Homework 2

1 用loop指令实现

```
STKSEG SEGMENT STACK
    DW 32 DUP(0)
STKSEG ENDS

DATASEG SEGMENT
    newline DB 0Dh, 0Ah, '$'    ; 换行符
DATASEG ENDS

CODESEG SEGMENT
    ASSUME CS:CODESEG, DS:DATASEG
MAIN PROC FAR
    MOV AX, DATASEG         ; 初始化数据段
    MOV DS, AX

    ; 输出小写字母 a-z，每行 13 个
    MOV CX, 26              ; CX = 26，输出 26 个小写字母
    MOV AL, 'a'             ; AL 初始化为 'a'
    MOV BX, 13              ; BX = 13，每行输出 13 个字符

A:                           ; 主循环标签
    MOV DL, AL              ; 将当前字母放入 DL
    MOV AH, 2               ; DOS 中断功能号 2：显示字符
    INT 21H                 ; 调用中断输出字符

    INC AL                  ; AL 加 1，指向下一个字母
    DEC BX                  ; 每输出一个字符，BX 减 1
    JNZ B                   ; 如果还没输出 13 个字符，继续
    ; 输出换行符
    MOV AH, 9               ; DOS 中断功能号 9：显示字符串
    LEA DX, newline         ; 指向换行符
    INT 21H                 ; 输出换行符

    MOV BX, 13              ; 重置 BX 为 13，继续输出下一行

B:                           ; 换行标签
    LOOP A                  ; 循环输出直到 CX=0

    ; 程序结束
    MOV AX, 4C00H           ; 程序结束
    INT 21H
MAIN ENDP
CODESEG ENDS
    END MAIN
```

2 用条件跳转指令实现

```
STKSEG SEGMENT STACK
```

```
    DW 32 DUP(0)                ; 定义32个字的栈空间，初始化为 0
STKSEG ENDS

DATASEG SEGMENT
    newline DB 0Dh, 0Ah, '$'    ; 定义换行符和回车符，DOS 显示字符串时使用 '$' 结尾
DATASEG ENDS

CODESEG SEGMENT
    ASSUME CS:CODESEG, DS:DATASEG
MAIN PROC FAR
    MOV AX, DATASEG             ; 将数据段寄存器的地址放入 AX
    MOV DS, AX                  ; 将 AX 的值传送给 DS，设置数据段为 DATASEG

    MOV CX, 26                  ; 将循环计数器 CX 设为 26，表示要输出 26 个字符（a-z）
    MOV AL, 'a'                 ; 将 AL 寄存器设为 ASCII 字符 'a'，这是打印字符的起始位
置
    MOV BX, 13                  ; 将 BX 设为 13，用作换行控制的计数器，每打印 13 个字符换
行一次

A:                              ; 循环开始标签 A
    MOV DL, AL                  ; 将当前字符（AL 中的值）传送给 DL，DL 是 INT 21H 调用
时要显示的字符
    MOV AH, 2                   ; 设置 AH 为 2，表示调用 DOS 的功能 02H，输出字符
    INT 21H                     ; 调用 DOS 中断 21H 显示字符

    INC AL                      ; 将 AL 中的字符递增，下次循环时打印下一个字符
    DEC BX                      ; 将 BX 减 1，控制换行
    JNZ C                       ; 如果 BX 不为 0，跳转到标签 C（不换行）

    ; 执行到这里时，表示已经打印了 13 个字符，需要换行
    MOV AH, 9                   ; 设置 AH 为 9，表示调用 DOS 的功能 09H，输出字符串
    LEA DX, newline             ; 使用 LEA 指令将 newline 的地址加载到 DX，准备输出换行
符
    INT 21H                     ; 调用 DOS 中断 21H 输出换行符（0D 0A）

    MOV BX, 13                  ; 重置 BX 为 13，准备下一轮的换行控制

C:                              ; 标签 C，继续主循环
    DEC CX                      ; 将 CX 减 1，表示还有多少个字符要输出
    JNZ A                       ; 如果 CX 不为 0，跳转到标签 A，继续打印下一个字符

    ; 执行到这里时，表示所有字符都已输出完毕，程序结束
    MOV AX, 4C00H               ; 设置 AX 为 4C00H，DOS 功能 4C 是正常结束程序
    INT 21H                     ; 调用 DOS 中断 21H 结束程序
MAIN ENDP
CODESEG ENDS
    END MAIN
```

3 用c语言实现并反编译

3.1 c语言实现

```
#include <stdio.h>
```

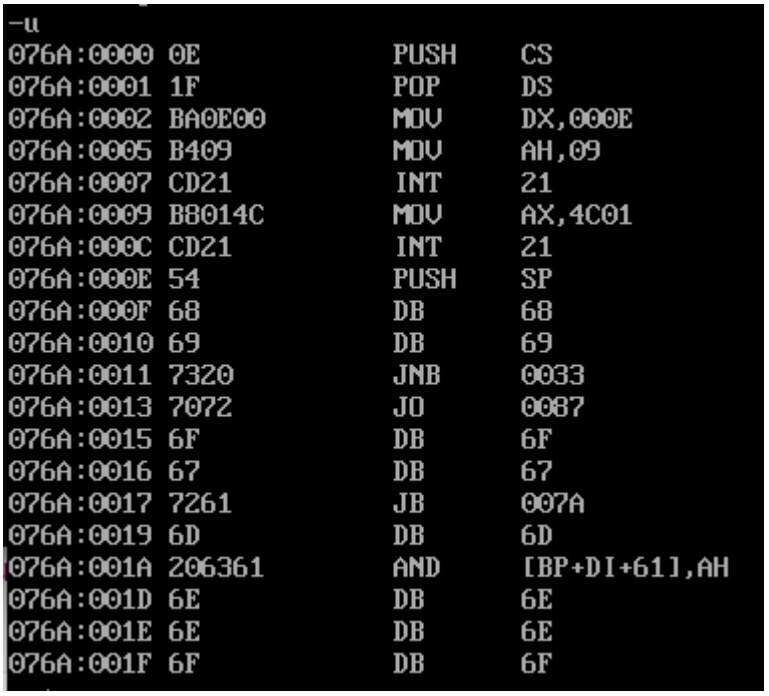
```c
int main()
{
    char ch;
    int count = 0;

    for (ch = 'a'; ch <= 'z'; ch++)
    {
        printf("%c ", ch);
        count++;

        if (count % 13 == 0) {
            printf("\n");
        }
    }
    return 0;
}
```

3.2 反编译

进入debug模式，并且执行反汇编指令-u：




```
076A:0000  0E          PUSH    CS          ; 保存代码段寄存器的值到堆栈
076A:0001  1F          POP     DS          ; 弹出堆栈顶的值到数据段寄存器
076A:0002  BA0E00      MOV     DX,000E     ; 将 000E 装入 DX 寄存器
076A:0005  B409        MOV     AH,09       ; 将 09 装入 AH 寄存器，DOS 中断 21H 的子
功能：显示字符串
076A:0007  CD21        INT     21          ; 调用 DOS 中断 21H
076A:0009  BB014C      MOV     BX,4C01     ; 将 4C01 装入 BX，DOS 中断 21H 的子功能
4C：正常终止程序
076A:000C  CD21        INT     21          ; 调用 DOS 中断 21H，程序终止
076A:000E  54          PUSH    SP          ; 保存栈指针的值到堆栈
076A:000F  68          DB      68          ; 定义一个字节常量 68
076A:0010  69          DB      69          ; 定义一个字节常量 69
076A:0011  7320        JNB     0033        ; 如果上次操作未设置进位标志，则跳转到 0033
076A:0013  7072        JO      0087        ; 如果发生溢出，则跳转到 0087
076A:0015  6F          DB      6F          ; 定义一个字节常量 6F
```

```
076A:0016  67            DB      67          ; 定义一个字节常量 67
076A:0017  7261          JB      007A        ; 如果上次操作设置了进位标志，则跳转到 007A
076A:0019  6D            DB      6D          ; 定义一个字节常量 6D
076A:001A  206361        AND     [BP+DI+61],AH ; 对内存地址 [BP+DI+61] 与 AH 做按位与操作
076A:001D  6E            DB      6E          ; 定义一个字节常量 6E
076A:001E  6F            DB      6F          ; 定义一个字节常量 6F
```