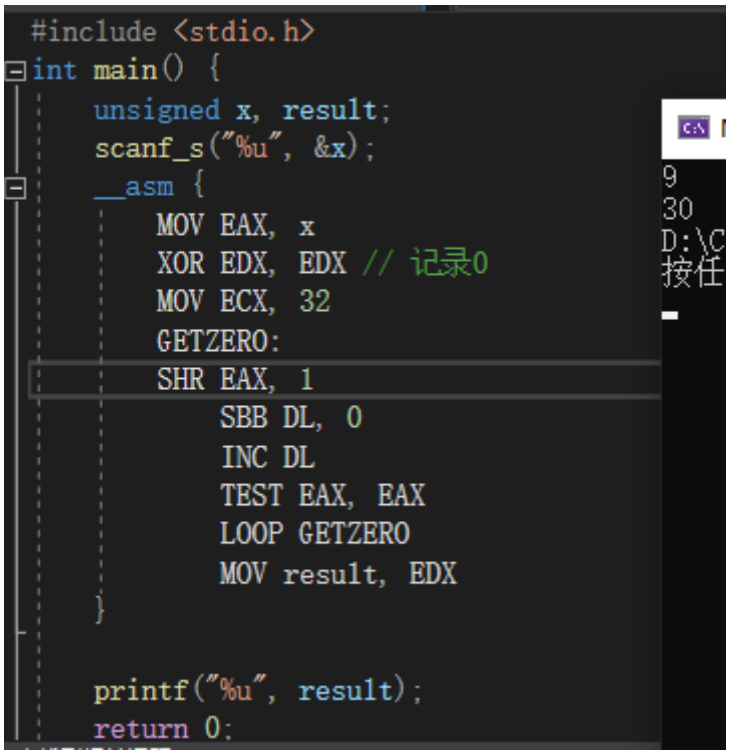
35. 由用户输入一个无符号整数(整型)；统计该32位整数中位值为0的个数；显示输出统计结果

```
#include <stdio.h>
int main() {
    unsigned x, result;
    scanf_s("%u", &x);
    __asm {
        MOV EAX, x
        XOR EDX, EDX // 记录0
        MOV ECX, 32
        GETZERO:
            SHR EAX, 1
            SBB DL, 0
            INC DL
            TEST EAX, EAX
            LOOP GETZERO
        MOV result, EDX
    }

    printf("%u", result);
    return 0;
}
```

结果：



38. 由用户从键盘输入一个字符串；分别统计字符串中英文字母、十进制数字字符和其他符号的个数；显示输出统计结果

```
#include <stdio.h>
int main() {
```

```
    unsigned letters, nums, others = 0;
    char str[1000];
    gets_s(str);
    __asm {
        LEA EDI, str
        CMP BYTE PTR [EDI], 0
        JE OVER
        XOR EBX, EBX // letter
        XOR EDX, EDX // nums
        XOR ECX, ECX // others
        ALOOP:
            MOV AL, BYTE PTR[EDI]
            SUB AL, 48 // 假设是0
            CMP AL, 9
            JA SHORT LETTER
            INC EDX
            JMP NEXT

        LETTER:
            SUB AL, 17 // 假设是A
            CMP AL, 25
            JA SHORT LETTER1
            INC EBX
            JMP NEXT
        LETTER1:
            SUB AL, 32 // 假设是a
            CMP AL, 25
            JA OTHER
            INC EBX
            JMP NEXT

        OTHER:
            INC ECX

        NEXT:
            INC EDI
            CMP BYTE PTR[EDI], 0
            JNE ALOOP

        OVER:
            MOV letters, EBX
            MOV nums, EDX
            MOV others, ECX
    }

    printf("字母数：%u，十进制数字符数：%u，其他符号数：%u", letters, nums, others);
    return 0;
}
```

结果：

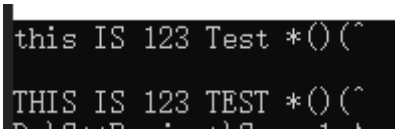
41.由用户从键盘输人一个字符串；将所有可能的小写字母转换为对应的大写字母；最后显示输出字符串。请采用子程序实现把可能的小写字母转换为大写字母。

```
#include <stdio.h>
int main() {
    char str[1000];
    gets_s(str);
    __asm {
        LEA EDX, str
        XOR ECX, ECX
        ALOOP:
            MOV EAX,[EDX]
            CALL LOWTOUP
            INC EDX
            CMP BYTE PTR[EDX], 0
            LOOPNE ALOOP
    }


    printf("\n%s", str);
    return 0;

    __asm {
    LOWTOUP:
        MOV AL, BYTE PTR [EDX]
        CMP AL, 0
        JE OVER
        SUB AL, 97 // 假设是a
        CMP AL, 25
        JA OVER
        ADD AL, 65
        MOV BYTE PTR[EDX], AL
    OVER:
        RET
    }
}
```

结果：



44.由用户输入一个无符号十进制整数（整型）；将该整数转换为对应十六进制数输出。要求输出时，只能采用字符串格式（先转换为对应的十六进制数字字符串）

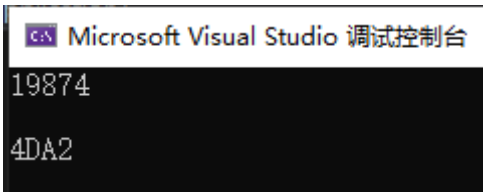
```
#include <stdio.h>
char str[1000];
int main() {
    unsigned x;
    scanf_s("%u", &x);
    __asm {
        PUSH -1
```

```
        MOV EDI, x
        LEA EDX, str
        ALOOP:
            MOV EAX, EDI
            AND AL, 0x0f
            ADD AL, '0'
            CMP AL, '9'
            JLE NEXT
            ADD AL, 7
        NEXT:
            PUSH EAX
            SHR EDI, 4
            TEST EDI, EDI
            JE DEAL
            JMP ALOOP
        DEAL:
            POP EAX
            CMP EAX, -1
            JE OVER
            MOV BYTE PTR[EDX], AL
            INC EDX
            JMP DEAL
        OVER:
    }

    printf("\n%s\n", str);
    return 0;

}
```

结果：



47.由用户从键盘输入一个十进制数字符串（假设不含其他字符）；然后把该十进制数字符串转换成对应的数值；接着把该数值转换成对应的十六进制数字符串；最后输出十六进制数字符串。请采用子程序实现把十进制数字符串转换为对应的数值。

```
#include <stdio.h>
unsigned decimalNum;
char decimalStr[10], hexadecimalStr[10];
int main() {
    scanf_s("%s", decimalStr, 10);
    __asm {
        LEA EDX, decimalStr
        MOV ECX, 10
        CALL TONUM
        MOV decimalNum, EAX
        MOV ECX, EAX
        LEA EDX, hexadecimalStr
        CALL TOHEXSTR
    }
```

```
    printf("十进制数值：%u\n", decimalNum);
    printf("十六进制字符串：%s\n", hexadecimalStr);
    return 0;


    __asm {
    TONUM:
        XOR EAX, EAX
        PUSH EBX
        PUSH EDI
        XOR EBX, EBX
        MOV EDI, EDX
        ALOOP:
            MOV BL, BYTE PTR[EDI]
            SUB BL, '0'
            MUL ECX
            ADD EAX, EBX
            INC EDI
            CMP BYTE PTR[EDI], 0
            JNE ALOOP
            POP EDI
            POP EBX
            RET
    }

    __asm {
    TOHEXSTR:
        PUSH -1
        BLOOP:
            MOV EAX, ECX
            AND AL, 0x0f
            ADD AL, '0'
            CMP AL, '9'
            JLE NEXT
            ADD AL, 7
        NEXT:
            PUSH EAX
            SHR ECX, 4
            TEST ECX, ECX
            JE DEAL
            JMP BLOOP
        DEAL:
            POP EAX
            CMP EAX, -1
            JE OVER
            MOV BYTE PTR[EDX], AL
            INC EDX
            JMP DEAL
        OVER :
            RET
    }
}
```

结果：

50.由用户从键盘先后输入两个自然数；然后分别计算这两个数的和、差、积；分别显示输出结果。请采用合适的子程序。要求在输入和输出时，都只能采用字符串格式。

```
#include <stdio.h>
char sum[10], difference[10], product[10];
char a[10], b[10];

int main() {
    scanf_s("%s", a, 10);
    scanf_s("%s", b, 10);
    __asm {
        LEA EDX, b
        MOV ECX, 10
        CALL STRTONUM
        PUSH EAX
        LEA EDX, a
        CALL STRTONUM
        PUSH EAX

        MOV EDI, [ESP + 4] // b
        MOV EAX, [ESP] // a
        MUL EDI
        LEA ECX, product
        CALL NUMTOSTR

        MOV EAX, [ESP] // a
        ADD EAX, [ESP + 4]
        LEA ECX, sum
        CALL NUMTOSTR

        MOV EAX, [ESP] // a
        SUB EAX, [ESP + 4]
        LEA ECX, difference
        CALL NUMTOSTR
        POP EAX
        POP EAX
    }
    printf("a = %s, b = %s\n", a, b);

    printf("a + b = %s\n", sum);
    printf("a - b = %s\n", difference);
    printf("a * b = %s\n", product);
    return 0;
    __asm {
    STRTONUM:
        XOR EAX, EAX
        PUSH EBX
        PUSH EDI
        XOR EBX, EBX
        MOV EDI, EDX
```

```
        ALOOP :
            MOV BL, BYTE PTR[EDI]
            SUB BL, '0'
            MUL ECX
            ADD EAX, EBX
            INC EDI
            CMP BYTE PTR[EDI], 0
            JNE ALOOP
            POP EDI
            POP EBX
            RET
    }

    __asm {
    NUMTOSTR:
        PUSH EBX
        MOV EBX, 10
        PUSH - 1
        CMP EAX, 0
        JGE BLOOP
        MOV BYTE PTR[ECX], '-'
        INC ECX
        NEG EAX
        BLOOP:
            XOR EDX, EDX
            DIV EBX
            ADD DL, '0'
            PUSH EDX
            TEST EAX, EAX
            JNE BLOOP
        DEAL :
            POP EAX
            CMP EAX, -1
            JE OVER
            MOV BYTE PTR[ECX], AL
            INC ECX
            JMP DEAL
        OVER :
            POP EBX
            RET
    }

}
```

结果：