《汇编语言》作业（二）

参考答案

（1）指出下列指令的错误

指令	错误
XCHG [SI], 30H	XCHG 要求源操作数与目的操作数不能为段寄存器以及立即数。
POP CS	POP 指令目标操作数可以是字长为 16 位或 32 位的寄存器或内存单元，以及除 CS 外的段寄存器。
SUB [SI], [DI]	SUB 指令源操作数及目标操作数不能同时为内存操作数。
ADC AX, DS	段寄存器不能进行算术运算。
ROR DX, AL	移位指令移位次数可以是立即数或者预先放进 CL 寄存器中的数值，此处使用 AL 寄存器，错误。
PUSH AH	PUSH 指令的操作数可以是 16 位或者 32 位的立即数，寄存器操作数或内存操作数。针对内存操作数，根据 PTR 运算符使用规则决定是否需要加上 PTR 运算符。本条指令错在 AH 为 8 位的寄存器操作数。

（2）

利用移位指令除以 2:

```
CODES SEGMENT
    ASSUME CS:CODES,DS:DATAS,SS:STACKS
START:
    MOV AX,DATAS
    MOV DS,AX

    MOV AL,0F7H     ;利用 AL 存储-9
    SAR AL,1        ;利用移位指令带符号除以 2
    ;下面两条指令(NOT ADD)目的在于将结果负值转换为正值，方便结果查看，可不加
    NOT AL          ;取反加一查看结果绝对值
    ADD AL,1
    MOV AH,4CH
    INT 21H
CODES ENDS
    END START
```

结果:

```
AX=0770  BX=0000  CX=0011  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=0770  ES=0760  SS=076F  CS=0770  IP=0005   NV UP EI PL NZ NA PO NC
0770:0005 B0F7          MOV     AL,F7
-T

AX=07F7  BX=0000  CX=0011  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=0770  ES=0760  SS=076F  CS=0770  IP=0007   NV UP EI PL NZ NA PO NC
0770:0007 D0F8          SAR     AL,1
-T

AX=07FB  BX=0000  CX=0011  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=0770  ES=0760  SS=076F  CS=0770  IP=0009   NV UP EI NG NZ AC PO CY
0770:0009 F6D0          NOT     AL
-T

AX=0704  BX=0000  CX=0011  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=0770  ES=0760  SS=076F  CS=0770  IP=000B   NV UP EI NG NZ AC PO CY
0770:000B 0401          ADD     AL,01
```

分析：可以看到执行 SAR 指令后，C 标变为 1，因为 SAR 将 AL 最后一位移入 C 标寄存器

利用算术类指令实现除以 2:

```
CODES SEGMENT
    ASSUME CS:CODES,DS:DATAS,SS:STACKS
START:
    MOV AX,DATAS
    MOV DS,AX

    MOV AX,0FFF7H     ;利用 AL 存储-9
    MOV BH,2     ;存储除数
    IDIV BH
    NOT AL       ;取反加一查看结果绝对值
    ADD AL,1
    MOV AH,4CH
    INT 21H
CODES ENDS
    END START
```

结果:

```
AX=FFF7  BX=0000  CX=0014  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=0770  ES=0760  SS=076F  CS=0770  IP=0008   NV UP EI PL NZ NA PO NC
0770:0008 B702          MOV     BH,02
-T

AX=FFF7  BX=0200  CX=0014  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=0770  ES=0760  SS=076F  CS=0770  IP=000A   NV UP EI PL NZ NA PO NC
0770:000A F6FF          IDIV    BH
-T

AX=FFFC  BX=0200  CX=0014  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=0770  ES=0760  SS=076F  CS=0770  IP=000C   NV UP EI PL NZ NA PO NC
0770:000C F6D0          NOT     AL
-T

AX=FF03  BX=0200  CX=0014  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=0770  ES=0760  SS=076F  CS=0770  IP=000E   NV UP EI PL NZ NA PO NC
0770:000E 0401          ADD     AL,01
```

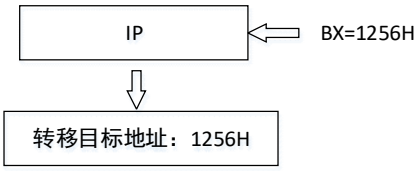
分析：可以看到标志寄存器在计算过程中未发生变化。
对比：

	对比
商	利用移位指令及算术类指令实现除以 2 的程序段执行所得商均为-4。
标志寄存器值	利用移位指令的程序段 C 标志寄存器发生改变，利用算术类指令实现功能的程序段标志寄存器未发生改变。

（3）

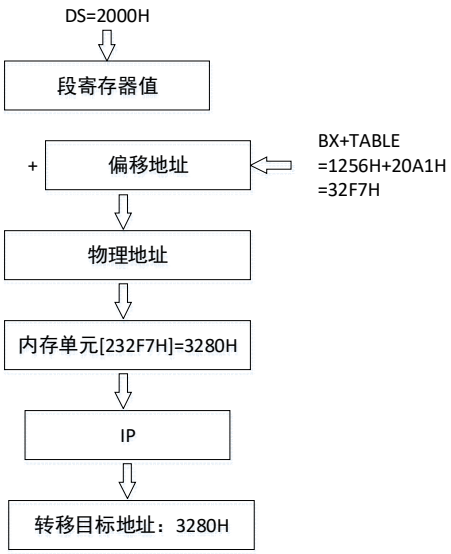
- JMP BX （寄存器寻址）

转移目的地址：1256H
寻址过程：



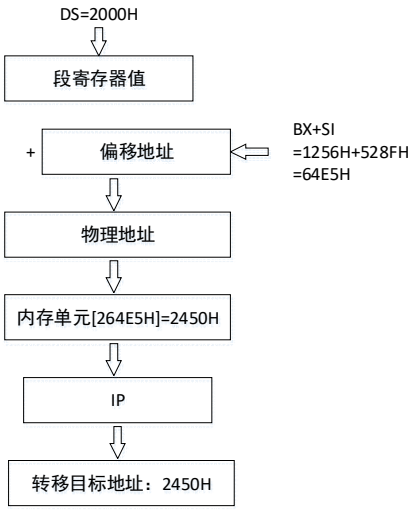
- JMP TABLE[BX] （寄存器相对寻址）

转移目的地址：3280H
目的地址存储在内存单元 232F7H 中 -> DS:(BX+TABLE) = 20000H+1256H+20A1H = 232F7H
又因为[232F7H]=3280H
所以转移目的地址为 3280H.
寻址过程：



- JMP [BX][SI] （基址加变址寻址方式 ）

转移目的地址：2450H
DS:(BX+SI) = 20000H+1256H+528FH=264E5H
又因为[264E5H]=2450H
所以转移目的地址为 2450H.
寻址过程：



（4）判断下列程序段跳转的条件

程序段	跳转条件
XOR AX, 1E1EH JE EQUAL	AX 等于 1E1EH（0001 1110 0001 1110B）。因为 JE 表示 Z 标为 1 跳转，而只有当 AX 与 1E1EH 相等时，异或结果为 0，此时 Z 标会被置 1。
TEST AL, 10000001B JNZ THERE	AL 内容第 0 位或者第 7 位为 1 时跳转。因为 JNZ 表示 Z 标为 0 跳转，TEST 执行按位与操作，若 AL 与 10000001B 按位与非零，则要求 AL 最低位（第 0 位）或最高位（第 7 位）非零。
CMP CX, 64H JB THERE	CX 内容小于 64H 时跳转。因为 JB 表示低于跳转（C 标为 1 跳转），CMP 比较 CX 与 64H 的大小，其根据 CX-64H 的结果设置标志位，当 CX 小于 64H 时，C 标置 1，满足跳转条件。

（5）假设 AX 和 SI 存放的是有符号数，DX 和 DI 存放的是无符号数，请用比较指令和条件转移指令实现以下判断：

需实现判断	实现指令
若 DX>DI，转到 ABOVE 执行	CMP DX,DI JA ABOVE
若 AX>SI，转到 GREATER 执行	CMP AX,SI JG GREATER
若 AX-SI 产生溢出，转到 OVERFLOW 执行	CMP AX,SI JO OVERFLOW
若 SI<=AX，转到 LESS_EQ 执行	CMP SI,AX JLE LESS_EQ
若 CX=0，转到 ZERO 执行	CMP CX,0 JZ ZERO
若 DI<=DX，转到 BELOW_EQ 执行	CMP DI,DX JNA BELOW_EQ

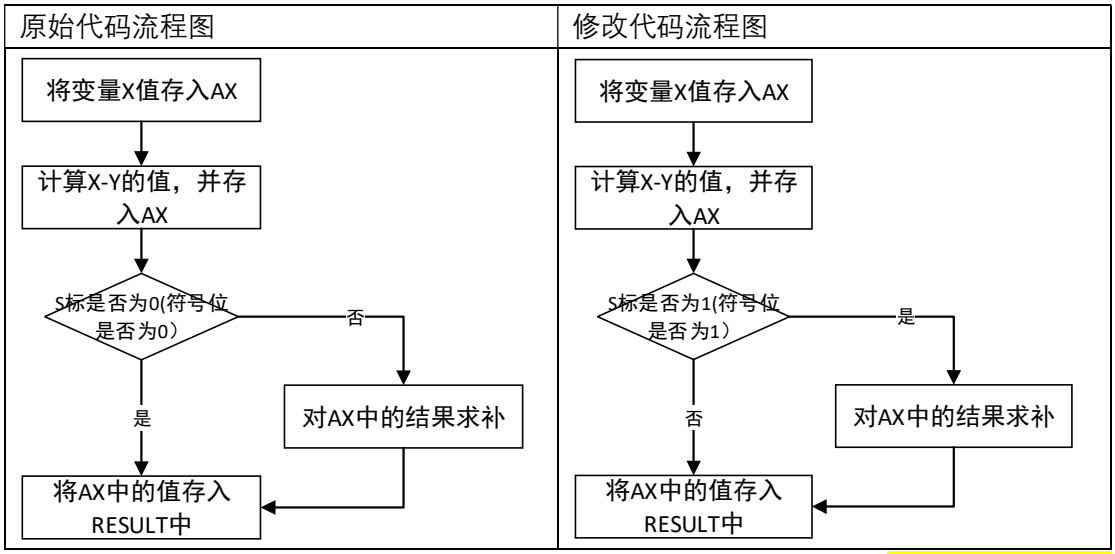
附加题：

（1）

调整代码：

```
        MOV AX, X
        SUB AX, Y
        JS ISNEG
        JMP STORE
ISNEG:  NEG AX
STORE:  MOV RESULT, AX
```

两段代码流程图对比：



对比分析：原始代码与修改后代码流程图相差不大，然而在代码设计上，主要区别在于 JMP 跳转指令的使用，过多的 JMP 指令，会破坏程序的结构性。

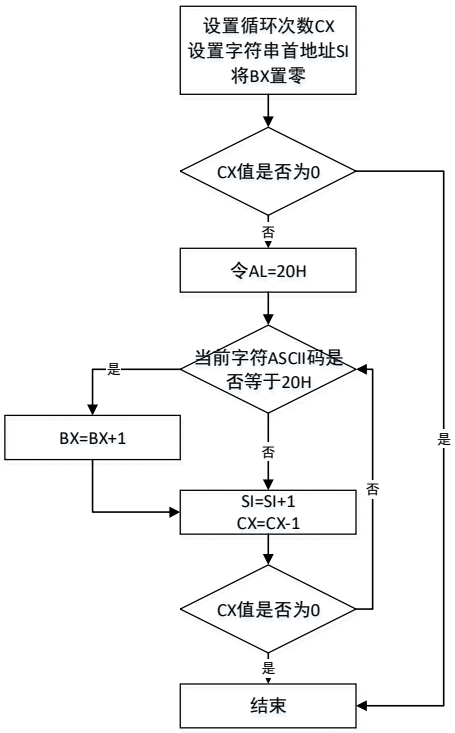
（2）

修改后的程序段：

```
    ;在字符 ASCII 码中“1”的个数为偶数时
    ;则令其最高位为“0”;否则令最高位为“1”
    AND AL,7FH ;最高位置 0，同时判断"1"的个数
    JP NEXT ;"1"的个数为偶数，转向 NEXT
    OR AL,80H   ;最高位置"1"
NEXT: …
```

（3）

a. 流程图



b. 若将其中的 JNZ NEXT 条件更改为 JZ，代码应该修改如下：

```
    MOV CX,COUNT
    MOV SI,OFFSET STRING
    XOR BX,BX
    MOV AL,20H
AGAIN:
    JCXZ DONE     ;循环开始判断 CX 是否为空
    CMP AL,ES:[SI]
    JZ NEXT     ;ZF=1 是空格，转移
    INC SI
    DEC CX
    JMP AGAIN
NEXT:
    INC BX
    INC SI
    DEC CX
    JMP AGAIN
```

c. 若去掉 JCXZ DONE，则先执行后判断 CX 是否为 0。若 CX 初始为 0，则在第一次执行完后变为-1，此时 JNZ 将永远满足，程序陷入死循环；若 CX 初始不为 0，则无影响。

d. 若采用基址加变址寻址，修改代码如下：

```
MOV CX,COUNT
    MOV BX,OFFSET STRING ;BX 存储字符串首字符偏移地址
    XOR SI,SI    ;SI=0,记录当前判断字符位置
    XOR DX,DX ;DX=0,记录空格数
```

```
    JCXZ DONE
    MOV AL,20H
AGAIN:
    CMP AL,BYTE PTR [BX+SI]
    JNZ NEXT
    INC DX
NEXT:
    INC SI
    DEC CX
    JNZ AGAIN
```