# 《汇编语言》实验报告07

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 班级 | 2022秋 | 实验日期 | 2022.12.30 | 实验成绩 |  |
| 姓名 | 黄勖 | 学号 | 22920212204392 | | |
| 实验名称 | 汇编语言第七次实验 | | | | |
| 实  验  目  的  、  要  求 | 1）汇编指令综合应用  2）熟悉32位Intel汇编指令 | | | | |
| 实  验  内  容  、  步  骤  及  结  果 | 1. 请使用32位的Intel x86的指令，打印计算10000以内的水仙数的程序（正确的“水仙花数”其实是一个3位数，在这里我们不限制它的位数。“水仙花数”是指一个n位数，其各个位的数字的n次方的和为它本身，例：153是一个3位数，153=1^3+5^3+3^3,则153即为一个水仙花数。）；并在32位的Intel x86汇编语言环境下运行通过。需要注意的点是，在32位系统下，仍可以使用16位寄存器，但是地址变成32位了。存地址一般就需要用32位寄存器。   **分析：10000以内的水仙数通过查找相关资料的值，不存在1、2、5位的答案，3位的被称为水仙花数，4位的被称为四叶玫瑰数，3位的数需要每一位的立方和为本身，4位的需要每一位的4次方和为本身，通过这个思路来枚举数字进行计算。**  **编写过程：**   1. **实验环境设置**   解压实验发的编译器压缩包，运行install.exe文件，安装在C盘。    安装结束后，在C盘目录下会产生一个masm32的文件夹    正确配置系统环境变量即可在其他目录进行编译连接操作。   1. **开始编写代码，先声明汇编与链接库** 2. **声明数据段**      1. **然后是代码段**     这是主函数，它调用了求水仙花数函数和四叶玫瑰数的函数   1. **接下来是水仙花数函数**          1. **判断水仙花数的函数**        1. **接下来是将数字转换为字符串的函数**      1. **编写四叶玫瑰数函数**        1. **判断四叶玫瑰数的函数**        1. **将数字转换为字符串**      * **运行结果：**     经测试，可以正确输出原有的答案。 | | | | |
| 总  结 | 这一次实验我对汇编语言指令有了更深的理解，并且这一次实验的实践操作颇丰，在练习编码的过程中我加深了32位指令的编写的操作的熟练度，我对每一个指令的用途和用法有了更深的认识；通过一步步地解决问题，我的实践能力提高了，这让我受益匪浅；具体遇到问题的解决方案我在下文做了更详细的总结，在此就不多赘述；在未来我还要探索汇编语言的更多应用方面，寻找更多问题，并在发现问题的过程中继续提高我对汇编语言的掌握能力，这是一次颇有意义的实验！  **算法及其实现方式总结：**  在寻找水仙花数的过程中，汇编语言的基址变址寻址方式和相对基址变址寻址方式起到了重要的作用。  基址变址（base-index）操作数把两个寄存器的值相加，得到一个偏移地址。两个寄存器分别称为基址寄存器（base）和变址寄存器（index）。格式为[base + index]，例如mov eax, [ebx + esi]。在例子中，ebx是基址寄存器，esi是变址寄存器。基址寄存器和变址寄存器可以使用任意的32位通用寄存器。  相对基址变址（based-indexed with displacement）操作数把偏移、基址、变址以及可选的比例因子组合起来，产生一个偏移地址。常见的两种格式为：[base + index + displacement]和displacement[base + index]，例子如下：  table dword 10h, 20h, 30h, 40h  row\_size = ($ - table)  dword 50h, 60h, 70h, 80h  dword 90h, 0a0h, 0b0h, 0c0h  mov ebx, row\_size  mov esi, 2  mov eax, table[ebx + esi \* 4]  table是一个二维数组，共3行4列。ebx是基址寄存器，相当于二维数组的行索引，esi是变址寄存器，相当于二维数组的列索引。  **知识点总结：**  1、交换指令xchg。xchg指令交换两个操作数的内容，但不能直接交换两个内存的内容，可用于数组内的交换。  2、循环指令loop。Loop以ecx为循环计数器进行循环，可用于遍历数组  3、判断指令cmp，条件跳转指令，无条件跳转。  Cmp指令通过修改cpu的标志位达到比较的目的，通常和je,jne,ja,jb,jg,jl等条件跳转配合使用，以及无条件跳转jmp指令。  4、间接寻址—变址操作数、基址变址操作数。  形如[eax + array1]的操作数便称为变址操作数，最常用于遍历数据。  形如[ebx+esi]的操作数称为基址变址操作数，可用于访问二维数组  5、dup操作符用于声明大型数组，包括需要初始化的数组和不要初始化的数组。  6、寄存器esi和edi是常用的变址寄存器。他们类似于指针，对字符操作非常有用。  7、在处理dword类型的数组时，偏移量是以4为单位，而不是1，因为一个带符号双字节占用4个字节的内存空间  8、操作符lengthof用于计算数组的元素个数，操作符sizeof用于计算数组占用字节空间 | | | | |