汇编语言程序设计测试

学号 2353761　姓名 王晢晶

1.内存单元的基本单位是 字节 。

2. P和Q是用DW定义的变量，则下列指令正确的是 B 。
A. mov P, [1000h]　B. mov P, Q　C. mov 1000h, Q　D. mov Q, 1000h

3. 8-bit补码0c3表示的数是 -61 (十进制表示)，扩展成16-bit补码是 1111 1111 1100 0011 (十六进制表示)

4.下列二进制码均为有符号数编码，其中数值最大的数是 D 。
A.3C34H　B.8793H　C.9645H　D.5640H

5. 下列将AX清0的指令有 A,B,C,D 。(多选)
A. SUB AX,AX　B. XOR AX,AX　C. AND AX, 0　D. MOV AX, 0

6. 8086寄存器中，8位的寄存器共有 8 个, 其中8-bit计数器是 CL，DL 。

7. 在16-bit DOS环境下, 汇编模块中数据段名为data, 则对数据段的初始化操作应为 B 。
A. mov ax, data　B. mov ax, data　C. mov ax, data　D. mov ax, data
mov cs, ax　mov ds, ax　mov es, ax　mov ss, ax

8. 查80386指令手册关于BT指令说明如下:

则下列指令正确的有 B,C 。(多选)
A. BT DWORD PTR [EAX], [EBX]　B. BT DWORD PTR [EAX], EBX
C. BT DWORD PTR [EAX], 5　D. BT DWORD PTR [5], EBX

下图debug调试程序的一个截图,根据截图回答问题

9. 当前16-bit;累加器内容是 001E H, 将要执行的下一条指令是 MOV CX,AX ，栈顶位置 1B9EC H。

10. 此时的标志位CF= 0 (0/1)。

11. 指令“MOV [1234], DI” 中目的操作数的寻址方式是 直接寻址

12. 指令“JNZ 0011” 的操作码是 75 H, 位移量 -11 (十进制数)

12. 程序中执行时循环 30 次(十进制数)。

13. 这段程序功能是 计算1^2+2^2+3^2+.....+30^2 , 结果存放在 1234 H字单元中。

设数据定义如下:

```
DATA    SEGMENT
A       DB          2,3
B       DW          1234h,5678h
CC      DB          "12345"
D       DB          20 DUP(?)
E       EQU         $-A
DATA    ENDS
```

根据上述定义回答:
14. E的数值是多少？ 31D ；执行指令“MOV AX,WORD PTR A+1”后AX= 0302 H;

X、Y、Z为16-bit字单元，U为32-bit的双字单元,阅读下列程序段回答问题。

```
MOV AX, Z
IMUL X
```

```
ADD AX, Y
ADC DX, 0
MOV WORD PTR U, AX
MOV WORD PTR U+2, DX
```

15. 写出该程序段的计算表达式。 U=Z*X+Y

16. X内容为: 0EB14H, Y内容为: 0E841h, Z内容为:7FC8h,则计算结果为： 007A6689H ，存放在变量 U 中。

17. 下面的程序段是判断两个带符号字数据X和Y的大小，当X>Y时计算X-Y，当X<Y时计算Y-X，当X=Y时计算X+Y，运算后的结果存入字变量W中。在程序的空格处填写适当的指令。

```
        MOV     AX,X
        MOV     BX,Y
        CMP     AX,BX
         JG  L1
        JL      L2
        ADD     AX,BX
         MOV  W,AX
L1:      SUB   AX,BX
        JMP     L3
L2: SUB     BX,AX
 MOV     AX, BX
L3: MOV W,AX
```

有如下的C代码

```
short  x=3,y=5;
int max(int a, int b)
{
 int r;
 if(a>b) r=a; else r=b;
 return r;
}
void main()
{
 ...
 r=max(x,y)
 ...

}
```

18. 对应的汇编代码如下,请将代码补充完整

```
data    segment
x    dw    3
y    dw    4
data    ends
code    segment
...
max    proc
   push    bp
   mov    bp, sp
   mov    ax,  [BP+4]     ;取参数a
   mov    dx,  [BP+6]     ;取参数b
   cmp    ax, dx
   jg     loc
   mov    ax, dx
loc:
   mov    sp, bp
   pop    bp
    RET
max    endp
...
```

```
main:
 ...
 push    x
 push    y
 call max
  ADD SP,4
 mov    r, ax
 ...
code    ends
 end main
```

下列C代码关于两变量A和B内容交换的代码。

```
short a=3, b=4;

void swap(short *x, short y)
{
   short t=*x;
   *x=*y;
   *y=t;
}
...
void main()
{
   swap(&a,&b)
}
```

19. 下列A~B四组汇编代码中，可以与上面的C代码对应，完成相同的功能是 B ，简短说明理由 上述c代码实现的功能是将两个数值互换，而在B给出的汇编代码中，通过栈改变存储顺序，从而达到传递参数的目的，成功将两个数值互换了。 。

A组汇编代码

```
data    segment
a    dw   3
b    dw   4
data    ends

code    segment
swap   proc
      push    bp
      mov     bp, sp
      push    si
      push    di

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

      pop     di
      pop     si
      mov     sp, bp
      pop     bp
      ret
swap   endp
main:
     ...
     push   a
     push   b
     call   swap
     add    sp, 4
     ...
code    ends
      end main
```

B组汇编代码

```
data    segment
a    dw   3
b    dw   4
data    ends

code    segment
swap   proc
      push    bp
      mov     bp, sp

      mov     ax, [bp+4]
      mov     dx, [bp+6]
      mov     [bp+4], dx
      mov     [bp+6], ax

      mov     sp, bp
      pop     bp
      ret
swap   endp
main:
     ...
     push   a
     push   b
     call   swap
     add    sp, 4
     ...
code    ends
       end main
```

C组汇编代码

```
data    segment
a    dw   3
b    dw   4
data    ends

code    segment
swap   proc
      push    bp
      mov     bp, sp
      push    si
      push    di

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

      pop     di
      pop     si
      mov     sp, bp
      pop     bp
      ret
swap   endp
main:
     ...
     lea    ax, a
     push   ax
     lea    b
     push   ax
     call   swap
     add    sp, 4
     ...
code    ends
       end main
```

D组汇编代码

```
data    segment
a    dw   3
b    dw   4
data    ends

code    segment
swap   proc
      push    bp
      mov     bp, sp

      mov     ax, [bp+4]
      mov     dx, [bp+6]
      mov     [bp+4], dx
      mov     [bp+6], ax

      mov     sp, bp
      pop     bp
      ret
swap   endp
main:
     ...
     lea    ax, a
     push   ax
     lea    b
     push   ax
     call   swap
     add    sp, 4
     ...
code    ends
       end main
```

20.编写以十进制形式从键盘输入一个不超过655535(即在16-bit范围内无符号整数)的子程序 read16。

```
data segment
x dw
data segment

code segment
...
display16  proc
 push bp
 mov bp,sp

 mov ax, [bp+4]
 mov cx, 0
 mov bx, 10

Rep1b:
 MOV DX, 0
 DIV BX
 PUSH DX
 INC CX
 OR AX, AX
 JNZ Rep1b

Rep2b:
 POP DX
 ADD DL, 48
 MOV AH, 2
 INT 21h
 LOOP Rep2b

 mov sp, bp
 pop bp
 ret
display16 endp
...
read16 proc
 ;//写出实现代码, 约定I输入的数通过AX传递出去
```

```
    mov bx, 0
    mov ah, 1  ; 功能号 1 用于从键盘输入单个字符

input_loop:
    int 21h    ; 调用中断获取字符

    cmp al, '0'
    jl input_loop  ; 如果输入小于 '0'，重新输入

    cmp al, '9'
    jg input_loop  ; 如果输入大于 '9'，重新输入

    sub al, '0'  ; 将字符转换为数字
    mov cx, 10
    mul cx      ; 乘以 10
    add bx, ax  ; 累加到 bx 中
    jmp input_loop

done:
    mov ax, bx  ; 将结果传递到 ax
    ret
```

```
read16 endp

main:
 ...
 call read16 ;输入一个整数
 mov x, ax

 push x
 call display16 ;验证输入是否正确
 add sp, 2
 ....
code ends
    end main
```

21. 编写程序，将数组A中的奇数和偶数分别存放到A1和A2中，然后以十进制形式显示出A1和A2中所有的数。
给出的数据样例如下，调试时可以直接使用。

```
DATA SEGMENT
A   DW      1234H,5678H,7D4CH,0D7H,0,1,7D2AH,6A0EH,10F5H,645DH
N    EQU     ($-A)/2
A1 DW N DUP{0}
A2 DW N DUP{0}
DATA ENDS
```

在以下编写完整的代码,包括数据段,代码段

```
DATA SEGMENT
A   DW      1234H,5678H,7D4CH,0D7H,0,1,7D2AH,6A0EH,10F5H,645DH
N    EQU     ($-A)/2
A1 DW N,0,n DUP(0)
A2 DW N,0,n DUP(0)
MSG1   DB     "THE ODD NUMBER IS:",13,10,"$"
MSG2   DB      13,10,"THE EVEN NUMBER IS:",13,10,"$"
DATA ENDS

CODE  SEGMENT
     ASSUME  CS:CODE,DS:DATA
START:  MOV     AX,DATA
 MOV     DS,AX

      mov si, 0
    mov di, 0
    mov Dx, 0

    mov cx, N
```

```
sort_loop:
    mov ax, A[si]
    test ax,1H                  ;判断是否为奇数
    jz even_number              ;如果为0，说明是偶数

    LEA BX,A1
   mov [BX], ax
    inc Dx                      ;记录奇数个数
    jmp next_number

even_number:
 lea BX,A2
    mov  [BX], ax
    inc di
    add si,2
 loop  sort_loop

next_number:
 add si,2
    loop sort_loop

    mov cx, Dx                  ;奇数个数
    mov si, 0

print_odd_numbers:
 mov dx,offset MSG1
 mov ah,09h
 int 21h

_odd:
 MOV SI,0
    mov ax, A1[si]
    call print_number

    add si, 2
    loop _odd

    mov cx, di                  ;偶数个数
    mov si, 0

print_even_numbers:
 mov dx,offset MSG2
 mov ah,09h
 int 21h

_even:
 mov si,0
    mov ax, A2[si]
    call print_number
    add si, 2
    loop _even


print_number proc
    mov bx, 10
 mov cx,0
loc:
    mov dx, 0
    div bx

    push dx
    inc cx
 cmp ax,0
 jnz loc

loc2:
    pop dx
    add dl, '0'
```

```
    mov ah, 2
    int 21h
    loop loc2
print_number endp

    mov ah, 4ch
    int 21h

code ends
        end start
```