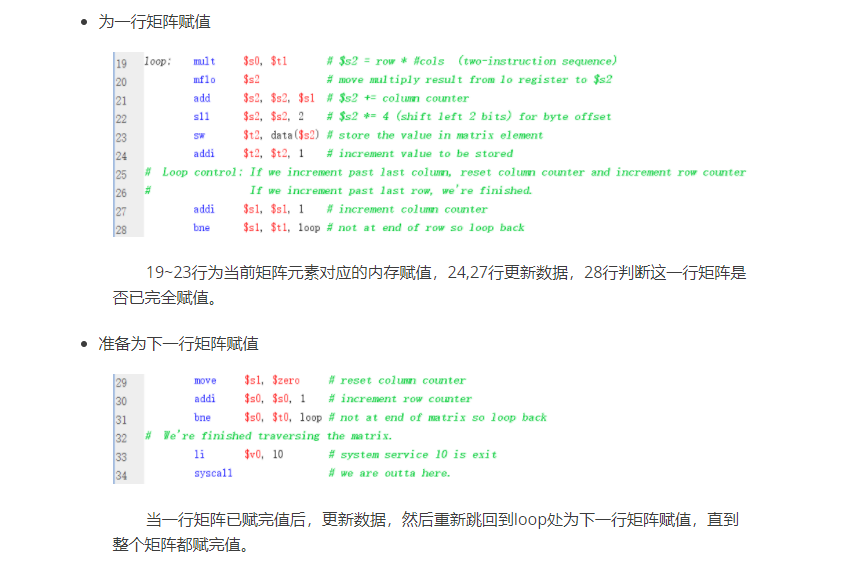
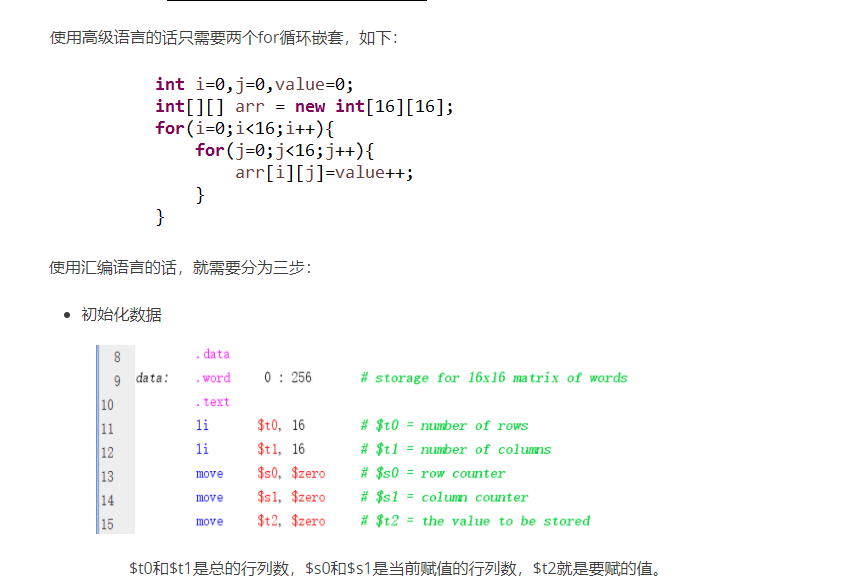
**P2**











**#include<stdio.h>**

**#int a[8][8],b[8][8],c[8][8];//不初始化为0容易出事**

**#int n;**

**#int i,j,k;**

**#int main()**

**#{**

**# scanf("%d",&n);**

**# for(i=0;i<n;i++)**

**# for(j=0;j<n;j++)**

**# scanf("%d",&a[i][j]);**

**# for(i=0;i<n;i++)**

**# for(j=0;j<n;j++)**

**# scanf("%d",&b[i][j]);**

**# for( i=0;i<n;i++)**

**# for( j=0;j<n;j++)**

**# for( k=0;k<n;k++)**

**# c[i][k]+=a[i][j]\*b[j][k];//c第i行第j列的值是a第i行和b第k列对应值乘积之和**

**# for(i=0;i<n;i++)**

**# {**

**# for(j=0;j<n;j++)**

**# printf("%d ",c[i][j]);**

**# printf("\n");**

**# }**

**# return 0;**

**#}**

**.macro scanf(%d)**

**li $v0,5**

**syscall**

**move %d,$v0**

**.end\_macro**

**.macro printf\_str(%d)**

**li $v0,4**

**move $a0,%d**

**syscall**

**.end\_macro**

**.macro printf\_int(%d)**

**li $v0,1**

**move $a0,%d**

**syscall**

**.end\_macro**

**.data**

**mat1: .word 0:80 # storage for 8\*8 matrix of words**

**mat2: .word 0:80**

**mat3: .word 0:80**

**line: .asciiz "\n"**

**space: .asciiz " "**

**.text**

**scanf($t0) #set t0 as n,as the row and column**

**move $s0,$zero # $s0 = row counter**

**move $s1,$zero # $s1 = column counter**

**move $t1,$zero # $t1 = the value to be stored**

**la $t2,space #$t2 = address of space**

**la $t3,line #$t3 = address of line**

**loop\_matrix\_1:**

**mult $s0,$t0 # $s2 = row\_now \* matrix\_col**

**mflo $s2 # move multiply result from lo register to $s2**

**add $s2,$s2,$s1 # $s2 += column counter**

**sll $s2,$s2,2 # $s2 \*= 4 (shift left 2 bit2) for byte offset**

**#得出序号，且内存乘四**

**scanf($t1) #get the value**

**sw $t1,mat1($s2) # store the value in matrix element**

**addi $s1,$s1,1 # increment column counter**

**bne $s1,$t0,loop\_matrix\_1 # not at end of row so loop back**

**move $s1,$zero #reset column counter**

**addi $s0,$s0,1 #increment row counter**

**bne $s0,$t0,loop\_matrix\_1 #not at end of matrix so loop back**

**move $s0,$zero # reset $s0 = row counter**

**move $s1,$zero # reset $s1 = column counter**

**move $t1,$zero # reset $t1 = the value to be stored**

**loop\_matrix\_2:**

**mult $s0,$t0 # $s2 = row \* cols (two-instruction sequence)**

**mflo $s2 # move multiply result from lo register to $s2**

**add $s2,$s2,$s1 # $s2 += column counter**

**sll $s2,$s2,2 # $s2 \*= 4 (shift left 2 bit2) for byte offset**

**#得出序号，且内存乘四**

**scanf($t1) #get the value**

**sw $t1,mat2($s2) # store the value in matrix element**

**addi $s1,$s1,1 # increment column counter**

**bne $s1,$t0,loop\_matrix\_2 # not at end of row so loop back**

**move $s1,$zero #reset column counter**

**addi $s0,$s0,1 #increment row counter**

**bne $s0,$t0,loop\_matrix\_2#not at end of matrix so loop back**

**move $s0,$zero #set $s0 as i**

**move $s1,$zero #set $s1 as j**

**move $s2,$zero #set $s2 as k**

**loop\_matrix\_3:**

**mult $s0,$t0**

**mflo $s3**

**add $s3,$s3,$s2**

**sll $s3,$s3,2**

**lw $s4,mat3($s3) #get c[i][k]**

**mult $s0,$t0**

**mflo $t9**

**add $t9,$t9,$s1**

**sll $t9,$t9,2**

**lw $s5,mat1($t9) #get a[i][j]**

**mult $s1,$t0**

**mflo $t9**

**add $t9,$t9,$s2**

**sll $t9,$t9,2**

**lw $s6,mat2($t9) #get b[j][k]**

**mult $s5,$s6**

**mflo $s7**

**add $s7,$s7,$s4**

**sw $s7,mat3($s3)**

**addi $s2,$s2,1**

**bne $s2,$t0,loop\_matrix\_3**

**move $s2,$zero**

**addi $s1,$s1,1 # increment column counter**

**bne $s1,$t0,loop\_matrix\_3 # not at end of row so loop back**

**move $s1,$zero #reset column counter**

**addi $s0,$s0,1 #increment row counter**

**bne $s0,$t0,loop\_matrix\_3#not at end of matrix so loop back**

**move $s0,$zero #set $s0 as i**

**move $s1,$zero #set $s1 as j**

**loop\_matrix\_4:**

**mult $s0,$t0**

**mflo $s3**

**add $s3,$s3,$s1**

**sll $s3,$s3,2**

**lw $s4,mat3($s3) #get c[i][j]**

**printf\_int($s4)**

**printf\_str($t2)**

**addi $s1,$s1,1**

**bne $s1,$t0,loop\_matrix\_4**

**printf\_str($t3)**

**move $s1,$zero**

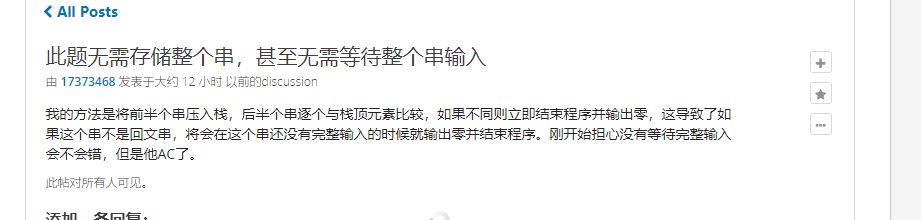
**addi $s0,$s0,1**

**bne $s0,$t0,loop\_matrix\_4**

**li $v0,10**

**syscall**





**##include<stdio.h>**

**#char a[25];**

**#int n,front,back,i,j;**

**#**

**#int main()**

**#{**

**# scanf("%d",&n);**

**# for(i=0;i<n;i++)**

**# scanf("%c",&a[i]);**

**# i--;**

**# for(j=0;j<=i;j++,i--)**

**# if(a[i]!=a[j])**

**# {**

**# printf("0");**

**# return 0;**

**# }**

**# printf("1");**

**# return 0;**

**#}**

**.macro scanf\_int(%d)**

**li $v0,5**

**syscall**

**move %d,$v0**

**.end\_macro**

**##################################################**

**.macro scanf\_char(%d)**

**li $v0,12**

**syscall**

**move %d,$v0**

**.end\_macro**

**##################################################**

**.data**

**char\_a: .word 0:25 # storage for char**

**##################################################**

**.text**

**scanf\_int($t0) #set $t0 as n**

**move $s0,$zero # set $s0 as i**

**loop\_in:**

**scanf\_char($t1) # get the value as $t1**

**sll $s1,$s0,2 # transfer the number to the address as $s1**

**sw $t1,char\_a($s1) #save the value**

**addi $s0,$s0,1 #i++**

**bne $s0,$t0,loop\_in**

**addi $s0,$s0,-1 # s0 as the back pointer**

**move $s2,$zero # s2 as the front pointer**

**loop\_check:**

**sll $t8,$s0,2 # transfer the number to address**

**sll $t9,$s2,2 # transfer the number to address**

**lw $t4,char\_a($t8)**

**lw $t5,char\_a($t9)**

**bne $t4,$t5,printf\_zero**

**addi $s2,$s2,1 # front ++**

**addi $s0,$s0,-1 # back --**

**ble $s2,$s0,loop\_check # front <= back go on**

**li $a0,1**

**li $v0,1**

**syscall**

**li $v0,10**

**syscall**

**printf\_zero:**

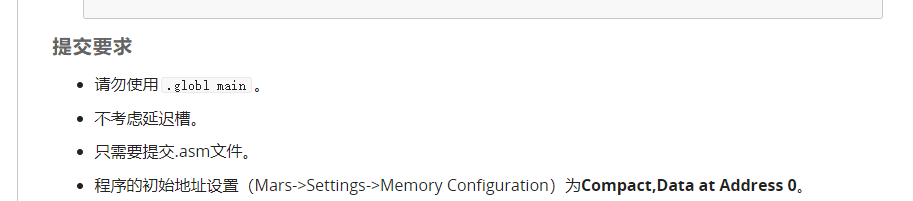
**li $a0,0**

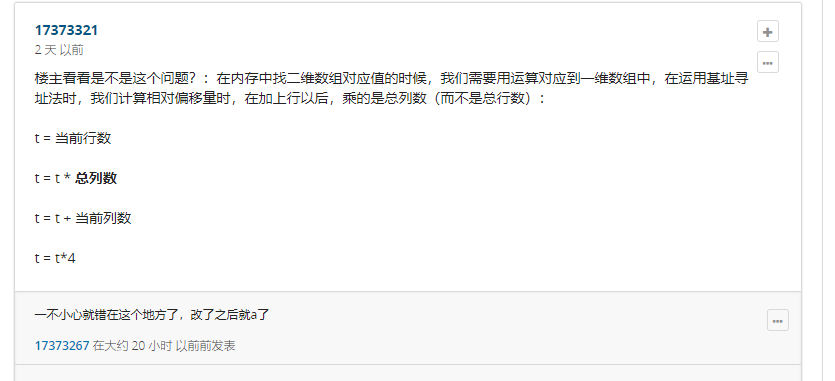
**li $v0,1**

**syscall**

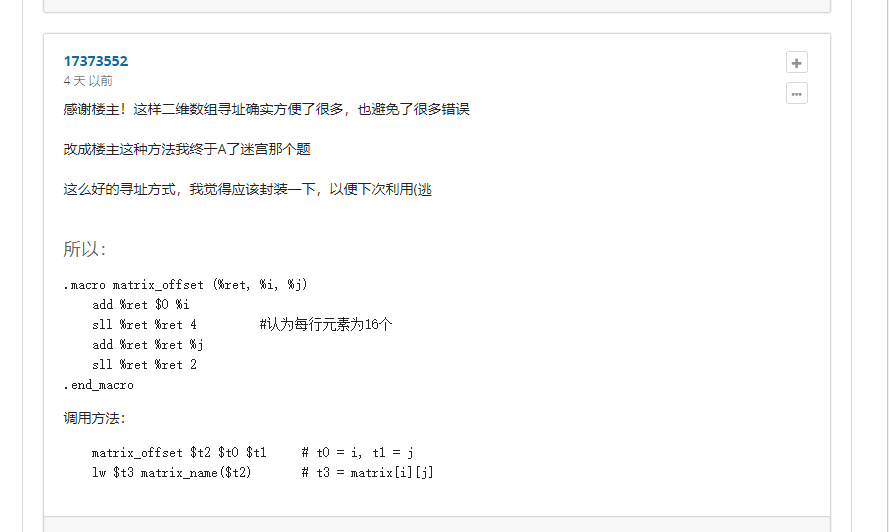
**li $v0,10**

**syscall**





**注意寻址是乘以列数！！！**

.**.macro scanf\_int(%d)**

**li $v0,5**

**syscall**

**move %d,$v0**

**.end\_macro**

**#############################################**

**.macro printf\_int(%d)**

**li $v0,1**

**move $a0,%d**

**syscall**

**.end\_macro**

**#############################################**

**.macro printf\_char(%d)**

**la $a0,%d**

**li $v0,4**

**syscall**

**.end\_macro**

**#############################################**

**.macro matrix\_offset(%num,%i,%j,%column)**

**mul %num,%i,%column**

**add %num,%num,%j**

**sll %num,%num,2**

**.end\_macro**

**#############################################**

**.data**

**mat1: .word 0:256 # storage for 16\*16 matrix**

**mat2: .word 0:256**

**mat3: .word 0:256**

**space: .asciiz " "**

**line: .asciiz "\n"**

**#############################################**

**.text**

**move $t1,$zero # set t1 as m1**

**move $t2,$zero # set t2 as n1**

**move $s1,$zero # set s1 as m2**

**move $s2,$zero # set s2 as n2**

**move $s3,$zero # set s3 as m3**

**move $s4,$zero # set s4 as n3**

**scanf\_int($t1)**

**scanf\_int($t2)**

**scanf\_int($s1)**

**scanf\_int($s2)**

**sub $s3,$t1,$s1**

**sub $s4,$t2,$s2**

**addi $s3,$s3,1**

**addi $s4,$s4,1**

**###########################################**

**move $t0,$zero # set t0 as row counter**

**move $s0,$zero # set s0 as column counter**

**loop\_in\_convolution:**

**scanf\_int($t4)**

**matrix\_offset($t3,$t0,$s0,$t2)**

**sw $t4,mat1($t3)**

**addi $s0,$s0,1 # column ++**

**bne $s0,$t2,loop\_in\_convolution**

**move $s0,$zero # reset**

**addi $t0,$t0,1 # column ++**

**bne $t0,$t1,loop\_in\_convolution**

**###########################################**

**move $t0,$zero # reset**

**move $s0,$zero # reset**

**loop\_in\_convolution\_core:**

**scanf\_int($t4)**

**matrix\_offset($t3,$t0,$s0,$s2)**

**sw $t4,mat2($t3)**

**addi $s0,$s0,1 # row ++**

**bne $s0,$s2,loop\_in\_convolution\_core**

**move $s0,$zero # reset**

**addi $t0,$t0,1 # column ++**

**bne $t0,$s1,loop\_in\_convolution\_core**

**###########################################**

**move $t0,$zero # set t0 as mat3's and mat1' i**

**move $s0,$zero # set s0 as mat3's and mat1' j**

**move $t5,$zero # set t5 as mat2's i**

**move $s5,$zero # set s5 as mat2's j**

**move $t4,$zero # reset the value t4**

**loop\_calculate:**

**matrix\_offset($t3,$t5,$s5,$s2)**

**lw $s6,mat2($t3) # get mat2' num**

**add $t8,$t0,$t5 # get mat1' num**

**add $t9,$s0,$s5**

**matrix\_offset($t3,$t8,$t9,$t2)**

**lw $t6,mat1($t3)**

**mul $t7,$s6,$t6**

**add $t4,$t4,$t7**

**addi $s5,$s5,1**

**bne $s5,$s2,loop\_calculate**

**move $s5,$zero**

**addi $t5,$t5,1**

**bne $t5,$s1,loop\_calculate**

**matrix\_offset($t3,$t0,$s0,$s4)**

**sw $t4,mat3($t3)**

**move $t4,$zero**

**move $t5,$zero**

**addi $s0,$s0,1**

**bne $s0,$s4,loop\_calculate**

**move $s0,$zero**

**addi $t0,$t0,1**

**bne $t0,$s3,loop\_calculate**

**############################################**

**move $t0,$zero # set t0 as mat3's i**

**move $s0,$zero # set s0 as mat3's j**

**loop\_out:**

**matrix\_offset($t3,$t0,$s0,$s4)**

**lw $t4,mat3($t3)**

**printf\_int($t4)**

**printf\_char(space)**

**addi $s0,$s0,1**

**bne $s0,$s4,loop\_out**

**printf\_char(line)**

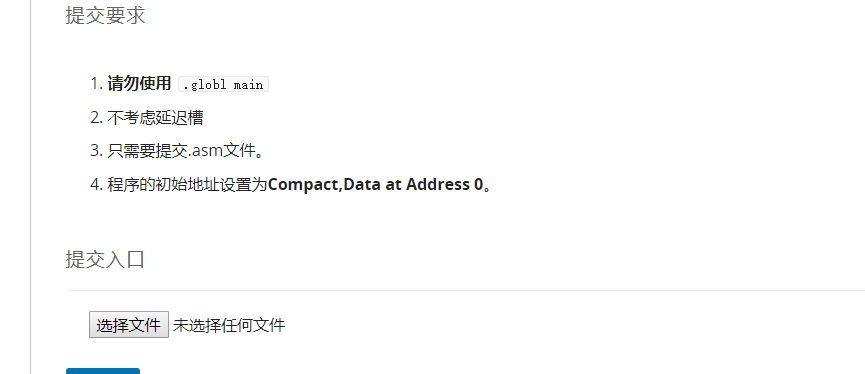
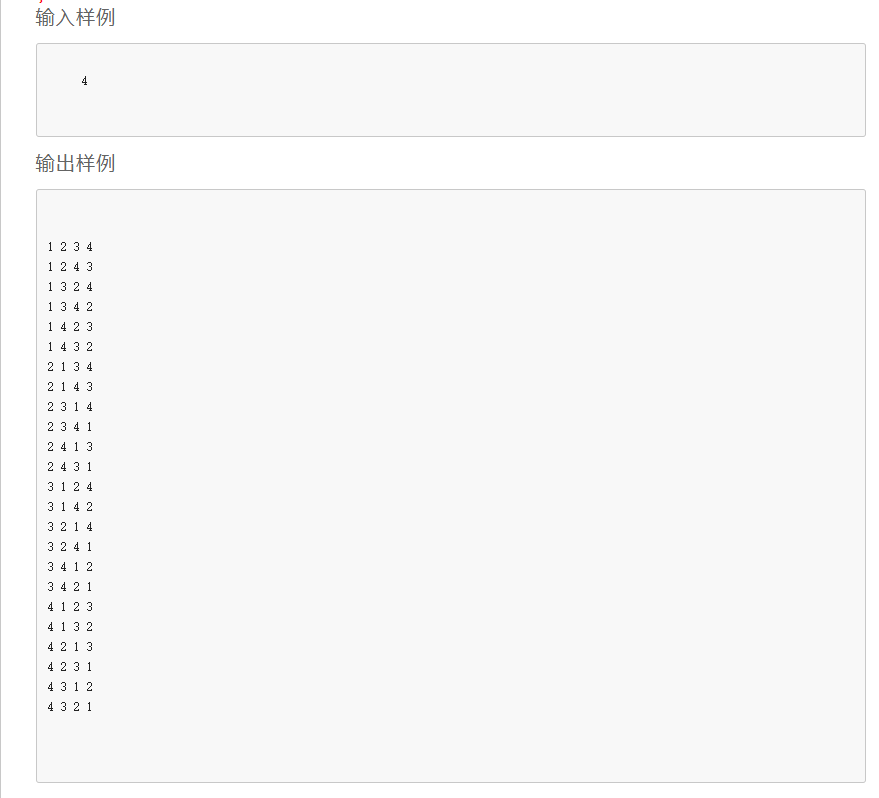
**move $s0,$zero**

**addi $t0,$t0,1**

**bne $s3,$t0,loop\_out**

**li $v0,10**

**syscall**



**.macro printf\_int(%d)**

**move $a0,%d**

**li $v0,1**

**syscall**

**.end\_macro**

**####################################**

**.macro printf\_char(%d)**

**la $a0,%d**

**li $v0,4**

**syscall**

**.end\_macro**

**#####################################**

**.macro value (%address,%offset,%value)**

**sll $s7,%offset,2**

**add $s7,%address,$s7**

**sw %value,0($s7)**

**.end\_macro**

**#####################################**

**.macro get\_value(%address,%offset,%get\_value)**

**sll $s7,%offset,2**

**add $s7,%address,$s7 # don't forget to change it as address**

**lw %get\_value,0($s7)**

**.end\_macro**

**######################################**

**.data**

**symbol: .word 0:64**

**array: .word 0:64**

**space: .asciiz " "**

**line: .asciiz "\n"**

**.text**

**##############################################**

**main:**

**li $v0,5 # scanf n**

**syscall**

**move $s0,$v0 # set s0 as n**

**la $s1,symbol # set s1 as address of symbol**

**la $s2 array # set s2 as address of array**

**li $a0,0**

**jal full\_array**

**li $v0,10**

**syscall**

**###############################################**

**full\_array:**

**move $t0,$a0 # set t0 as index**

**blt $t0,$s0,full\_array\_work # if index < n,goto work**

**#else if index >= n**

**#################################################**

**move $t1,$zero # i = 0**

**loop\_print\_array:**

**get\_value($s2,$t1,$t9)**

**printf\_int($t9)**

**printf\_char(space)**

**addi $t1,$t1,1**

**bne $t1,$s0,loop\_print\_array**

**printf\_char(line)**

**jr $ra**

**##################################################**

**full\_array\_work:**

**move $t1,$zero # set i as 0**

**loop\_full\_array\_work:**

**get\_value($s1,$t1,$t9) # get symbol[i]**

**bne $t9,$zero,loop\_full\_array\_work\_next # if symbol[i] != 0 goto next**

**#else if symbol[i] == 0**

**addi $t9,$t1,1**

**value($s2,$t0,$t9) # array[index] = i+1**

**li $t9,1**

**value($s1,$t1,$t9) # symbol[i] = 1**

**sw $ra,0($sp) #save this full\_array**

**addi $sp,$sp,-4**

**sw $t0,0($sp) # save this index**

**addi $sp,$sp,-4**

**sw $t1,0($sp) # save this i**

**addi $sp,$sp,-4**

**addi $a0,$t0,1**

**jal full\_array**

**addi $sp,$sp,4**

**lw $t1,0($sp) # get this i**

**addi $sp,$sp,4**

**lw $t0,0($sp) # get this index**

**addi $sp,$sp,4**

**lw $ra,0($sp) # get this full\_array**

**li $t9,0**

**value($s1,$t1,$t9) # symbol[i] = 0**

**loop\_full\_array\_work\_next:**

**addi $t1,$t1,1**

**bne $t1,$s0,loop\_full\_array\_work**

**jr $ra # don't forget return**

1. **封装常用的函数！！！**
2. **注意二维矩阵寻址时是按照列寻址！！！**
3. **别忘了将序号转换成地址！！！**
4. **用 name(address)的方式取数或存数，记得乘4**
5. **注意在调用宏函数时，在宏函数内部设置中间变量，别改变传入宏函数的寄存器的值**
6. **注意压栈是先sw再add，弹栈是先add再lw**
7. **千万不要忘记调用递归函数最后要return，虽然C是默认return**