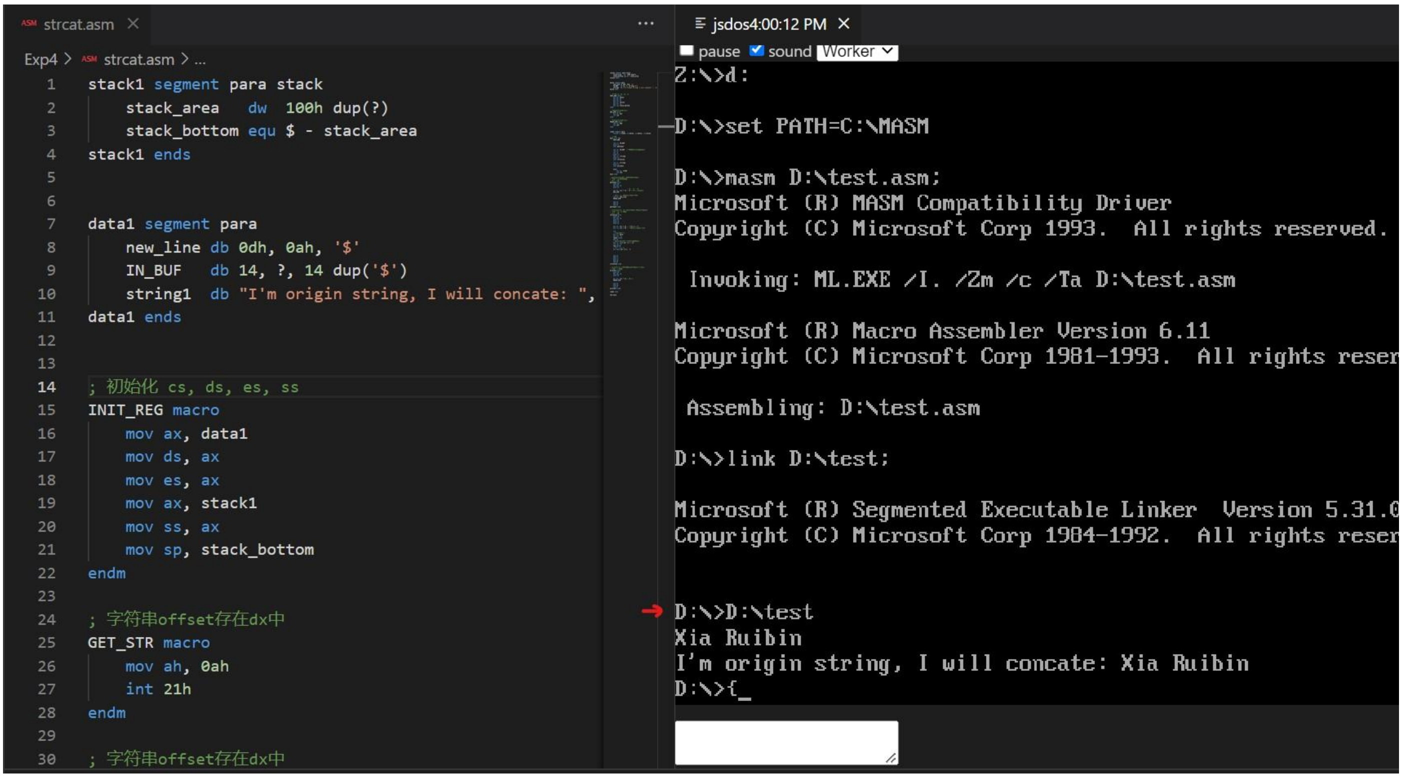
程序运行截图

第 1 题：strcat

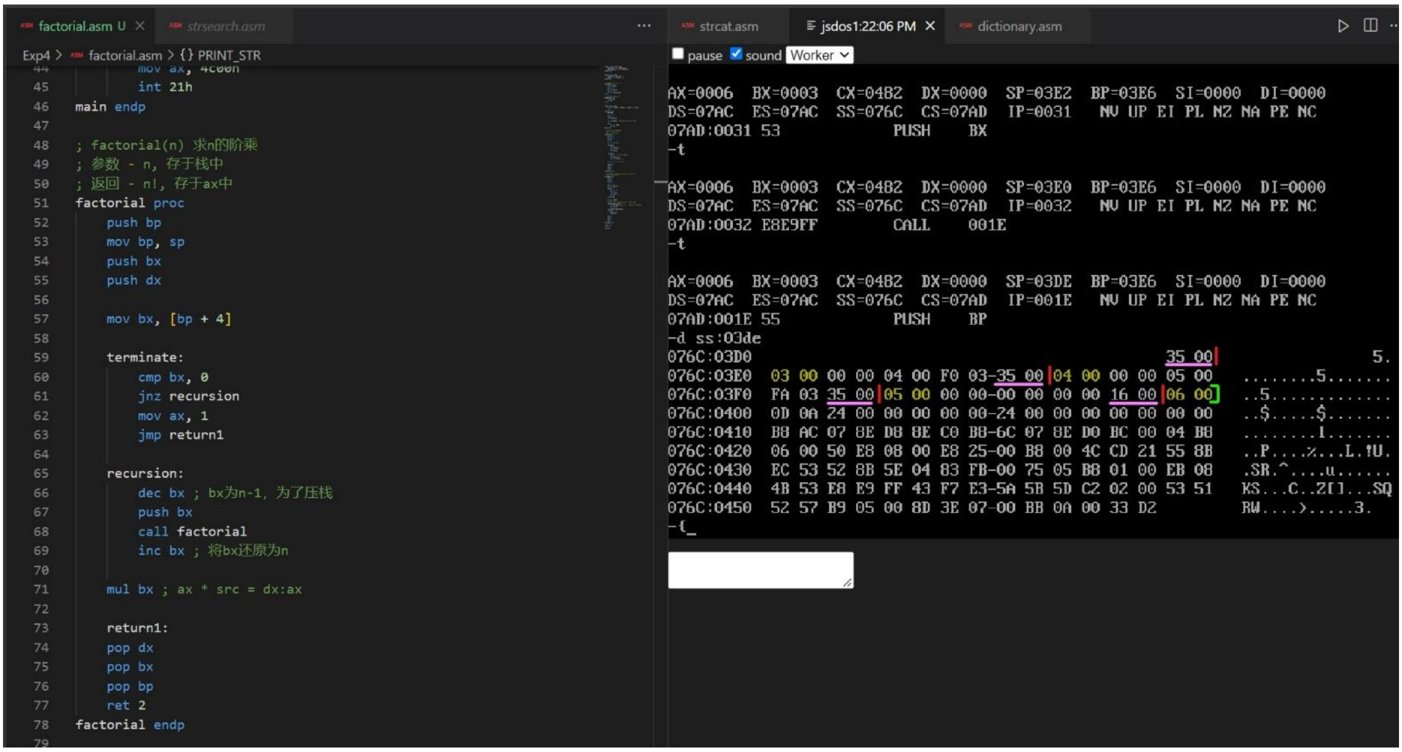


```
stack1 segment para stack
    stack_area   dw  100h dup(?)
    stack_bottom equ $ - stack_area
stack1 ends


data1 segment para
    new_line db 0dh, 0ah, '$'
    IN_BUF   db 14, ?, 14 dup('$')
    string1  db "I'm origin string, I will concate: ",
data1 ends


; 初始化 cs, ds, es, ss
INIT_REG macro
    mov ax, data1
    mov ds, ax
    mov es, ax
    mov ax, stack1
    mov ss, ax
    mov sp, stack_bottom
endm

; 字符串offset存在dx中
GET_STR macro
    mov ah, 0ah
    int 21h
endm

; 字符串offset存在dx中
```

```
Z:\>d:

D:\>set PATH=C:\MASM

D:\>masm D:\test.asm;
Microsoft (R) MASM Compatibility Driver
Copyright (C) Microsoft Corp 1993.  All rights reserved.

 Invoking: ML.EXE /I. /Zm /c /Ta D:\test.asm

Microsoft (R) Macro Assembler Version 6.11
Copyright (C) Microsoft Corp 1981-1993.  All rights reser

 Assembling: D:\test.asm

D:\>link D:\test;

Microsoft (R) Segmented Executable Linker  Version 5.31.0
Copyright (C) Microsoft Corp 1984-1992.  All rights reser


D:\>D:\test
Xia Ruibin
I'm origin string, I will concate: Xia Ruibin
D:\>{_
```

三个用栈传递参数的子程序，将 IN_BUF 拼接到 string1 后面

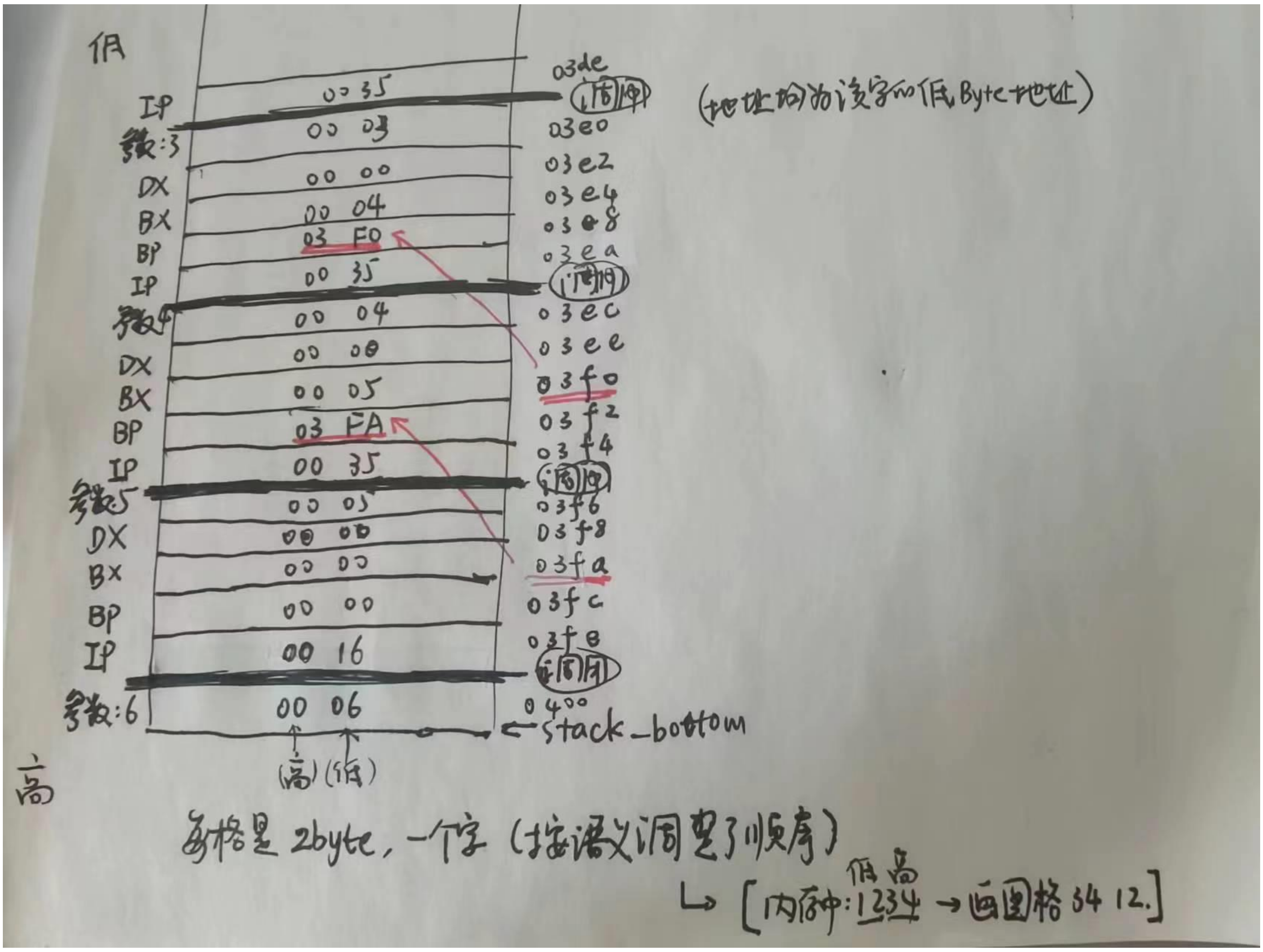
第 2 题：factorial



```
        mov ax, 4c00h
        int 21h
main endp

; factorial(n) 求n的阶乘
; 参数 - n, 存于栈中
; 返回 - n!, 存于ax中
factorial proc
    push bp
    mov bp, sp
    push bx
    push dx

    mov bx, [bp + 4]

    terminate:
        cmp bx, 0
        jnz recursion
        mov ax, 1
        jmp return1

    recursion:
        dec bx ; bx为n-1, 为了压栈
        push bx
        call factorial
        inc bx ; 将bx还原为n

    mul bx ; ax * src = dx:ax

    return1:
    pop dx
    pop bx
    pop bp
    ret 2
factorial endp
```

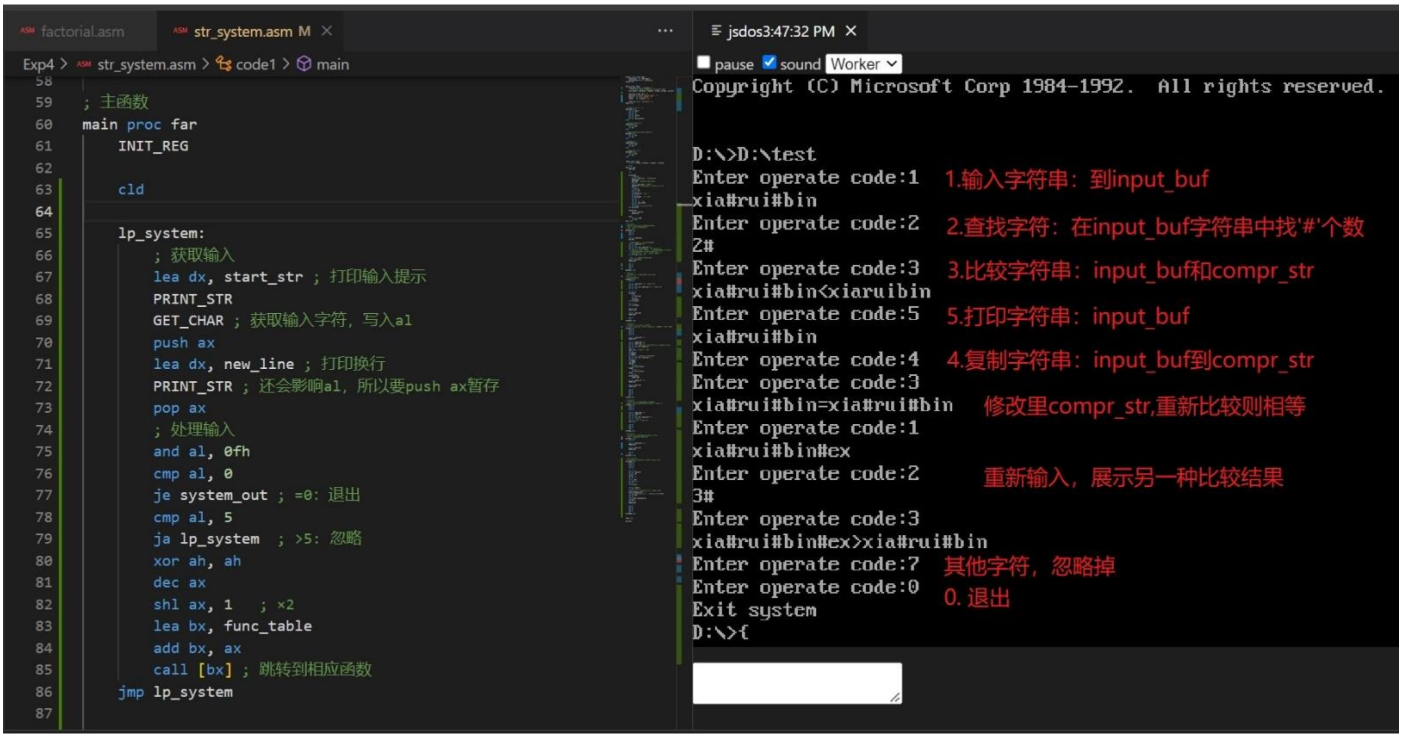
```
AX=0006  BX=0003  CX=04B2  DX=0000  SP=03E2  BP=03E6  SI=0000  DI=0000
DS=07AC  ES=07AC  SS=076C  CS=07AD  IP=0031   NV UP EI PL NZ NA PE NC
07AD:0031 53            PUSH    BX
-t

AX=0006  BX=0003  CX=04B2  DX=0000  SP=03E0  BP=03E6  SI=0000  DI=0000
DS=07AC  ES=07AC  SS=076C  CS=07AD  IP=0032   NV UP EI PL NZ NA PE NC
07AD:0032 E8E9FF        CALL    001E
-t

AX=0006  BX=0003  CX=04B2  DX=0000  SP=03DE  BP=03E6  SI=0000  DI=0000
DS=07AC  ES=07AC  SS=076C  CS=07AD  IP=001E   NV UP EI PL NZ NA PE NC
07AD:001E 55            PUSH    BP
-d ss:03de
076C:03D0                                            35 00                 5.
076C:03E0  03 00 00 00 04 00 F0 03-35 00 04 00 00 00 05 00   ........5.......
076C:03F0  FA 03 35 00 05 00 00 00-00 00 00 00 16 00 06 00   ..5.............
076C:0400  0D 0A 24 00 00 00 00 00-24 00 00 00 00 00 00 00   ..$.....$.......
076C:0410  B8 AC 07 8E D8 8E C0 B8-6C 07 8E D0 BC 00 04 B8   ........l.......
076C:0420  06 00 50 E8 08 00 E8 25-00 B8 00 4C CD 21 55 8B   ..P....%...L.!U.
076C:0430  EC 53 52 8B 5E 04 83 FB-00 75 05 B8 01 00 EB 08   .SR.^....u......
076C:0440  4B 53 E8 E9 FF 43 F7 E3-5A 5B 5D C2 02 00 53 51   KS...C..Z[]...SQ
076C:0450  52 57 B9 05 00 8D 3E 07-00 BB 0A 00 33 D2         RW....>.....3.
-{_
```

N=3 时的堆栈图见上图右侧。绿色框为栈底；红色分割线标识每一次调用；标黄的数标识每次压栈的参数；粉色下划线标识调用后第一个压栈的 IP+2（返回时的 IP 位置）



以上为手画堆栈图

第 3 题：str_system



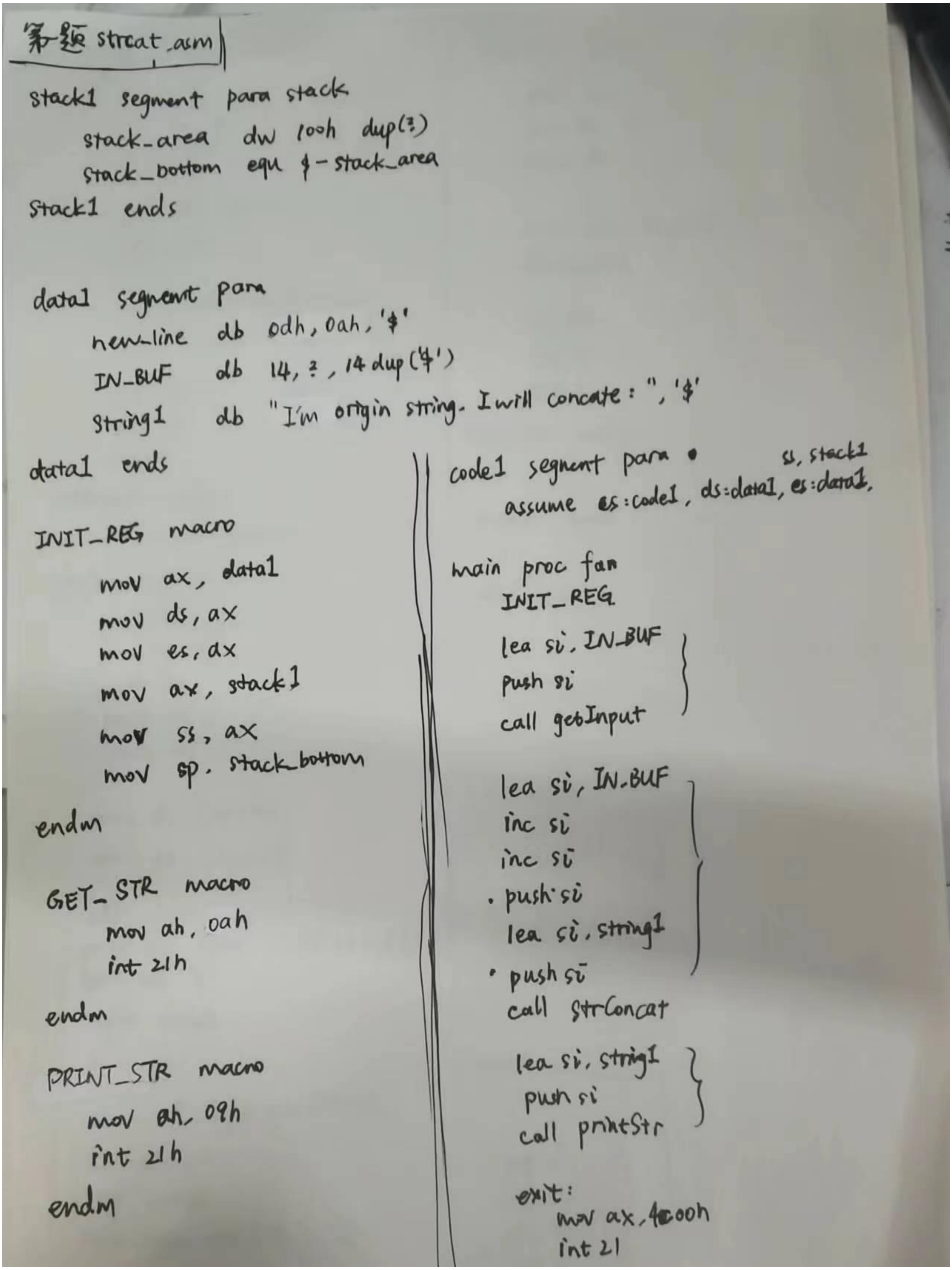
```
; 主函数
main proc far
    INIT_REG

    cld

    lp_system:
        ; 获取输入
        lea dx, start_str ; 打印输入提示
        PRINT_STR
        GET_CHAR ; 获取输入字符，写入al
        push ax
        lea dx, new_line ; 打印换行
        PRINT_STR ; 还会影响al，所以要push ax暂存
        pop ax
        ; 处理输入
        and al, 0fh
        cmp al, 0
        je system_out ; =0: 退出
        cmp al, 5
        ja lp_system  ; >5: 忽略
        xor ah, ah
        dec ax
        shl ax, 1   ; ×2
        lea bx, func_table
        add bx, ax
        call [bx] ; 跳转到相应函数
    jmp lp_system
```

```
Copyright (C) Microsoft Corp 1984-1992.  All rights reserved.


D:\>D:\test
Enter operate code:1   1.输入字符串：到input_buf
xia#rui#bin
Enter operate code:2   2.查找字符：在input_buf字符串中找'#'个数
2#
Enter operate code:3   3.比较字符串：input_buf和compr_str
xia#rui#bin<xiaruibin
Enter operate code:5   5.打印字符串：input_buf
xia#rui#bin
Enter operate code:4   4.复制字符串：input_buf到compr_str
Enter operate code:3
xia#rui#bin=xia#rui#bin   修改里compr_str,重新比较则相等
Enter operate code:1
xia#rui#bin#ex
Enter operate code:2      重新输入，展示另一种比较结果
3#
Enter operate code:3
xia#rui#bin#ex>xia#rui#bin
Enter operate code:7   其他字符，忽略掉
Enter operate code:0   0. 退出
Exit system
D:\>{
```

完整操作的含义解释见上图右侧的红色字。

由于题干描述模糊，便按自己的理解写了；要求直接通过跳转表调用函数，那就没法好好传递多样的参数，干脆直接把参数写死了。

手抄第 1 题

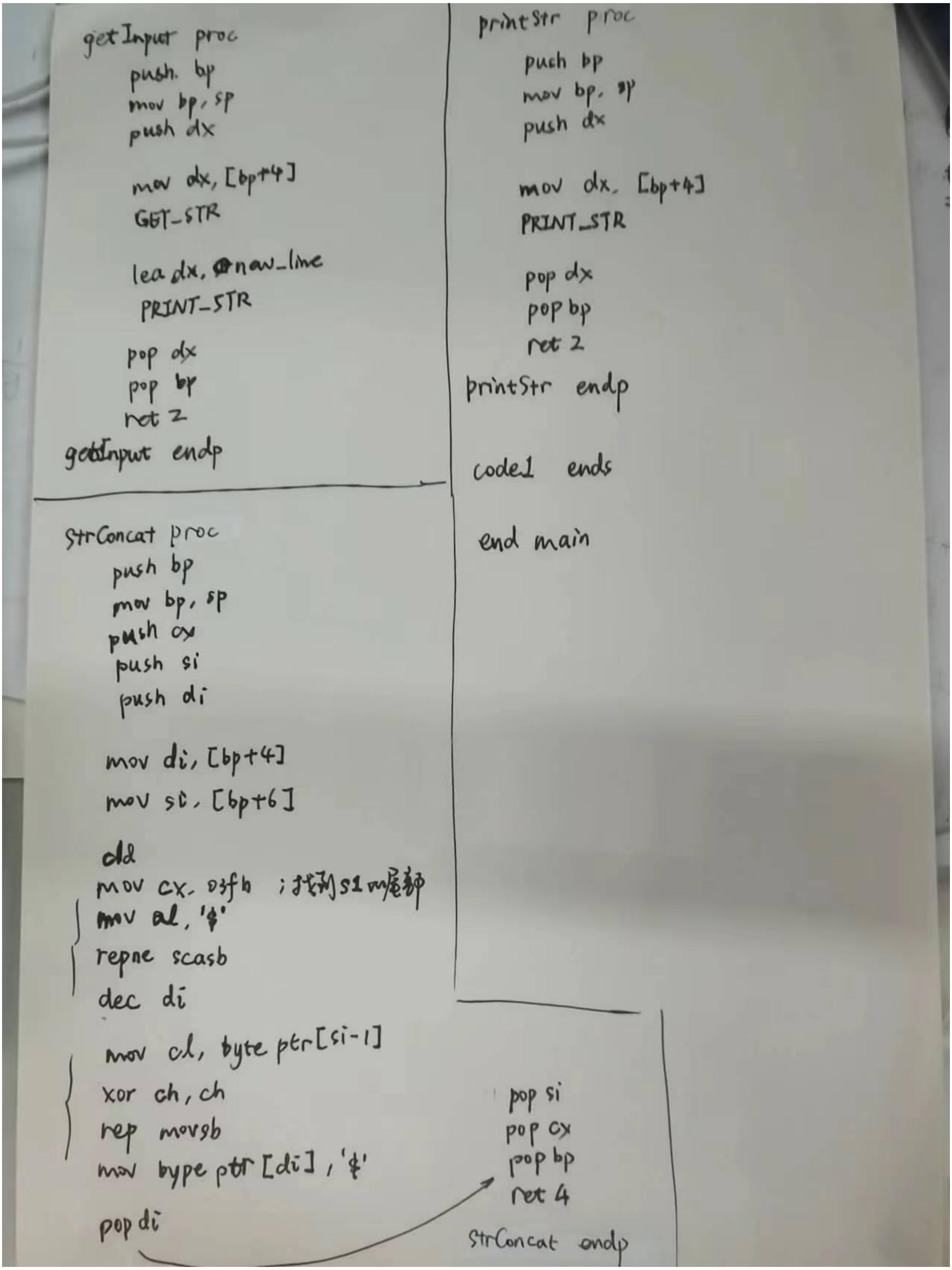


第一题 strcat.asm

```
stack1 segment para stack
    stack_area  dw  100h  dup(?)
    stack_bottom  equ  $-stack_area
stack1  ends


data1  segment  para
    new_line   db  0dh, 0ah, '$'
    IN_BUF     db  14, ?, 14 dup('$')
    string1    db  "I'm origin string. I will concate: ", '$'
data1  ends

INIT_REG  macro
    mov  ax, data1
    mov  ds, ax
    mov  es, ax
    mov  ax, stack1
    mov  ss, ax
    mov  sp, stack_bottom
endm

GET_STR  macro
    mov ah, 0ah
    int 21h
endm

PRINT_STR  macro
    mov  ah, 09h
    int 21h
endm
```

```
code1 segment para                     ss, stack1
    assume  cs:code1, ds:data1, es:data1,

main proc far
    INIT_REG
    lea si, IN_BUF
    push si
    call getInput

    lea si, IN_BUF
    inc si
    inc si
    push si
    lea si, string1
    push si
    call strConcat

    lea si, string1
    push si
    call printStr

    exit:
      mov ax, 4c00h
      int 21
```

（续）



```
getInput proc
    push bp
    mov bp, sp
    push dx

    mov dx, [bp+4]
    GET_STR

    lea dx, new_line
    PRINT_STR

    pop dx
    pop bp
    ret 2
getInput endp

strConcat proc
    push bp
    mov bp, sp
    push cx
    push si
    push di

    mov di, [bp+4]
    mov si, [bp+6]

    cld
    mov cx, 03fh   ;找到s1的尾部
    mov al, '$'
    repne scasb
    dec di

    mov cl, byte ptr[si-1]
    xor ch, ch
    rep movsb
    mov bype ptr [di], '$'

    pop di
    pop si
    pop cx
    pop bp
    ret 4
strConcat endp

printStr proc
    push bp
    mov bp, sp
    push dx

    mov dx, [bp+4]
    PRINT_STR

    pop dx
    pop bp
    ret 2
printStr endp

code1 ends

end main
```