件即可实现界面编程。windows鼠标事件主要是通过消息类型来进行判断的，给每一个需要设置鼠标事件的控件设置ID，当鼠标点击了这个控件之后，在WM\_COMMAND事件里面通过ID进行判断。在数据计算中，首先需要获取完整符号和操作数，然后不同符号进行不同运算即可。

目前我们实现的计算器仅能支持双目运算，即运算的操作数最多支持两个。而且没有考虑到运算符优先级等等。如果要完全实现科学计算器功能，则要用到字符串处理，难度较大，在短期内不容易完成。

1. **文件对比**
   1. **实验要求**

Windows界面风格实现两个文本文件内容的比对。若两文件内容一样，输出相应提示；若两文件不一样，输出对应的行号。

* 1. **运行环境**

表 3-1

|  |  |
| --- | --- |
| 名称 | 信息 |
| 操作系统 | Windows 10 |
| 汇编环境 | MASM 32 SDK |
| IDE | Visual Studio 2019 |
| 汇编语言 | x86 |

* 1. **实验原理**

读取两个文本的内容，逐行的进行比较。如果有不同的地方，则在最后的输出中显示不同的行数，如果完全一致，则在最后的输出中输出提示信息。

主要应用的函数如下

表3-2

|  |  |
| --- | --- |
| 函数原型 | 函数作用 |
| readfile | 库函数，用于读取文件内容，可以设定一次读取的长度等 |
| readline | 自定义函数，指定读取一行内容 |
| MyCompareFile | 自定义函数，负责处理内容的比较 |
| strcmp | 用于比较两个字符串并根据比较结果返回整数，如果第一个串小于第二个串，则返回负数，反之则返回正数 |
| CreateFile | 库函数，给定一个路径，若路径指向的文件存在，则打开该文件；若不存在，则新建一个文件 |

* + 1. **Windows窗口程序**

窗口主要元素应该包括

1. 两个用于显示输入输出的编辑框
2. 一个确认比较按钮
3. 结果提示窗口
4. 报错窗口

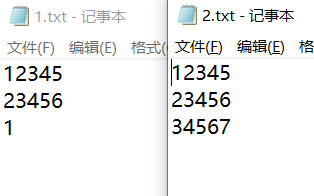
窗口初始化在\_WinMain函数中实现，具体需要完成的工作为

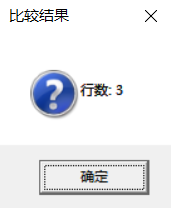
1. 创建两个文本框
2. 创建按钮，并设置对应的响应信号
3. 设置对响应信号的操作，获取两个文本框中的路径，调用MyCompareFile文件进行比较，若文本框中路径为空会报错。
   * 1. **文件比较功能**

由于需要输出不同的行，所以需要逐行处理文件。处理行数的方法即为读取换行符“\n”，当遇到换行符时，暂停读取操作。将该行内容存入缓冲区，使用strcmp进行比较，同时行数indexline加一。如读入空字节，说明文件读取完毕，停止读取操作。

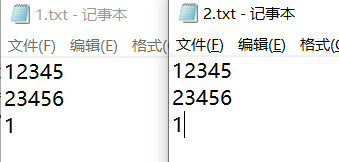
* 1. **实验结果**

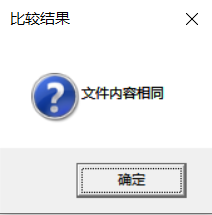
组1：





组2：





组3：

输入为空



* 1. **实验小结**

在本实验中用汇编语言实现了文本的匹配。匹配的算法比较简单原始，逐行从文件中读入内容，逐行进行匹配，需要考虑的一点是要忽略换行符对文档比较时的影响。