实验十二　演奏乐曲

学号： 04022212　　**姓名：** **钟　源**

一、实验任务和实验结果

请写出已通过验收的各个实验任务的具体内容、调试通过的源程序（加注释）和实验结果。

实验结果请截图，并加以必要说明。

1.**任务：**

利用 IBM-PC 机上的发音装置产生声响与音调，编制音乐演奏程序。

基本任务：

图 1-12-1 给出了乐曲“四季歌”的一段简谱。

根据图 1-12-2 程序流程图，调用 SING 子程序（其作用是从数据段中取出每个音符的频率数据和节拍时间数据，然后调用 SOUND 子程序让电脑扬声器发出相应频率和节拍的声音)，编制演奏“SONG OF FOUR SEASON”乐曲程序。

在程序的数据段中要定义频率数据（FREQ）和节拍时间数据（TIME），并以 0000H 作为频率数据结束标志。在代码段中，将频率数据首地址送 SI，节拍时间数据的首地址送 BP。

附加任务：

1）换一首乐曲演奏；

能提供乐谱，解释程序中频率和节拍数据与乐谱之间的对应关系。

2）用键盘模拟电子琴演奏乐曲；

①显示一些提示信息；

比如：高音/中音/低音的 7 个音和键盘按键字符的对应关系,按什么键结束演奏等；

②至少能模拟中音的 7 个音，最好高、中、低音都能模拟；

③弹奏时，发声可以固定时长（如 1 秒、2 秒...），最好能模拟电子琴真实的发声，键按下发声，键松开不发声。

3）实现有休止符的乐曲演奏（或者在附加任务 1 演奏的乐曲中加入休止符）。

2.源程序及注释：

（1）基础任务：演奏《四季歌》。

```
DATA    SEGMENT                                     ;定义数据段
SHOW DB  0AH,0DH, 'SING OF FOUR SEASONS:$'          ;0AH,0DH 为换行，回车
FREQ    DW  660, 660, 588, 524, 588, 524, 494, 440, 440, 440 ;查表可得频率
            DW  698, 698, 660, 588, 524, 588, 698, 660
            DW  698, 698, 660, 588, 588, 698, 660, 660, 524, 440, 524
            DW  494, 660, 588, 524, 494, 524, 440, 0

TIME    DW  100, 50, 50, 50, 50, 50, 50, 100, 100, 200      ;多少个 10ms
            DW  100, 50, 50, 50, 50, 50, 50, 400
            DW  100, 50, 50, 100, 50, 50, 100, 50, 50, 100, 100
            DW  100, 100, 50, 50, 50, 50, 400
DATA    ENDS

STACK   SEGMENT PARA STACK 'STACK'                  ;定义堆栈段
        DW  200 DUP(?)
STACK   ENDS

CODE    SEGMENT
    ASSUME  CS:CODE, DS:DATA,SS:STACK

START:
    MOV AX,DATA
    MOV DS,AX               ;初始化数据段
    MOV AX,STACK
    MOV SS,AX               ;初始化堆栈段
    MOV DX,OFFSET SHOW
    MOV AH,09
    INT 21H                 ;显示提示信息
    MOV SI,OFFSET FREQ      ;将频率数据地址给 SI
    MOV BP,OFFSET TIME      ;将节拍时间数据的地址给 DI
    CALL SING
    MOV AH,4CH
    INT 21H                 ;程序结束，返回 DOS
```

```
SING PROC NEAR
            PUSH DI
            PUSH SI
            PUSH BP
            PUSH BX
RETP:   MOV DI,[SI]         ;取频率
            CMP DI,0        ;0 意味着程序结束
            JE END_SING
    MOV BX,DS:[BP]          ;取节拍
            CALL SOUND      ;参数为 DI 和 BX
            ADD SI,2
            ADD BP,2
            JMP RETP
END_SING:
        POP  BX
         POP  BP
        POP  SI
         POP  DI
         RET
SING ENDP

SOUND PROC  NEAR
            PUSH AX
        PUSH BX             ;BX 节拍时间数据
            PUSH CX
        PUSH DX
            PUSH DI         ;入口参数 DI 给定频率数据
            MOV AL,10110110B;8253 初始化(通道 2，方式 3，产生方波信号)
            OUT 43H,AL      ;43H 端口是 8253 的命令寄存器

            MOV DX,12H      ;计算时间常数
            MOV AX,34DCH
            DIV DI

            OUT 42H,AL      ;给 8253 通道 2 设置计数初值
            MOV AL,AH
            OUT 42H,AL

            IN AL,61H       ;读 8255B 口
            MOV AH,AL
            OR AL,3         ;8255 PB1PB0 置 1，开喇叭
            OUT 61H,AL
DELAY:
    MOV CX,15000
```

```
DL10ms:
    LOOP  DL10ms            ;延时 10ms
    DEC BX                  ;BX—节拍时间对应 10ms 的倍数，如:BX=100,节拍时间=10ms*100=1s
            JNZ DELAY
            MOV AL,AH
            OUT 61H,AL      ;8255 PB1PB0 恢复为零，关喇叭
            POP DI
            POP DX
            POP CX
            POP BX
        POP AX
            RET
SOUND   ENDP

CODE ENDS
    END START
```

（2）附加任务 2：用键盘模拟电子琴演奏乐曲。

```
DATA    SEGMENT                                     ;定义数据段
SHOWM   DB  0AH,0DH, 'Please play the virtual piano.$'
SHOW1   DB  0AH,0DH, 'Low_pitch   :Q,W,E,R,T,Y,U.$'
SHOW2   DB  0AH,0DH, 'Middle_pitch:A,S,D,F,G,H,J.$'
SHOW3   DB  0AH,0DH, 'High_pitch  :Z,X,C,V,B,N,M.$'     ;0AH,0DH 为换行，回车
SHOWN   DB  0AH,0DH,'Please press key O to end the piano.$'
FREQ    DW  131,147,165,175,196,220,247                 ;查表可得频率
        DW  262,294,330,349,392,440,494
        DW  524,588,660,698,784,880,988
ISPLAY  DB 1
ISBREAK DB 0
DATA    ENDS
STACK   SEGMENT PARA STACK 'STACK'                      ;定义堆栈段
        DW  200 DUP(?)
STACK   ENDS

CODE    SEGMENT
    ASSUME  CS:CODE, DS:DATA,SS:STACK

START:
    MOV AX,DATA
    MOV DS,AX
    MOV AX,STACK
    MOV SS,AX
```

```
    MOV DX,OFFSET SHOWM
    MOV AH,09H              ;显示文字
    INT 21H
    MOV DX,OFFSET SHOW1     ;显示文字
    INT 21H
    MOV DX,OFFSET SHOW2     ;显示文字
    INT 21H
    MOV DX,OFFSET SHOW3     ;显示文字
    INT 21H
    MOV DX,OFFSET SHOWN     ;显示文字
    INT 21H

    MOV  AL,9               ; 指定要读取的中断向量号，这里是 9 号中断，即键盘中断。
    MOV  AH,35H             ; 35H 功能号，读取中断向量
    INT  21H
    MOV  AX,BX              ; 保存原来的中断向量偏移地址
    PUSH AX
    MOV  AX,ES              ; 保存原来的中断向量段地址
    PUSH AX
    CLI
    MOV DX,OFFSET KEYINT    ; 将新的键盘中断处理程序的偏移地址加载到 DX 寄存器
    MOV AX,SEG KEYINT       ; 将新的键盘中断处理程序的段地址加载到 AX 寄存器
    MOV BX,DS               ; 保存当前数据段地址
    MOV DS,AX               ; 设置数据段为新的键盘中断处理程序段地址
    MOV AL,9
    MOV AH,25H              ; 25H 功能号，设置中断向量
    INT 21H
    MOV DS,BX               ; 恢复原来的数据段地址
    STI

    LEA BX,FREQ
AGAIN:
    CMP ISBREAK,0
    JE  AGAIN               ; 检查是否退出(若如果 ISBREAK 为 0，跳转回 AGAIN，继续循环)
    CLI

    POP     AX              ; 从堆栈中弹出先前保存的原中断向量偏移地址。
    MOV     BX,DS
    MOV     DS,AX
    POP     AX              ; 从堆栈中弹出先前保存的原中断向量段地址。
    MOV     DX,AX
    MOV     AL,9
    MOV     AH,25H
    INT     21H             ; 调用 DOS 中断，设置 9 号中断向量为原中断处理程序的地址。
```

```
    MOV     DS,BX
    STI

    MOV     AH,4CH          ; 结束程序返回 DOS
    INT     21H

;————————————————KEYINT 子程序————————————————
;根据按键产生对应的音调，并控制扬声器的发声
KEYINT PROC
;开启中断和保存寄存器状态
    STI                     ;开中断
    PUSH    AX              ;保护入栈
    PUSH    BX
    PUSH    SI
;读取按键扫描码
    IN  AL,60H
;检测是否为按键释放
    CMP AL,80H              ;如果是释放按键，高位为 1，不发声
    JAE GO
;检测是否按下结束键
    CMP AL,18H              ;如果按到字母 O，电子琴程序结束
    JE  BREAK
;判断按键是否在有效范围内并处理音调
;;低音区
    CMP  AL,10H             ;判断是否按在可以发声的按键
    JB   GO                 ;低于 10H 则不在有效范围内
    CMP  AL,17H
    JAE  JUDGE1             ;高于或等于 17H 则在中音区
    MOV  AH,0               ;清空 AH，准备计算频率表的索引。
    SUB  AL,10H             ;将扫描码减去 10H，计算出在低音区的索引。
    ADD  AL,AL              ;索引乘以 2，得到频率表的实际索引(由于频率表占用 1 字，扫描码占用 1 字节)。
    MOV  SI,AX              ;将索引存入 SI
    MOV  ISPLAY,1           ;设置发声标志位。
    CALL SOUND              ;调用 SOUND 子程序发声。
    JMP  GO2                ;跳转到 GO2，进行中断结束处理。
;;中音区
JUDGE1:
    CMP  AL,1EH             ;判断是否在中音区
    JB   GO                 ;低于 1EH 则不在有效范围内
    CMP  AL,25H
    JAE  JUDGE2             ;高于或等于 25H 则在高音区
    MOV  AH,0
    SUB  AL,17H
    ADD  AL,AL
```

```
    MOV  SI,AX
    MOV  ISPLAY,1
    CALL SOUND
    JMP  GO2                ;其他处理类似低音区
;;高音区
JUDGE2:
    CMP  AL,2CH             ;判断是否在高音区
    JB   GO                 ;低于 2CH 则不在有效范围内
    CMP  AL,33H
    JAE  GO                 ;高于或等于 33H 则同样不在范围内
    MOV  AH,0
    SUB  AL,1EH
    ADD  AL,AL
    MOV  SI,AX
    MOV  ISPLAY,1
    CALL SOUND
    JMP  GO2                ;其他处理类似低音区
;设置结束标志位
BREAK:
    MOV     ISBREAK,1
    JMP     GO2             ;设置结束标志位，通知主程序结束。
;处理未在有效范围内的按键
GO:
    MOV ISPLAY,0
    CALL    SOUND
;结束处理
GO2:
    IN  AL,61H
    OR  AL,80H              ;将 PB7 置为 1
    OUT 61H,AL
    AND AL,7FH              ;将 PB7 清 0
    OUT 61H,AL
    MOV AL,20H
    OUT 20H,AL              ;中断结束，复位

    POP SI
    POP BX
    POP AX
    IRET
KEYINT  ENDP
;————————————————SOUND 子程序————————————————
;根据键盘中断处理程序传递的频率信息控制扬声器发声
SOUND   PROC NEAR
;保存寄存器状态
```

```
    PUSH    AX
    PUSH    BX
    PUSH    DX
    PUSH    DI              ;入口参数 DI 给定频率数据
;加载频率值
    MOV DI,[BX+SI]          ;BX 存放频率表的基址，SI 存放索引值
;设置 8253 计数器
    MOV AL,0B6H             ;8253 初始化（控制字：通道 2，方式 3，产生方波信号）
    OUT 43H,AL              ;43H 端口是 8253 的命令寄存器
;计算并设置计数初值
    MOV DX,12H
    MOV AX,34DCH            ;用于计算频率的时间常数
    DIV DI                  ;计算公式：1234DCH÷(给定频率)，得到计数初值
    OUT 42H,AL              ;给 8253 通道 2 设置计数初值，先写低八位
    MOV AL,AH
    OUT 42H,AL              ;后写高八位
;控制扬声器
    AND AL,0FCH             ;8255PB1PB0 置 0，关喇叭
    OUT 61H,AL
;检查是否需要发声
    CMP ISPLAY,0
    JE  NOO
;开启扬声器
    OR  AL,3
    OUT 61H,AL
;恢复寄存器状态并返回
NOO:
    POP DI
    POP DX
    POP BX
    POP AX
    RET
SOUND   ENDP
CODE    ENDS
    END START
```

（3）附加任务 1+3：换一首有休止符的乐曲演奏。

```
DATA    SEGMENT                                 ;定义数据段
SHOW DB  0AH,0DH, 'ONE LAST KISS:$'              ;0AH,0DH 为换行，回车
FREQ    DW  002,392,440,440,440,001,440,330,294,001,330,294,294,294,001,001,002
        DW  392,440,440,440,001,440,330,330,294,001,330,294,001,001,262,001,002
        DW  382,440,440,440,001,440,330,330,294,001,330,294,001,002,262,001
        DW  440,440,440,001,440,330,294,001,330,294,001,001,294,001,002
```

```
        DW  392,440,440,440,001,440,330,330,294,001,330,294,001,002,262,262,001
        DW  392,440,440,440,001,440,001,330,294,001,330,294,001,001,262,001,002
        DW  392,440,440,440,001,440,330,330,294,001,330,294,001,001,001
        DW  330,294,262,262,001,001,002
        ;副歌
        DW  392,440,440,001,392,440,001
        DW  588,660,392,440,392,392,440,440,440
        DW  440,440,001,002,440,660,001
        DW  588,524,524,660,588,524,524,588,588
        DW  588,440,001,001,002,392,440,392
        DW  588,660,392,440,392,392,440,440,440
        DW  440,440,001,002,440,660,001
        DW  588,524,524,623,588,524,524,588,001,002
        ;oh,oh,oh
        DW  440,001,440,001,440,001,440,001,524,588,001
        DW  440,001,440,001,440,001,660,588,524,001
        DW  440,001,440,001,440,001,440,001,002,440,660,001
        DW  588,524,524,588,660,588,524,524,588,001,002
        DW  440,001,440,001,440,001,440,001,524,588,001
        DW  440,001,440,001,440,440,001,440,001,001
        DW  440,001,440,001,440,001,660,001,002,440,660,001
        DW  588,524,524,588,660,588,524,524,588,440,001,002,000
        ;查表可得频率,001 为休止符

TIME    DW  002,025,025,025,025,005,025,050,025,005,025,075,005,025,025,005,002
        DW  025,025,025,025,005,025,025,025,025,005,025,025,050,005,100,005,002
        DW  025,025,025,025,005,025,025,025,025,005,025,075,050,002,050,005
        DW  050,025,025,005,025,050,025,005,025,025,050,005,100,005,002
        DW  025,025,025,025,005,025,025,025,025,005,050,050,050,002,025,025,005
        DW  025,025,025,025,005,025,025,025,025,005,025,025,050,005,100,005,002
        DW  025,025,025,025,005,025,025,025,025,005,050,050,005,100,005
        DW  050,100,050,100,100,005,002
        ;副歌
        DW  050,150,050,050,050,050,005
        DW  050,050,050,050,050,050,050,050,005
        DW  100,100,100,002,050,050,005
        DW  050,050,050,050,050,050,025,075,005
        DW  050,050,100,100,002,050,050,005
        DW  050,050,050,050,050,050,050,050,005
        DW  100,100,100,002,050,050,005
        DW  050,050,050,050,050,050,025,075,005,002
        ;oh,oh,oh
        DW  050,025,025,050,050,025,025,050,050,050,005
        DW  050,025,025,050,050,025,075,050,050,005
```

```
        DW  050,025,025,050,050,025,025,050,002,050,050,005
        DW  050,050,050,025,025,050,050,025,075,005,002
        DW  050,025,025,050,050,025,025,050,050,050,005
        DW  050,025,025,050,050,050,050,050,050,005
        DW  050,025,025,050,050,025,025,050,002,050,050,005
        DW  050,050,050,025,025,050,050,025,075,100,100,002,000
        ;全音符 100，二分音符 50，四分音符 25

LINE1    DB  0AH,0DH,'The first time I went to the Louvre'
SLEN1 EQU $-LINE1;
LINE2    DB  0AH,0DH,'It does not feel like anything special'
SLEN2 EQU $-LINE2;
LINE3    DB  0AH,0DH,'Because I had already met'
SLEN3 EQU $-LINE3;
LINE4    DB  0AH,0DH,'My own Mona Lisa'
SLEN4 EQU $-LINE4;
LINE5    DB  0AH,0DH,'The day I first met you'
SLEN5 EQU $-LINE5;
LINE6    DB  0AH,0DH,'The gear start revolving'
SLEN6 EQU $-LINE6;
LINE7    DB  0AH,0DH,'Can not stop the feeling of what you are about to lose'
SLEN7 EQU $-LINE7;
LINE8    DB  0AH,0DH,'I mean, it has been a lot of times'
SLEN8 EQU $-LINE8;
LINE9    DB  0AH,0DH,'Let us have another kiss'
SLEN9 EQU $-LINE9;
LINE10   DB  0AH,0DH,'Can you give me one last kiss?'
SLEN10 EQU $-LINE10;
LINE11   DB  0AH,0DH,'Things I do not want to forget'
SLEN11 EQU $-LINE11;
LINE12   DB  0AH,0DH,'Oh oh oh oh oh oh oh oh woo~ oh oh oh oh'
SLEN12 EQU $-LINE12;
LINE13   DB  0AH,0DH,'Things I do not want to forget'
SLEN13 EQU $-LINE13;
LINE14   DB  0AH,0DH,'Oh oh oh oh oh oh oh oh woo~ oh oh oh oh'
SLEN14 EQU $-LINE14;
LINE15   DB  0AH,0DH,'I love you more than you will ever know'
SLEN15 EQU $-LINE15;
LINE16   DB  0AH,0DH,'ONE LAST KISS END'
SLEN16 EQU $-LINE16;
;0AH,0DH 为换行，回车
Count DB 0;
DATA    ENDS
```

```
STACK   SEGMENT PARA STACK 'STACK'  ;定义堆栈段
        DW  400 DUP(?)
STACK   ENDS

CODE    SEGMENT
    ASSUME  CS:CODE, DS:DATA,SS:STACK

START:
;初始化数据段和堆栈段
    MOV AX,DATA
    MOV DS,AX
        MOV ES,AX
    MOV AX,STACK
    MOV SS,AX
;显示提示信息
    MOV DX,OFFSET SHOW
    MOV AH,09
    INT 21H
;地址传给寄存器
    MOV SI,OFFSET FREQ          ;将"频率"数据地址给     SI
    MOV BP,OFFSET TIME          ;将"节拍"时间数据地址给 BP
;调用 SING 主程序
    CALL SING
;程序结束，返回 DOS
    MOV AH,4CH
    INT 21H
;————————————————SING 子程序————————————————
SING PROC NEAR
;保存寄存器状态
    PUSH DI
    PUSH SI
    PUSH BP
    PUSH BX
RETP:
    MOV DI,DS:[SI]              ;取频率给 DI
    CMP DI,0                        ;0 意味着程序结束
    JE END_SING
    MOV BX,DS:[BP]              ;取节拍给 BX
    CALL SOUND                      ;参数为 DI 和 BX
    ADD SI,2
    ADD BP,2
    JMP RETP
;恢复寄存器状态并返回
END_SING:
```

```
    POP  BX
    POP  BP
    POP  SI
    POP  DI
    RET
SING ENDP
;————————————————SOUND 子程序————————————————
;根据频率信息控制扬声器发声
SOUND PROC  NEAR
;保存寄存器状态
    PUSH AX
    PUSH BX                         ;BX 节拍时间数据
    PUSH CX
    PUSH DX
    PUSH DI                         ;DI 给定频率数据
;歌词显示的实现
    CMP DI,2                            ;判断是否为休止符（只延时，不发声）
    JNE SOUND_NEXT
    CALL DRAW;参数无，直接画即可
SOUND_NEXT:
;设置 8253 计数器
    MOV AL,10110110B                ;8253 初始化(通道 2，方式 3，产生方波信号)
    OUT 43H,AL                      ;43H 端口是 8253 的命令寄存器
;休止符的实现
    CMP DI,1                            ;判断是否为休止符（只延时，不发声）
    JE  DELAY
    CMP DI,2                            ;判断是否为休止符（只延时，不发声）
    JE  DELAY
;计算并设置计数初值
    MOV DX,12H                      ;计算折算频率（时间常数）
    MOV AX,34DCH                        ;1234CDH 除以给定频率
    DIV DI
    OUT 42H,AL                      ;给 8253 通道 2 设置计数初值（先写低字节，再写高字节）
    MOV AL,AH
    OUT 42H,AL
;打开扬声器
    IN  AL,61H                  ;读 8255B 口
    MOV AH,AL
    OR  AL,3                    ;8255——PB1PB0 置 1，开喇叭
    OUT 61H,AL
;延时循环（调用了 BX 的信息）
DELAY:
    MOV CX,15000
DL10ms:
```

```
    LOOP  DL10ms                    ;延时 10ms
    DEC BX                  ;BX—节拍时间对应 10ms 的倍数，如:BX=100,节拍时间=10ms*100=1s
    JNZ DELAY
;关闭扬声器
    MOV AL,AH
    OUT 61H,AL                      ;8255——PB1PB0 恢复为零，关喇叭
;恢复寄存器状态并返回
    POP DI
    POP DX
    POP CX
    POP BX
    POP AX
    RET
SOUND   ENDP
;————————————————DRAW 子程序————————————————
;显示歌词
DRAW PROC  NEAR
;保存寄存器状态
    PUSH AX
    PUSH BX                         ;BX 取地址
    PUSH CX
    PUSH DX
    PUSH SI
    PUSH BP                             ;DI 给定频率数据
    PUSH DI                            ;DI 给定频率数据
;计数器加 1
   LEA SI, Count
   MOV AL, DS:[SI]
   INC AL
   MOV DS:[SI], AL
;清屏
    MOV AH,06H              ; 调用功能号 06H
    MOV AL,0
    MOV CH,0
    MOV CL,0
    MOV DH,24
    MOV DL,79
    MOV BH,00111111B    ;不闪烁（0），青底（011），高亮（1），白字（111）
    INT 10H
;输出
    MOV AH,13H          ; 调用功能号 13H
    MOV DH,0AH          ;行
    MOV DL,08H          ;列
    MOV AL,01H
```

```
    MOV BL,00111111B    ;不闪烁（0），青底（011），高亮（1），白字（111）
    MOV BH,00H
SHOW1:
    CMP BYTE PTR[SI],1;
    JNE SHOW2
    LEA BP,LINE1
    MOV CX,SLEN1
    JMP SHOW_END
SHOW2:
    CMP BYTE PTR[SI],2;
    JNE SHOW3
    LEA BP,LINE2
    MOV CX,SLEN2
    JMP SHOW_END
SHOW3:
    CMP BYTE PTR[SI],3;
    JNE SHOW4
    LEA BP,LINE3
    MOV CX,SLEN3
    JMP SHOW_END
SHOW4:
    CMP BYTE PTR[SI],4;
    JNE SHOW5
    LEA BP,LINE4
    MOV CX,SLEN4
    JMP SHOW_END
SHOW5:
    CMP BYTE PTR[SI],5;
    JNE SHOW6
    LEA BP,LINE5
    MOV CX,SLEN5
    JMP SHOW_END
SHOW6:
    CMP BYTE PTR[SI],6;
    JNE SHOW7
    LEA BP,LINE6
    MOV CX,SLEN6
    JMP SHOW_END
SHOW7:
    CMP BYTE PTR[SI],7;
    JNE SHOW8
    LEA BP,LINE7
    MOV CX,SLEN7
    JMP SHOW_END
```

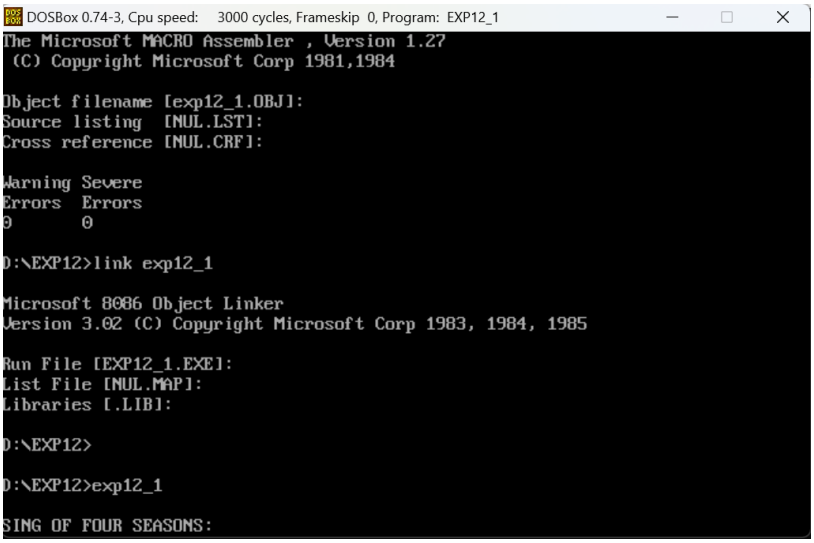
```
SHOW8:
    CMP BYTE PTR[SI],8;
    JNE SHOW9
    LEA BP,LINE8
    MOV CX,SLEN8
    JMP SHOW_END
SHOW9:
    CMP BYTE PTR[SI],9;
    JNE SHOW10
    LEA BP,LINE9
    MOV CX,SLEN9
    JMP SHOW_END
SHOW10:
    CMP BYTE PTR[SI],10;
    JNE SHOW11
    LEA BP,LINE10
    MOV CX,SLEN10
    JMP SHOW_END
SHOW11:
    CMP BYTE PTR[SI],11;
    JNE SHOW12
    LEA BP,LINE11
    MOV CX,SLEN11
    JMP SHOW_END
SHOW12:
    CMP BYTE PTR[SI],12;
    JNE SHOW13
    LEA BP,LINE12
    MOV CX,SLEN12
    JMP SHOW_END
SHOW13:
    CMP BYTE PTR[SI],13;
    JNE SHOW14
    LEA BP,LINE13
    MOV CX,SLEN13
    JMP SHOW_END
SHOW14:
    CMP BYTE PTR[SI],14;
    JNE SHOW15
    LEA BP,LINE14
    MOV CX,SLEN14
    JMP SHOW_END
SHOW15:
    CMP BYTE PTR[SI],15;
```

```
    JNE SHOW16
    LEA BP,LINE15
    MOV CX,SLEN15
    JMP SHOW_END
SHOW16:
    CMP BYTE PTR[SI],16;
    JNE SHOW_END
    LEA BP,LINE16
    MOV CX,SLEN16
SHOW_END:
    INT 10H
;恢复寄存器状态并返回
    POP DI
    POP BP
    POP SI
    POP DX
    POP CX
    POP BX
    POP AX
    RET
DRAW   ENDP

CODE ENDS
END START
```

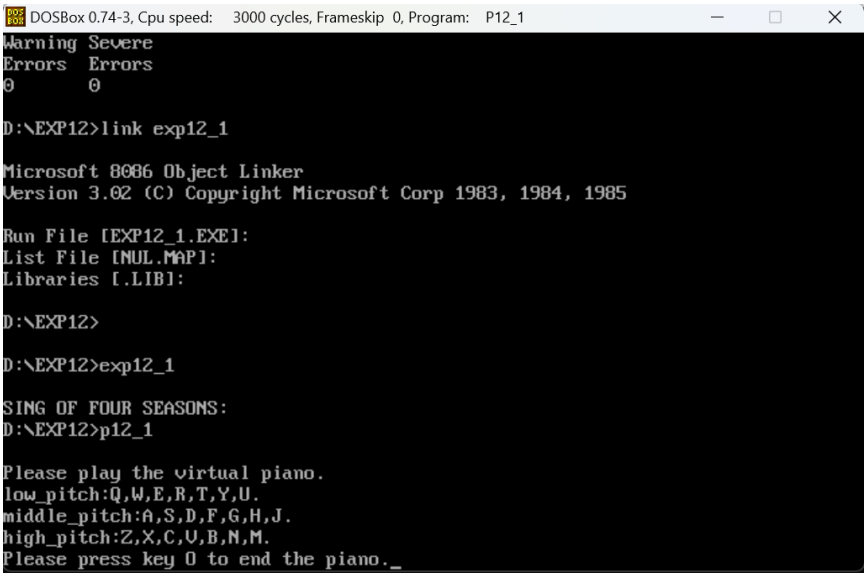
3.**实验结果：**

（1）基础任务：演奏《四季歌》。



实现了演奏《四季歌》并给出相应提示信息的功能。

（2）附加任务 2：用键盘模拟电子琴演奏乐曲。



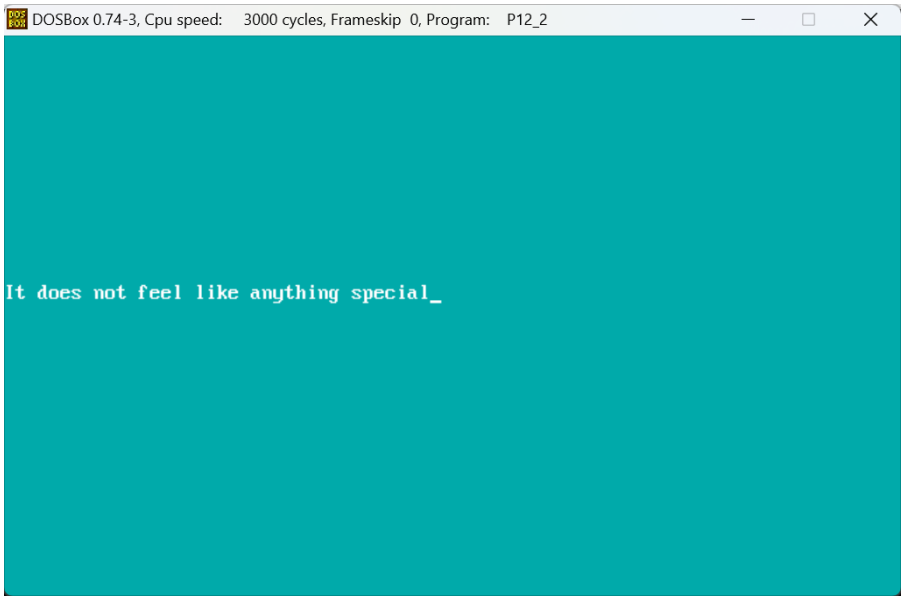
实现了如下功能：

① 显示一些提示信息，包括高音/中音/低音的 7 个音和键盘按键字符的对应关系，按什么键结束演奏等；

② 能模拟高、中、低音共 21 个音；

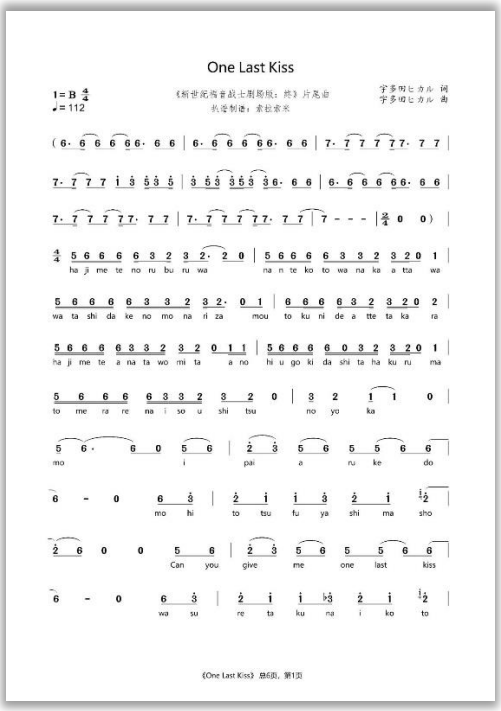
③ 弹奏时，发声可以固定时长，能模拟电子琴真实的发声，键按下发声，键松开不发声。

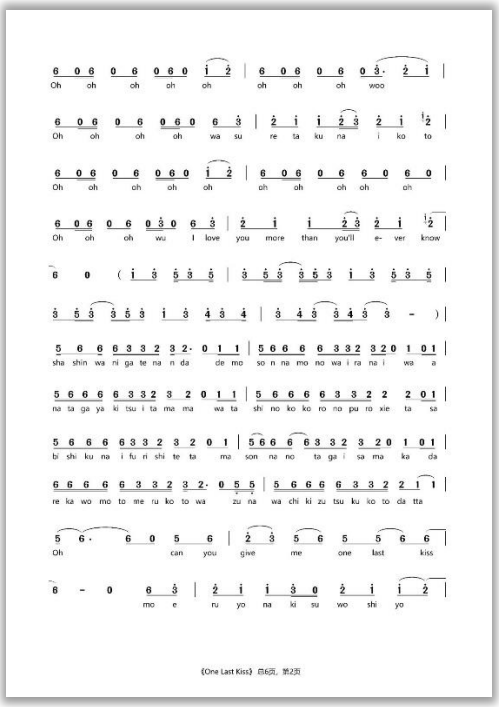
（3）附加任务 1+3：换一首有休止符的乐曲演奏。



实现了换一首有休止符的乐曲演奏，并且动态显示歌词的功能。

曲谱如下：





二、实验总结（实验中遇到的问题、解决方法和实验收获）

1．怎么实现按不同的键发不同的声音？

答：此处用了键盘扫描码，注意到键盘扫描码是连续的，则将键盘扫描码处理后变成 1-21 的顺序，一一对应发出的频率，但是要注意的是，频率定义为了字，但是键盘扫描码以字节为一单位，所以必须将键盘扫描码得到的顺序×2，才能对应上频率，不然就会出现隔一个键发出一个声音的问题。

2．怎么跳出中断，并且恢复原来的中断？

答：通过定义了一个单位 ISBREAK，并设置了一个跳出程序的按键“字母 O”，只要按下 O，程序就结束，实现的方法其实并不难，只要将读到的键盘扫描码与 O 的比对，如果是 O，则跳过一系列的发声，并将 ISBREAK 设置为 1，并且尤为重要的是，一定是在主程序中跳出循环，即主程序一直判断 ISBREAK 的值，当值变成 1，说明触发了跳出程序的按键，此时恢复中断向量，结束程序。

3．怎么判断哪些按键发声，哪些不发声？

答：在判断之后设置标志，同样在数据段中设置 ISPLAY，若 ISPLAY 为 1，则发出声音，ISPLAY 为 0，不发出声音。

4．怎么实现键按下发声，键松开不发声

答：按键在按下和释放的时候键盘扫描码不同，且释放的键盘扫描码最高位为 1，说明必大于 80H，则将键盘扫描码先与 80H 作比较，若大于 80H，则关闭喇叭，若需要发声，即键盘扫描码一直是发声的键所对应的，则扬声器打开，这样就实现一释放，扬声器就关闭。

三、回答思考题：

1. 若要求乐曲中有休止符，程序中应如何实现？

答：在程序中用 001 作为休止符标志，在 SING 子程序中加入判断语句，若读到 001，则程序只延时，但不发声。这样就可实现控制休止时间的休止符效果，具体实现代码也已在源程序中给出。

2. 如果要用键盘模拟电子琴演奏乐曲，请说明程序设计思路。

答：运用键盘扫描码，由之前的实验已经知道，键盘扫描码释放的时候，最高位为 1，但我设置的发出声音的键按下时键盘扫描码最高位都为 0，所以在读取到键盘扫描码之后，先与 80H 比较大小，若大于 80H，就在 SOUND 程序中设置关闭扬声器，而若需要发声，则设置扬声器打开。

这样一来，当符合要求的键盘扫描码被读到（即对应键按下时），扬声器发出声音，但按键释放，键盘扫描码判断大于 80，扬声器关闭，就实现了键按下发声，键松开不发声。

3. 如果完成了附加功能 1，请举例解释一下程序中频率数据(以及 8253 计数通道 2 的计数初值计算)、节拍数据与乐谱之间的对应关系。

答：当处理乐谱时，程序根据频率数据确定要播放的音符，根据节拍数据确定每个音符

的持续时间。这种映射关系通过指令加载频率和节拍数据，并调用 `SOUND` 子程序来实现。在 `SOUND` 子程序中，根据加载的频率数据，通过计算得到 8253 计数器的初值，控制扬声器发出对应频率的声音，同时根据节拍数据延时，实现对应持续时间的音符演奏。、

具体例子：频率和节拍对应乐谱，假设乐谱中的音符是 C4 和 A4，持续时间分别是四分音符和二分音符。

1）C4 音符：

频率：262 Hz

对应的频率数据：262

计数初值计算：1193180/262≈4556

对应节拍数据（四分音符）：假设四分音符对应 25（250ms）

2）A4 音符

频率：440 Hz

对应的频率数据：440

计数初值计算：1193180/440≈2714

对应节拍数据（二分音符）：假设二分音符对应 50（500ms）