

**微机原理与汇编语言课程设计**

学院名称： 计算机工程学院

专 业： 计算机科学与技术

班 级： 18计3z

学 号： 2018141524

姓 名： 刘展鹏

指导教师： 傅中君

2020年6月18日

目录

[一、 摘要 2](#_Toc437021299)

[二、 课程设计目的 2](#_Toc622899553)

[三、 设计任务及要求 2](#_Toc90008146)

[四、 程序总体结构及基本算法 2](#_Toc938422334)

[五、 概要设计 3](#_Toc944263970)

[1 各模块设计思想 3](#_Toc340392460)

[1.1 菜单 3](#_Toc79639612)

[1.2 数据的录入 3](#_Toc620646803)

[1.3 数据的存储 3](#_Toc882744542)

[1.4 数据的转换 3](#_Toc1470483918)

[1.5 数据的排序 3](#_Toc1181400150)

[1.6 数据的查询 3](#_Toc158520888)

[1.7 数据的输出 3](#_Toc1380842336)

[2 程序各模块流程图 3](#_Toc2108851670)

[六、 程序详细设计 5](#_Toc1399907602)

[1. 数据的定义 5](#_Toc416230282)

[2.数据的输入 6](#_Toc1228111295)

[3.数据的存储 7](#_Toc1401203748)

[4.数据的排序 8](#_Toc725719634)

[5.数据的输出 9](#_Toc1610257325)

[6.数据的查询 10](#_Toc1005941781)

[七、 测试数据及bug分析 13](#_Toc1872244083)

[八、 收获与体会 14](#_Toc1875907137)

[九、 参考文献 14](#_Toc1163829952)

[十、 附录(源代码) 15](#_Toc1208946388)

1. 摘要

**摘要**：用汇编语言作为自己程序设计的主流语言是不现实的，当代码量达到一定程度后，代码的维护会变困难，甚至不可能实现。但我们仍要学习汇编语言，但学习汇编的目的不是用它去进行实实在在的程序设计，而是充分获得底层的编程体验和深刻理解机器运行程序的机理。另外，对于一个人来说，不能没有常识，尽管常识不能直接挣钱吃饭，但它影响谈吐，影响你的判断力和决断力，决定着你接受新事物和新知识的程度。相应的，汇编语言就是计算机语言里面的常识和基础。

1. 课程设计目的

1.了解汇编语言程序设计的过程；熟悉DEBUG的基本调试命令；掌握应用DEBUG调试汇编语言程序的基本方法，熟悉顺序、分支、循环、子程序等设计的方法。

2.熟练掌握子程序的参数传递的方式、子程序的嵌套及其设计方法和技巧

3.培养学生创新精神和独立运用所学基本知识和技能分析问题和解决问题的能力。

4.进行汇编语言程序设计应用的综合性训练；掌握基于DOS/BIOS功能调用的标准控制台输入/输出程序的设计方法；灵活应用调试手段，实现标准控制台输入/ 输出程序的调试。

1. 设计任务及要求

约瑟夫问题的重现：

约瑟夫问题是个有名的问题：N个人围成一圈，从第一个开始报数，第M个将被杀掉，最后剩下一个，其余人都将被杀掉。例如N=6，M=5，被杀掉的顺序是：5，4，6，2，3。

我们本次的任务就是利用汇编代码重新设计并解决约瑟夫问题。

本次实验中，由于汇编语言的限制，所以我们一开始就直接初始化最大报数值以及总人数，需要注意的是，初始化结束之后，我们需要将对应数据段中相应的内容做适当的修改。

1. 程序总体结构及基本算法

程序的总体来将结构不是特别的复杂，算法的部分我们利用取余成功的实现了约瑟夫环的不停的来回遍历。

1. 概要设计

1 各模块设计思想

1.1 数据的定义

我们数据的部分定义，主要是根据我们自己一开始初始化的人数和最大报数，我们设计一个数组，用来存放每一个人的编号，每一个编号是一个字节，所以说其可以表示的最大的编号值为255。

如果我们定义了6个人，那么就需要定义一个长度为6的数组，并分别初始化，对应的数组中的元素还需要我们去自定义其初始值：

比如说，如果我们定义了6个人，我们就需要初始化其为：01H、02H、03H、04H、05H、06H。

同时我们需要定义一个同样大小的数组来存放我们的结果，以便后面我们依次输出约瑟夫依次出局的数字.

就如下面所示:

M1 EQU 2 ; Upper limit of the number,m   
N EQU 6 ; the number of people,n   
DATA SEGMENT   
 SOURCE DB 01H,02H,03H,04H,05H,06H ;存储每一个人的编号  
 RESULT DB N DUP(0) ;用来存储前后出局的编号  
 M DB 00H ; 用来当作计数器，最后输出结果的时候使用  
 M2 DB 00H ; 存储每一个人出局的时候应当存放在结果的哪一个位置  
DATA ENDS

1.2 数据的处理，即将出局的人的编号依次存放在对应的数组中

该部分主要问题就是如何进行约瑟夫环的重复循环。我们利用汇编中的减法实现，然后其余部分的话，我们是先使用高级语言先实现了一遍，实现代码如下：

k=0;

while(n!=1) {

s=m%n;

if(k!=0)s=(s+k-2)%n+1;

printf(" %d ",a[s-1]);

for(i=s-1;i<n-1;i++)

a[i]=a[i+1];

k=s;

n--;

}

我们翻译成汇编代码之后应该是：

MOV DS,AX   
 MOV ES,AX   
   
 MOV DH,0 ;先将k的值，也就是0，进行初始化，并将其存放在DH寄存器  
 MOV DL,N ;将总人数N的数值存放到DL寄存器中  
 CMP DL,1 ;先进行判断，判断N是否等于1，如果等于的话，就直接执行LOP2  
 JZ LOP2   
LOP1:   
 MOV AL,M1 ;将m的值存放在AL寄存器中  
 CMP AL,DL ;利用取余，汇编中的取余，就是进行减法，我们先进行判断

;如果m的值小于n的话，就直接进入下一步，否则的话，就进行减法  
 JB LOP4  
LOP3:   
 SUB AL,DL   
 CMP AL,DL   
 JNB LOP3   
 ;该种情况就是当m的值小于n的话  
LOP4:   
 CMP DH,0 ; if (k != 0)   
 JE LOP5   
 ADD AL,DH ; s = (s + k - 2) % n + 1   
 SUB AL,2   
 CMP AL,DL   
 JB LOP6   
ALOP:   
 SUB AL,DL   
 CMP AL,DL   
 JNB ALOP   
LOP6:   
 ADD AL,1   
LOP5:   
 MOV AH,0 ;   
 MOV CL,AL ;暂存s的值在CL寄存器中  
 DEC AL ; s - 1   
 MOV BX,AX   
 MOV AL,SOURCE[BX] ; printf(" %d ",a[s-1])   
 MOV BL,M2 ;M2存储的是我们将编号存放在结果中的位置  
 MOV BH,0   
 MOV RESULT[BX],AL ;将人的对应的编号依次放在结果中的相应位置  
 INC BL   
 MOV M2,BL ;将结果的下标加一，以备下次存放结果的时候  
 MOV AL,CL   
 SUB AL,1 ; i = s - 1   
 MOV CH,AL ; CH ---> i   
 ; for(i=s-1;i<n-1;i++)a[i]=a[i+1]   
 MOV AL,DL ; AL ---> n   
 SUB AL,1 ; AL ---> n-1   
 CMP CH, AL ; 比较I 和n-1的值  
 JNB LOP7 ; not less than   
LOP8: ; enter the loop   
 MOV AL,CH   
 MOV AH,0   
 MOV BX,AX   
 MOV AL,SOURCE[BX+1]   
 MOV SOURCE[BX],AL   
 INC CH   
 MOV AL,DL   
 SUB AL,1   
 CMP CH, AL ; i < n-1   
 JB LOP8   
LOP7: ; 退出循环  
 MOV DH,CL ; k=s   
 DEC DL ; n--   
 CMP DL,1 ; if DL == 1 ,退出循环  
 JNE LOP1   
LOP2:   
 MOV BX,0   
 MOV AL,SOURCE[BX] ; printf(" %d ",a[0])   
 MOV BX,N-1 ; 直接将最后一个值放在结果中  
 MOV RESULT[BX],AL

1.3 输出结果结构

该部分结构主要涉及到输出两位数的情况，因为我们本次程序的设计是最多可以255个人同时进行，那么两位数字的话，我们就得另作考虑并进行处理。其中具体的实现如下:

取出高位数字：  
 MOV AL,[DI]   
 AND AL,0F0H ;取出高位数字，即如果是两位数的高位  
 MOV CL,4 ;然后右移，得到高位数字  
 SHR AL,CL   
 CMP AL,9   
 JA X1 ;如果高位数字大于9的话，说明其值为10、11、12、13、14、15

;此时我们需要另外进行处理

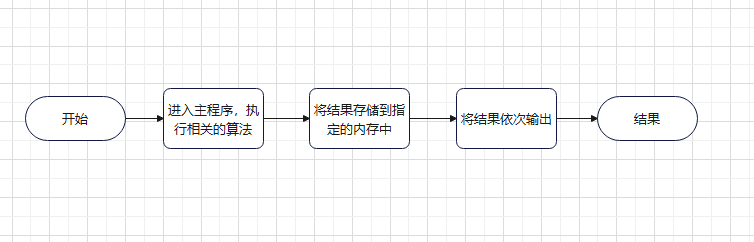
ADD AL,30H ; 如果小于9的话，我们将其转化为对应的ASCII码，

;然后调用中断，将其输出即可  
 MOV BL,AL   
 MOV AH,02H   
 MOV DL,BL   
 INT 21H ;输出高位数字  
 JMP X2   
取出低位数字：

MOV AL,[DI]   
 AND AL,0FH ;输出数字的低位数字  
 CMP AL,9   
 JA X3   
 ADD AL,30H

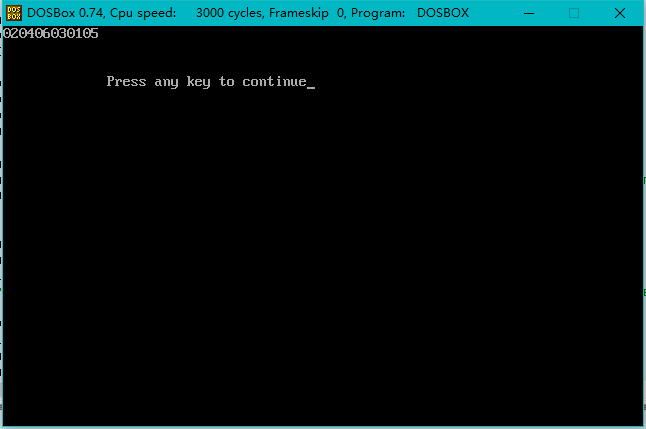
MOV BL,AL   
 MOV AH,02H   
 MOV DL,BL   
 INT 21H   
 JMP X6   
我们怎么处理大于9的数字呢，本次程序的设计是将其转化为对应的十六进制数字。

2 程序主要流程图



1. 测试数据及bug分析

本次程序中预设的值为：6和2，即我们设置总人数为6，最大报数为2，所以的话，输出序号的顺序为下面图示：



1. 收获与体会

学习计算机知识，学习的是计算机整套的系统知识,当然这其中包括的内容就多了:数据结构,操作系统，微机原理，编译原理，各种高级语言，甚至更高层次的应用,如此众多的知识对于一个初次接触计算机的学习者来说深奥,茫然.这就需要一个好的教学体系与之对应，让学生能更好的理解计算机知识.而这套教学体系的开启项就应该是汇编语言原因是由汇编语言本身的特性决定的:它充分接触硬件，理解计算机运行原理，可以让操作者感知计算机的运行过程和原理，从而能够对计算机硬件和应用程序之间的联系和交互形成一个清晰的认识.这也是最能够锻炼编程者编程思维逻辑的,只有这样，学习者才能形成一个软.硬兼备的编程知识体系，这是任何高级语言都无法给予的!相对于繁复的高级语言，汇编语言指令集合更简约，指令操作更直接,从汇编开始学习更符合循序渐进的学习原理!最根本的是让学习者知道计算机到底是什么东西，抹去计算机学习者对计算机神秘,深奥的感觉，同时增加学习者的信心。

1. 附录(源代码)
2. M1 EQU 2 ; Upper limit of the number,m   
   N EQU 6 ; the number of people,n   
   DATA SEGMENT   
    SOURCE DB 01H,02H,03H,04H,05H,06H ; number of the corresponding person   
    RESULT DB N DUP(0) ; A Boolean array that stores the current state of each person.   
    ; The initial value is 0,what we have here is the result   
    M DB 00H ; Control the times of outputs   
    M2 DB 00H ; to store the index of the results   
   DATA ENDS   
      
   SSEG SEGMENT PARA STACK 'STACK'   
    DB 256 DUP(?)   
   SSEG ENDS   
      
   CODE SEGMENT   
    ASSUME CS:CODE,DS:DATA,SS:SSEG,ES:DATA   
   START:   
    PUSH DS   
    XOR AX,AX   
    PUSH AX ; This corresponds to retf at the end   
    MOV AX,DATA   
    MOV DS,AX   
    MOV ES,AX   
      
    MOV DH,0 ; store the value of k into DH   
    MOV DL,N ; store the value of N into DL   
    CMP DL,1 ; compare DL(N) to 1,if it's equal to 1,break out of the loop   
    JZ LOP2   
   LOP1:   
    MOV AL,M1 ; store the value of m into AL   
    CMP AL,DL ; the mod operation in assembly first determines the size of m and n ,if m < n ,then the end value is m   
    JB LOP4 ; Otherwise, you just use substraction to get the remainder   
      
   LOP3:   
    SUB AL,DL   
    CMP AL,DL   
    JNB LOP3   
    ; LOP4, this is the case where m is less than n, m is the result and AL is storing the result   
   LOP4:   
    CMP DH,0 ; if (k != 0)   
    JE LOP5   
    ADD AL,DH ; s = (s + k - 2) % n + 1   
    SUB AL,2   
    CMP AL,DL   
    JB LOP6   
   ALOP:   
    SUB AL,DL   
    CMP AL,DL   
    JNB ALOP   
   LOP6:   
    ADD AL,1   
   LOP5:   
    MOV AH,0 ; store the value of AL(s) into AL   
    MOV CL,AL ; store the value of AL(s) into CL   
    DEC AL ; s - 1   
    MOV BX,AX   
    MOV AL,SOURCE[BX] ; printf(" %d ",a[s-1])   
    MOV BL,M2 ; int the begining ,the value is 0   
    MOV BH,0   
    MOV RESULT[BX],AL ; Store the number of the first person to die   
    INC BL   
    MOV M2,BL ; add the subscript of M2 to one   
    MOV AL,CL   
    SUB AL,1 ; i = s - 1   
    MOV CH,AL ; CH ---> i   
    ; for(i=s-1;i<n-1;i++)a[i]=a[i+1]   
    MOV AL,DL ; AL ---> n   
    SUB AL,1 ; AL ---> n-1   
    CMP CH, AL ; compare the sizes of i and n - 1,and exit the loop if i is less than n - 1   
    JNB LOP7 ; not less than   
   LOP8: ; enter the loop   
    MOV AL,CH   
    MOV AH,0   
    MOV BX,AX   
    MOV AL,SOURCE[BX+1]   
    MOV SOURCE[BX],AL   
    INC CH   
    MOV AL,DL   
    SUB AL,1   
    CMP CH, AL ; i < n-1   
    JB LOP8   
   LOP7: ; exit the loop   
    MOV DH,CL ; k=s   
    DEC DL ; n--   
    CMP DL,1 ; if DL == 1 ,exit the loop   
    JNE LOP1   
   LOP2:   
    MOV BX,0   
    MOV AL,SOURCE[BX] ; printf(" %d ",a[0])   
    MOV BX,N-1 ; The last person's subscript   
    MOV RESULT[BX],AL   
    LEA DI,RESULT   
    MOV M,N ; M ---> N ,the number of people   
      
   LOP0: ; The following sections output, in turn, what is stored in result   
    MOV AL,[DI]   
    AND AL,0F0H   
    MOV CL,4   
    SHR AL,CL ; Remove the high number   
    CMP AL,9   
    JA X1 ; if AL > 9   
    ADD AL,30H ; AL <= 9,The high digit of the output number   
    MOV BL,AL ; ^   
    MOV AH,02H ; |   
    MOV DL,BL ; |   
    INT 21H ; Print the number of the first person to die   
    JMP X2   
      
    ;   
   X1: ; The high number is greater than 9,即 10，11，12，13，14，15   
    CMP AL,61H ;   
    JA X5   
    ADD AL,40H ; ASCII values of letters corresponding to hexadecimal digits   
    SUB AL,09H   
    MOV BL,AL   
    MOV AH,02H   
    MOV DL,BL   
    INT 21H   
    JMP X2   
   X5: ; > 61   
    ADD AL,20H   
    MOV BL,AL   
    MOV AH,02H   
    MOV DL,BL   
    INT 21H   
   X2: ; Take out the lower four digits of the number   
    MOV AL,[DI]   
    AND AL,0FH   
    CMP AL,9   
    JA X3   
    ADD AL,30H ; Print the number of the first person to die   
    MOV BL,AL   
    MOV AH,02H   
    MOV DL,BL   
    INT 21H   
    JMP X6   
   X3:   
    CMP AL,61H   
    JA X4   
    ADD AL,40H   
    SUB AL,09H   
    MOV BL,AL   
    MOV AH,02H   
    MOV DL,BL   
    INT 21H   
    JMP X6   
   X4:   
    ADD AL,20H   
    MOV BL,AL   
    MOV AH,02H   
    MOV DL,BL   
    INT 21H   
   X6:   
    INC DI   
    DEC M   
    JNZ LOP0   
    RETF   
   CODE ENDS   
   END START