华中科技大学计算机科学与技术学院 2021~2022 第二学期

“ 汇编语言程序设计 ”考试试卷（A 卷）

考试方式 闭卷 考试日期 2022-05-14 考试时长 150 分钟

专业班级 _____ 学 号 _____ 姓 名 _____

题号	一	二	三	四	五	六		总分	核对人
分值	20	20	20	10	10	20		100	
得分									

分 数	
评卷人	

一、 填空题（共 20 分，每空 1 分）

1、 CPU 在读取一个 32 位段程序的指令时要用到指令指示器 EIP，EIP 保存着_指令的地址_____。若 CPU 读取到的指令是转移指令，则能够转移到指明的目的地址处执行； 设 L1 为代码段中的一个近标号，则语句“JMP L1”实现跳转的实质操作是将__L1 的地址_送到 EIP 中。JMP 指令除了直接跳转外，还可以进行间接跳转。借助于寄存器 EBX，通过间接跳转实现与“JMP L1”相同的功能，需要执行的语句是：

MOV EBX, OFFSET L1

JMP EBX

除了使用跳转指令外，还可以使用其他方法实现跳转。例如，借助堆栈段和子程序的返回指令，可以实现与“JMP L1”相同的功能，此时，需要执行的语句是：

__PUSH OFFSET L1

RET

2、 标志寄存器用来保存指令执行后 CPU 所处的状态信息。OF 标志位表示____溢出_____，CF 标志位表示___进位或借位_____。在汇编语言中，计算机内的数值本身并不注明数值有无符号，而是通过选择合适的指令来表明操作数的符号性。例如，若将 AX 和 BX 中的操作数看作是无符号数，若希望 （AX） > （BX）时，跳转到 L1 处执行,则空格处应填写语句：

SUB AX, BX ; (1)

JA L1

.....

L1: .....

设(AX)=1234H, (BX)=5678h, 则执行上述语句 (1) 后：

(AX) = 0BBBCh

OF = 0

CF = 1

SF = 1

3、 在实模式下，使用软中断方式调用 45H 号中断处理程序，调用语句为 INT 45H 。该中断处理程序入口地址的段首址存放在 物理地址为 45H*4+2 处 ，入口地址的偏移地址存放在 物理地址为 45H*4 处 。 在实模式下调用软中断时，首先将 标志寄存器 入栈，然后将调用处的返回点地址信息入栈，最后跳转到中断处理程序入口地址处。中断处理程序的返回指令为 IRET 。外部设备用 I/O 指令来访问。从 34H 端口输入一个字节到 AL 中的指令语句为 IN AL，34H 。若在模块 A.ASM 里使用了在模块 B.ASM 中定义的字变量 wBuf，则在 A.ASM 中需要对此变量有相应的说明语句： extern wBuf : word 。

分 数	
评卷人	

二、 问答题（共 20 分）

设一个 32 位段程序中有如下程序片段:

```
.data
  STR1   DB   '1357'
  LEN    = $ - STR1
  STR2   DB   LEN DUP(0)
  A1     DD   STR1
  A2     DB   2 DUP ('5', 9)
.code
  main   proc   c
  ……                          ;省略号代表其他代码
  TO_ADDR   STR2,   A2   ;将 STR2 对应的地址复制到
                              ；A2 中。
  TO_VALUE   STR1,   A1，LEN   ；将 STR1 为首址，
             ；长度为 LEN 的变量内容，复制到 A1 中。
  ……
```

31H	00328000H STR1
33H	
35H	
37H	
00	00328004H STR2
00	
00	
00	
00H	00328008H A1
80H	
32H	
00H	
35H	0032800CH A2
09H	
35H	
09H	

（1）请在右表格中以字节为单位填写 data 段中各数据在存储器中的存放形式，并标明各变量所处的位置及偏移地址(STR1 的偏移地址为 00328000H，对齐方式为紧凑方式)。（10 分）

（2）写出上述程序中宏指令 TO_ADDR 和宏指令 TO_VALUE 的定义，要求宏指令执行完之后不破坏通用寄存器的内容。TO_ADDR 宏指令的功能是：将一个变量的偏移地址复制到另一个变量中。TO_VALUE 宏指令的功能是：将一个给定变量为首址和长度（字节数）的存储区里的内容，复制到另一个变量为首址的存储区中。（10 分）

```
TO_ADDR  MACRO  V, VP        ; 将 V 对应的地址复制到 变量 VP 中
         PUSH   EBX
         LEA    EBX, V
         MOV    DWORD PTR VP, EBX
         POP    EBX
         ENDM
```

```
; 将 SRC 为首址，长度为 LEN 的变量内容，复制到 DST 中。
TO_VALUE  MACRO  SRC,  DST，LEN
          LOCAL  TO_L, TO_END
          PUSHAD
          LEA  ESI,  SRC
          LEA  EDI,  DST
          MOV  ECX,  LEN
TO_L:     CMP  ECX,  0
          JZ    TO_END
          MOV  AL, [ESI]
          MOV  [EDI], AL
          INC   ESI
          INC   EDI
          DEC  ECX
          JMP   TO_L
TO_END:   POPAD
          ENDM
```

解答内容不得超过装订线

分　数	
评卷人	

三、　完善题（程序填空与改错，共 20 分，每处 1 分）

1）下面程序的功能是：找出 buf1 中绝对值最大的数，将其保存到字变量 m1 中。（每空 1 分，共 10 分）。

```
.....
.data
   buf1    dw 1, 3, -9, -14, -33
   n =    ($-buf1)/2
   m1    dw 0
.code
   main    proc    c
   mov    ecx,    n
   mov    esi,    offset buf1    或者  0
   mov    bx,    0
next:
   mov    ax,    [esi]        或者    buf1[esi*2]
   cmp    ax, 0
   jge      posi
   neg      ax
posi:
   cmp ax, bx
   jae    less
   mov bx, ax
   mov ax,    [esi]        或者    buf1[esi*2]
   mov m1, ax
less:
    add    esi, 2        或者    inc    esi
    loop    next
   invoke ExitProcess, 0
main endp
end
```

2）在一个以 0 结束的字符串中，将所有的小写字母转换为对应的大写字母，并将转换结果输出。请将程序中的语法错误和逻辑错误圈出来，并在其右侧写出正确的形式（请重点关注带*的行，每改正一行中的错误得 1 分，共 10 分）。

```
   .686P
   .model flat, c
   includelib    kernel32.lib
   includelib    libcmt.lib
   includelib    legacy_ stdio_definitions.lib
   printf          proto    :ptr sbyte, :vararg
   ExitProcess proto stdcall :dword
.data
```

```
            str1    db "abcDEFgh"        ; *      str1    db "abcDEFgh", 0
            fmt     db "%s" , 0
    .code
    main proc
            mov esi, str1                ; *      mov esi, offset str1
    next:
            mov al, [esi]
            cmp al, '0'                  ; *      cmp al,    0
            jae exit                     ; *      jz      exit
            cmp al, 'a'
            jb next                      ; *      jb    cont
            cmp [esi], 'z'               ; *    cmp byte ptr [esi], 'z'
            jg cont                      ; *      ja    cont
            add al, 'A'                  ; *      add    al, 'A'-'a'
            mov    str1, al              ; *      mov    [esi], al
    cont:
            inc esi
            jne next                     ; *    jmp    next
    exit:
            invoke printf, offset fmt, offset str1
            invoke ExitProcess, 0
        main endp
    end
```

解答内容不得超过装订线

分　数	
评卷人	

四、　　分析思考题（10 分）

阅读下面的程序，回答问题。

```
    .686P
    .model flat, c
    includelib    kernel32.lib
    includelib    libcmt.lib
    includelib    legacy_stdio_definitions.lib
    printf          proto    :ptr sbyte, :vararg
    ExitProcess proto stdcall :dword

    .data
      d1       dd 1234
      buf1     db 11 dup(0)
      fmt      db "%s", 0
    .code
    main proc
      push d1
```

```
    push offset buf1
    call f2to10
    add   esp, 8
    invoke printf, offset fmt, offset buf1
    invoke ExitProcess, 0
main endp
f2to10 proc
    push ebp
    mov ebp, esp
    mov esi, [ebp+8]
    mov eax, [ebp+12]
    mov ebx, 10
    mov ecx, 0
next1:                  ; …… ①
    mov edx, 0          ; …… ②
    div ebx
    push dx
    inc ecx             ; …… ③
    cmp eax, 0
    jne next1
next2:
    pop ax
    add al, '0'
    mov [esi], al
    inc esi
    loop next2
    pop ebp
    ret
f2to10 endp
end
```

1） 上述程序运行后，屏幕上显示的是什么？（2 分）

1234

2） 子程序 f2to10 的功能是什么？它的入口参数和出口参数分别是什么？(3 分)

将 一个无符号的双字类型数转换成 十进制形式，然后显示在屏幕上。

入口参数：存放转换结果的首地址；

待转换的双字类型的数

出口参数：无

3） 若语句②写在标号①之前，程序会出现异常，请说明原因？（3 分）

被除数为 （EDX，EAX），第 1 次除 10 后，（EDX）=4；（EAX）=123；

之后，（EDX，EAX）再除以 10，余数 为 3，又会放到 EDX 中，使得 （EAX）不能为 0. 不断循环，会导致 PUSH DX 出现访问异常。

4）若漏写了语句③，程序也会出现异常，请说明原因？（2 分）

漏写了语句③, ecx 会保持为 0. 在执行” next2：… loop next”之间的循环时，表面上会执行 100000000H 次，但 POP ax 或者 mov [esi], al 会出现异常，即访问单元的地址超出程序的地址空间。

分 数	
评卷人	

五、 分析优化题（共 10 分）。

如下的 C 语言程序段（32 位段）实现了统计一个字符串 str 中的大写字母的个数，并将其放入 count 中的功能。其编译后调试版本的汇编语言代码如下(注：斜体部分为 C 语句)。(10 分)

```
    int count = 0;
00E91793  mov     dword ptr [ebp-1Ch],0
    for (int i = 0; str[i] != '\0'; i++) {
00E9179A  mov     dword ptr [ebp-28h],0
00E917A1  jmp     00E917AC
00E917A3  mov     eax,dword ptr [ebp-28h]
00E917A6  add     eax,1
00E917A9  mov     dword ptr [ebp-28h],eax
00E917AC  mov     eax,dword ptr [ebp-28h]
00E917AF  movsx   ecx,byte ptr [ebp+eax-10h]
00E917B4  test    ecx,ecx
00E917B6  je      00E917DD
        if (str[i] >= 'A' && str[i] <= 'Z') {
00E917B8  mov     eax,dword ptr [ebp-28h]
00E917BB  movsx   ecx,byte ptr [ebp+eax-10h]
00E917C0  cmp     ecx,41h
00E917C3  jl      00E917DB
00E917C5  mov     eax,dword ptr [ebp-28h]
00E917C8  movsx   ecx,byte ptr [ebp+eax-10h]
00E917CD  cmp     ecx,5Ah
00E917D0  jg      00E917DB
            count++;
00E917D2  mov     eax,dword ptr [ebp-1Ch]
00E917D5  add     eax,1
00E917D8  mov     dword ptr [ebp-1Ch],eax
        }
    }
00E917DB  jmp     00E917A3
```

(1) 指出该段程序执行效率不高的原因 （2 分）。

多次出现访问相同内存单元的操作，属于重复操作，故效率不高。

例如，多次读取变量 i 的值，有 mov eax,dword ptr [ebp-28h] 多次重复执行。

解答内容不得超过装订线

取 str[i] 也是多次重复的出现。

(2) 改编相应的汇编语言程序，以提高程序的执行效率。要求写出变量与寄存器对应关系。(6 分)

将串的首地址 送入 EBX, 之后，通过 EBX 增 1，访问字符串的下一个单元；

每个字符 存放 在 AL 中， 即 str[i] 对应 AL； 也即 ([EBX]) → AL

```
        mov   count, 0
        lea   ebx, str
lp:     mov   al, [ebx]
        cmp   al, 0
        jz    exit
        cmp   al, 'A'
        jb    next
        cmp   al, 'Z'
        ja    next
        inc   count
next: inc     ebx
      jmp     lp
```

(3) “00E917C3 jl 00E917DB”处指令的机器码为 7CH 16H，解释 16H 代表的含义（2 分）

该指令的下一条指令地址为 00E917C5， 转移的目的地为 00E917DB，

00E917DB – 00E917C5 = 16H， 即代表了目标地址 与 当前 EIP 之间的偏移量。

分 数	
评卷人	

六、 设计题（20 分）

在 SCORES 为首址的字类型存储区中，存放了 N 个学生的汇编课程考试分数，分数为百分制整数。现需要对所有学生的该门课程成绩进行统计分类：

A: 90-100，B：80-89，C：70-79，D：60-69，E：0-59。

统计每档的学生人数，分别存放到 S_COUNT 为首址，长度为 5 的字类型存储区中。S_COUNT 中依次存放 A、B、C、D、E 档的人数。输出各个档次的人数。要求：

(1) 简要描述设计思想，给出寄存器分配方案。

(2) 用子程序 GRADE 判断某个分数所数的档次，描述其入口参数、出口参数。

(3) 画出主程序和子程序 GRADE 的流程图。

(4) 程序完整（包括堆栈段、数据段、代码段定义等），至少给出 4 条必要的注释。

```
.686P
.model flat, c
includelib  kernel32.lib
includelib  libcmt.lib
includelib  legacy_stdio_definitions.lib
printf      proto  :ptr sbyte, :vararg
ExitProcess proto stdcall :dword

.data
  SCORES  DW 95, 88,75, 90, 67, 45, 82
  N  = ($ - SCORES)/2
  S_COUNT  DW  5 DUP (0)
  FMT  DB  "档次 %c  人数 %d ", 0dH, 0aH, 0
  DANGCI  DB  'ABCDE'
.code
main proc  c
    MOV  ESI, 0  ; ESI 指明访问第几个学生的分数
    MOV  ECX, N  ; ECX 学生人数,  控制循环次数
L1:
    MOV   DX,  SCORES[ESI*2]
    CALL  GRADE    ; 入口参数：DX 待判别档次的分数
                   ; 出口参数 ：EAX 为 0,1,2,3,4 分别对应 A、B、C、D、E档
    INC    S_COUNT[EAX*2]
    INC   ESI
    LOOP  L1

    MOV  ECX, 0
L2:   ;  显示档次 和分数
    PUSH  ECX
    invoke printf, offset FMT, DANGCI[ECX],  S_COUNT[ECX*2]
    POP   ECX
    INC   ECX
    CMP  ECX, 5
    JNE   L2
    invoke ExitProcess, 0
main endp

; GRADE : 判别分数 DX 的档次
; 入口参数：DX 待判别档次的分数
; 出口参数 ：EAX 为 0,1,2,3,4 分别对应 A、B、C、D、E档
; 算法思想 ：EAX初值为0，若分数大于等于90，则可直接返回；
;            否则 EAX增1，然后判断是否为下一档；依此类推
GRADE  proc
```

```
  MOV  EAX ,0
  CMP  DX,90
  JAE   EXIT
  INC   EAX
  CMP  DX,80
  JAE   EXIT
  INC   EAX
  CMP  DX,70
  JAE   EXIT
  INC   EAX
  CMP  DX,60
  JAE   EXIT
  INC   EAX
EXIT :
  RET
GRADE  ENDP
END
```