**实验三 子程序应用程序设计**

班级： 22计科3班 姓名：张雅瑞 学号： 2022334323029

1. **实验目的**
   1. 掌握程序设计中的子程序结构；
   2. 熟练使用过程伪指令、子程序调用和返回等汇编语言的指令编写子程序；
   3. 掌握数制转换方法；
   4. 掌握利用DOS系统功能调用进行字符输入及字符输出（显示）的方法。

**二、实验内容**

本次实验你所完成的具体内容加以说明。

1、从键盘上输入多个长度小于30的字符串，直到输入空行为止，将其中最长的一行字符串显示输出。

要求：（1）定义子过程GETS，实现从键盘输入字符串到参数数组中，读取包括空格在内的所有字符，直到遇到换行符为止，在字符数组中以‘\0’字符作为字符串末尾标记，该过程的返回值为输入的字符串的长度，返回值用寄存器AX传递。

（2）定义子过程PUTS，实现输出参数数组中的字符串，该过程的返回值为输出的字符个数，返回值用寄存器AX传递。

（3）定义主过程MAIN，调用两个子过程实现程序功能。

（4）输入一个字符通过调用21H号DOS中断的1号功能，输出一个字符通过该中断的2号功能调用。

2、编写十进制数到十六进制的转换程序。用户从键盘输入一个十进制无符号整数，然后把该数以十六进制形式（包括后缀字符H）在屏幕上显示出来。

要求：（1）定义两个子过程INPUT和OUTPUT。过程INPUT实现键盘输入一个十进制无符号整数，存入寄存器BX。过程OUTPUT实现以十六进制形式（包括后缀字符H）在屏幕上输出寄存器BX的值。

（2）定义主过程MAIN调用两个子过程实现程序功能。

**三、设计思想**

绘制程序框图，并说明原理及算法、程序及数据结构、主要符号名等。

程序一

**程序框图**

|  |
| --- |
| +-----------------------------------+  | main |  | 显示输入提示 |  | 循环: |  | 使用GETS读取字符串 |  | 如果字符串长度大于当前最长 |  | 保存为最长字符串 |  | 如果输入为空行，结束循环 |  | 调用PUTS输出最长字符串 |  | 显示结束提示并等待用户输入 |  +-----------------------------------+  |  +--------------------+---------------------+  | GETS |  | 从标准输入读取一行字符到数组 |  | 返回读取的字符数量 |  +--------------------+---------------------+  |  +--------------------+---------------------+  | PUTS |  | 输出DX指向的字符串到标准输出 |  | 紧接一个新行 |  +--------------------+---------------------+ |

**原理及算法描述：**

**主过程(main)：**

1. 初始化数据段指针，显示用户输入提示。
2. 进入循环，接收用户输入的字符串：
   * 使用GETS过程读取输入，将字符串保存到input数组中。
   * 检查输入的长度是否为0，若为0则退出循环。
   * 若输入非空且长度大于当前记录的maxLength，则进行更新：
     + 更新maxLength的值。
     + 将输入复制到longestString数组中保存最长字符串。
3. 循环结束后通过PUTS过程输出最长字符串。
4. 显示结束提示，等待用户任意按键后返回操作系统。

**GETS过程：**

读取用户输入直到回车符(13)，每读取一个字符就将其存入参数传递的数组，并维护一个计数器记录输入长度。在字符数组末尾添加NULL终止符。

**PUTS过程：**

接收一个指向字符串的指针，该字符串由NULL结束符终结。通过DOS中断21H，使用功能码02H将字符串逐字符输出到标准输出。每输出一个字符，指针递增，直到遇到NULL结束符。

**数据结构：**

* input: 一个数组，用来暂时存储每次用户输入的字符串。
* longestString: 一个数组，用以存储当前最长的字符串。
* maxLength: 一个字(word)，记录当前最长字符串的长度。
* prompt 和 input\_prompt: 字符串，显示输入提示和结束提示。

**主要符号名：**

* main: 程序的入口点和主过程。
* gets: 子过程，用于读取标准输入的字符串。
* puts: 子过程，用于输出字符串到标准输出。

程序二

**程序框图**

框图描述了实验二的主要流程，从初始化到输出结果，再到结束程序。

|  |
| --- |
| +-----------------------------------+  | main |  | 循环： |  | 调用decibin |  | 调用crlf |  | 调用binihex |  | 调用crlf |  | 无限循环回到开始 |  +-----------------------------------+  |  +--------------------+--------------------+  | decibin |  | 将键盘输入的十进制数转换为二进制数 |  | 存入bx寄存器 |  +--------------------+--------------------+  |  +--------------------+--------------------+  | binihex |  | 将bx寄存器内的二进制数转换为十六进制 |  | 输出到屏幕，并在末尾打印'H' |  +--------------------+--------------------+  |  +--------------------+--------------------+  | crlf |  | 回车换行操作 |  +--------------------+--------------------+ |

**原理及算法描述：**

**主过程(main)：**

程序开始后重复以下步骤：

1. 通过decibin子过程，将用户从键盘输入的十进制数转换为二进制数并保存在bx寄存器中。
2. 调用crlf子过程进行回车换行，为了更好地展示输出结果。
3. 通过binihex子过程，将bx寄存器中的二进制数转换为十六进制数，并在屏幕上显示转换后的结果。
4. 再次调用crlf子过程进行回车换行。
5. 重复执行上述步骤。

**decibin子过程：**

1. 初始化bx寄存器为0。
2. 循环接收用户从键盘输入的每个字符，将其从ASCII码转换为对应的数值，然后累加到bx中，实现从十进制到二进制的转换。
3. 当输入的字符不是数字时退出循环。

**binihex子过程：**

1. 初始化计数器ch为4，表示需要处理4个4位的十六进制数。
2. 循环将bx中的数循环左移4位，取出其最低4位，并转换为对应的十六进制字符，然后打印。
3. 循环完成后，输出字符'H'表示这是一个十六进制数。

**crlf子过程：**

执行一次回车换行，以便于在输出中清晰地分隔数据。

**数据结构：**

* 无额外的复杂数据结构。程序主要使用寄存器来保存和处理转换的数据。

**主要符号名：**

* main: 主程序，协调各个子过程的执行。
* decibin: 子过程，负责读取用户输入并将其从十进制转换为二进制。
* binihex: 子过程，负责将二进制数转换为十六进制并打印。
* crlf: 子过程，实现回车换行功能，提高输出的可读性。

这个程序的主要功能是以交互方式将用户输入的十进制数转换为十六进制数，并在每次转换后进行回车换行以分隔输出结果。利用了汇编语言中的基础中断调用、寄存器操作和循环控制来实现程序的核心功能。

**四、程序代码**

程序清单。

程序一

|  |
| --- |
| data segment  input db 30 dup(0)  longestString db 30 dup(0) ; 记录最长字符串  maxLength dw 0 ; 记录最长字符串的长度  prompt db "Press any key to continue...$"  input\_prompt db "Input some strings : ",13,10,"$"  data ends  code segment  assume cs:code, ds:data      main proc far  ; 把main过程的返回地址（ds:0）入栈，返回操作系统  push ds  mov ax, 0  push ax  ; 设置数组段寄存器为data段的段地址  mov ax, data  mov ds, ax  ;显示提示词"Input some strings : "  mov dx, offset input\_prompt  mov ah, 9  int 21h  input\_strings:  ; ax = GETS(input)  mov ax, offset input  ;push ax 这里的ax在之后会回退，不加也可以运行，把ret 2去掉  call gets  ;add sp, 2 ;栈顶回退ax 若ret 2 则不需要用  ; if (ax == 0) break 输入为空行（ax=0）时退出  cmp ax, 0  je input\_strings\_exit  ; if (ax > maxLength)  cmp ax, maxLength  jng input\_strings ;不是最大的字符串长度，继续输入  ; maxLength = ax  mov maxLength, ax  ; 从input到longestString复制字符串  ; int i=0 i:si 数组指针为0  mov si, 0  copy\_string:  ; while(input[i]!='\0')，input字符串没到末尾，则继续复制  cmp input[si], 0  je copy\_string\_exit  ; longestString[i] = input[i] 逐步复制字符串  mov al, input[si]  mov longestString[si], al  ; i++  inc si  jmp copy\_string  copy\_string\_exit:  ; longestString[i]='\0' 使得最末尾以0结尾  mov longestString[si], 0  jmp input\_strings  input\_strings\_exit:  ; PUTS(longestString) 输出最长的字符串  mov ax, offset longestString ;要输出字符串时，就用offset找出该字符串的偏移地址  push ax ;这里有待分析，ax用处  call puts  add sp, 2  ;输出结尾的提示词，并中断  mov dx, offset prompt  mov ah, 9  int 21h  mov ah,1  int 21h  ret  main endp      gets proc near  push bp  mov bp, sp  ; 保存寄存器 bx,si 因为只用ax来最后输出count的值，其他可以先保留  push bx  push si  push dx  mov bx, 0 ;bx=字符计数 bx:count  mov si, [bp+4] ;si=参数，input数组首地址  input\_char:  ;ch = getchar() ch:al  mov ah, 1  int 21h ;al=输入的字符  ;while (al != '\n')  cmp al, 13 ; '\n' ascii==13 输入回车后退出  je input\_string\_exit  ; 当前字符al写入参数数组 \*str++ = ch ch:al,str:si  mov [si], al  ;下标增加  inc si  ;参数增加  inc bx  jmp input\_char  input\_string\_exit:  ; \*str = '\0'  mov byte ptr [si], 0 ;将 0 这个字节值移动到由 si 寄存器指定的内存地址处。  ; 输出换行符  mov dl, 10  mov ah, 2  int 21h  ; return count gets用ax输出count  mov ax, bx  ; 恢复寄存器 si,bx  pop dx  pop si  pop bx  pop bp  ret  gets endp    puts proc near  push bp  mov bp, sp  ; 寄存器保存  push si  push bx  push dx  mov si, [bp+4] ;char \*str si:str [bp+4]:main压栈的实参  ; int count = 0 bx:count 计数器  mov bx, 0  output\_string:  ; while (\*str!='\0')  cmp byte ptr [si], 0  je output\_string\_exit  ; putchar(\*str++)  mov dl, [si]  mov ah, 2  int 21h  inc si  inc bx  jmp output\_string  output\_string\_exit:  ; putchar('\n')  mov dl, 13 ; 回车符=13  mov ah, 2  int 21h  mov dl, 10 ; 换行符=13  int 21h  ; return count  mov ax, bx  ; 恢复寄存器  pop dx  pop bx  pop si  pop bp  ret  puts endp      code ends  end main |

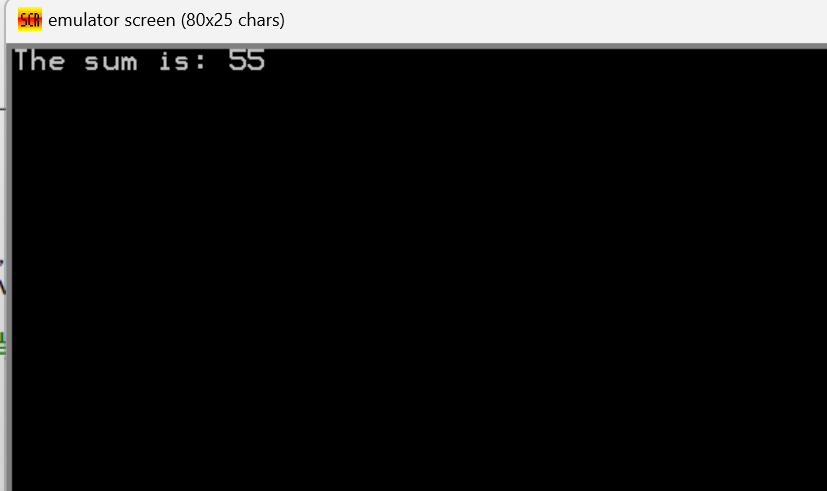
程序二

|  |
| --- |
| decihex segment  assume cs:decihex  main proc far  repeat: call decibin ;键盘输入转换为二进制数  call crlf ;回车换行  call binihex ;二进制转换为十六进制    call crlf ;回车换行  jmp repeat ;重复  main endp  ;子过程，键盘输入转换为二进制数存入bx （通过 bx 寄存器传送参数）  decibin proc near  mov bx,0 ;bx清零，初始化  newchar:  mov ah,1  int 21h ;调用21号中断1号指令，键盘输入al    sub al,30h ;ASCII转换为二进制  jl exit ;al<0退出  cmp al,9d  jg exit ;al>9退出  cbw ;al扩展到ax    xchg ax,bx  mov cx,10d  mul cx ;cx=cx\*ax  xchg ax,bx  add bx,ax  jmp newchar  exit:  ret  decibin endp  ;子程序，bx二进制数转换为十六进制数输出 （通过 bx 寄存器传送参数）  binihex proc near  mov ch,4 ;设置计数器  rotate:  mov cl,4 ;设置移位位数  rol bx,cl ;bx循环左移4位，将最高位移到低四位  mov al,bl ;bx低8位赋值给al  and al,0fh ;将al前四位清零  add al,30h ;al数据转换为ASCII码  cmp al,3ah ;判断al是否大于9  jl printit  add al,7h ;转换成相应的a~f小写字母，大写+27H  printit:  mov dl,al ;将ASCII赋值给dl  mov ah,2  int 21h ;调用21h中断2号指令，在显示器上输出dl    dec ch ;计数器减一  jnz rotate ;计数器不为零跳转到rotate再次转换    ;在这里修改功能，增加在末尾打印H的操作、  mov dl,'H'  int 21h    ret  binihex endp  ;子程序，回车换行  crlf proc near    mov dl,0dh ;回车ASCII码0dh赋值给dl  mov ah,2  int 21h ;调用int21h 2号功能，显示dl    mov dl,0ah ;换行ASCII码0ah赋值给dl  mov ah,2  int 21h ;调用int21h 2号功能，显示dl  ret  crlf endp  decihex ends  end main |

**五、结果分析**

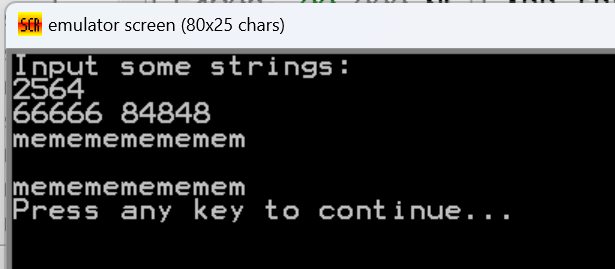
**1.程序的输出结果及对结果的分析。**

程序一



程序能接受一串十进制数字输入，并将这些输入转换为其对应的二进制，并进一步转换为十六进制数值，然后输出。输出的十六进制数应以'H'结尾以指示其为十六进制格式。由于程序无限循环，用户可以多次输入，并且每次输入后都应看到新的转换结果。

程序二



程序设计为读取一系列字符串，每次输入按回车键结束，并寻找最长的字符串。

**2.调试情况，如上机时遇到的问题及解决办法、观察到的现象及其分析。**

**实验一：**

问题一：判断字符串结束的条件可能不一致

* 观察到的现象：当输入包含回车（十进制的13）时，程序可能会预期地终止输入，但在复制到longestString时使用的是空字符作为终止标志，这可能导致数据丢失。
* 解决办法：在gets子程序一旦检测到回车后，立即写入空字符0，确保字符串是正确终止的。
* 分析：由于DOS中字符串结束标志为'$'，因此以0作为字符串的终止标志在DOS环境下可能会有问题。但在此代码中，我们假设0是正确的结束信号。

问题二：栈操作可能不恰当

* 观察到的现象：main和puts过程在调用时，压入栈的值并没有恰当的出栈操作，可能导致栈的状态不正确。
* 解决办法：确保每次对栈的操作都有对应的平衡操作。例如，如果压入参数后，确保返回地址之前在子程序结束时剔除这个参数。
* 分析：如果栈的状态没有正确维持，程序可能会遇到难以预测的行为或直接崩溃。

问题三：循环调用结构可能导致无止境的输入

* 观察到的现象：程序似乎设计为无限循环输入，最终只输出最长的字符串，但如果没有中断，它将不会结束。
* 解决办法：提供一个显式的退出条件，比如输入特殊字符串来终止循环。
* 分析：在逻辑设计中，创建有退出条件的循环是程序鲁棒性的一个重要方面。确保用户可以在完成输入后正常退出程序。

**实验二：**

问题一：输入非数字字符导致程序异常结束

* 观察到的现象：当输入字母或特殊字符时，程序并未像预期的那样退出，而是显示了一些不正确的输出或异常地结束了运行。
* 解决办法：调整decibin过程中的输入验证逻辑，确保只有数字字符才能通过并参与后续的转换计算。
* 分析：sub al, 30h操作预计只处理ASCII码表示的数字字符。如果输入的是字母或特殊字符，它们的ASCII码将在减去30h后不在有效范围内，这可导致jl和jg条件跳转指令不能正确工作，因此在转换之前需要更严格的字符验证。

问题二：转换错误的数值

* 观察到的现象：某些输入数值转换后的十六进制结果与预期不符。
* 解决办法：彻底检查并测试decibin和binihex子程序中的转换逻辑，确认算法实现是否正确处理了所有预期输入。
* 分析：二进制转十六进制的转换逻辑可能在位操作或ASCII码转换部分有误。算法需要正确处理超出单个十进制数字范围的输入，确保高位进位和低位累加得到正确结果。

问题三：用户输入的数值超出寄存器容量

* 观察到的现象：当输入较大的数值时，程序的输出结果不正确。
* 解决办法：实施输入大小的检查，限制用户输入，或者调整程序逻辑以正确处理大数值。
* 分析：由于bx寄存器是16位宽，如果用户连续输入太多的数字字符而不退出，那么mul cx 指令可能导致进位丢失，因为它只计算了16位结果。解决此问题可能需要更复杂的大数处理逻辑。

**3.对程序设计技巧的总结及分析；实验的心得体会等。**

汇编程序设计涉及多种技巧和方法，通过对上面三个实验的总结和分析，可以获得以下几点程序设计技巧和实验心得体会。

**程序设计技巧**

1. **模块化代码**：
   * 在汇编代码中，清晰的模块化结构有助于程序的维护和扩展。通过使用空行和注释来区分代码的不同部分，有助于理解代码逻辑。
2. **循环控制**：
   * 使用循环结构时，确保循环计数器的正确初始化和更新。注意循环条件的准确性，以避免无限循环或循环次数不足的问题。
3. **数据处理**：
   * 在汇编中，数据处理通常涉及寄存器操作和内存访问。确保数据的正确存储和读取，特别是涉及堆栈和内存偏移的操作。
   * 在转换数据时，确保正确的转换逻辑，如将 ASCII 字符转换为整数或十进制转换。
4. **判断逻辑**：
   * 使用条件跳转和比较操作时，确保逻辑准确。特别是在判断是否为素数等复杂操作时，确保比较和跳转条件的正确性。
5. **输出操作**：
   * 使用 DOS 中断进行字符串和字符输出时，确保调用正确的功能号，并注意输出的内容和格式。
   * 输出操作涉及堆栈和寄存器操作，确保正确处理，以避免输出错误或乱码。
6. **结束程序**：
   * 结束程序时，使用 DOS 中断 21H 的 4C 号功能，确保程序正常结束。
   * 注意释放资源，确保堆栈和寄存器的状态恢复正常。

**实验心得体会**

1. **调试技巧**：
   * 在汇编中，调试过程至关重要。通过逐步执行、观察寄存器和内存状态，可以找出程序中的问题。
   * 调试过程中，注意循环和条件跳转的逻辑，确保代码按预期运行。
2. **数据验证**：
   * 在数据输入和处理过程中，验证数据的正确性，确保输入数据符合预期。
   * 在转换和计算过程中，确保数据转换和计算结果正确。
3. **代码清晰度**：
   * 使用注释和模块化结构，有助于提高代码的可读性和维护性。
   * 保持代码简洁，避免冗余或不必要的操作。
4. **程序设计思维**：
   * 在设计程序时，确保逻辑连贯，流程清晰。在汇编中，结构化思维和逻辑分析是关键。
   * 在实现复杂逻辑时，考虑使用子程序和模块化设计，以提高代码的可维护性。

**结论**

通过对这个实验的总结和分析，可以看到程序设计技巧在汇编中具有重要意义。通过模块化代码、循环控制、数据处理、判断逻辑、输出操作等技巧，可以确保汇编程序的正确性和稳定性。在调试和设计过程中，注意观察和分析程序逻辑，确保代码按预期运行，输出正确结果。实验心得体会强调了调试技巧、数据验证、代码清晰度和程序设计思维的重要性。