**作业D** 等级：\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

学号：\_\_\_1927405135\_\_\_\_ 姓名:\_\_\_尤王杰\_\_\_\_

[F1]IA-32系列CPU常用的寻址方式有\_\_立即寻址\_\_、\_寄存器寻址\_和\_存储器寻址\_三大类，存储器寻址方式又可分为\_直接寻址\_、\_\_寄存器间接寻址\_\_、\_寄存器相对寻址\_\_、\_\_基址加变址\_\_、\_\_通用寻址\_\_。

[F2]指令“MOV AL,[EBX+4\*ECX+1234H]”的源操作数寻址方式是\_\_\_通用\_\_，目的操作数寻址方式是\_\_\_寄存器选址\_\_\_。

[F3]指令“MOV [EBX+ESI],12345678H”的源操作数寻址方式是\_\_\_立即寻址\_\_\_，目的操作数寻址方式是\_\_\_基址加变址\_\_\_。

[F4]假设双字存储单元5678H的内容是9ABCH，又假设寄存器ECX的内容是1234H，寄存器EDI的内容是1111H。在执行指令“MOV AX,[ECX+4\*EDI+1]”后，寄存器AL内容是\_\_\_B\_\_\_，AH内容是\_\_\_\_A\_\_\_。

[F5]如果只用一条指令实现把寄存器EBX和寄存器EDX的值相加，同时减去123，并把结果送到寄存器ECX，那么这条指令是\_\_\_\_LEA ECX,[ EBX + EDX – 123 ]\_\_\_。

[F6]假设寄存器ECX内含有一个较小无符号数，如果只用一条指令实现把寄存器ECX的值乘上5，同时加上78，那么这条指令是\_\_\_LEA EXC,[EXC + 4\*EXC + 78]\_\_\_。

[S1]以下关于寻址方式的叙述，错误的是（多选）？（ BD ）

（A）目的操作数不能采用立即寻址方式。

（B）通常情况下，源操作数和目的操作数不能同时是寄存器。

（C）通常情况下，源操作数和目的操作数不能同时是存储器单元。

（D）在一条指令中，不能同时出现寄存器寻址方式和存储器寻址方式。

[S2]以下关于寻址方式的叙述，正确的是（多选）？( AC )

（A）寻址方式指给出操作数的方法。

（B）寄存器寻址方式的效率最高。

（C）立即寻址方式的效率最高。

（D）存储器寻址方式的效率最高。

[S3]以下指令中，正确的是（多选）？ （B,E,F,H,I,）

（A）MOV [ESI], [EDI] （B）SBB EBX, [EBX]

（C）MOV 300, AX （D）ADD DL, [EDX+5\*ECX-100]

（E）MOV word ptr [EDX], 2\*3 （F）LEA AL, [2\*ESI]

（G）MOV SI, [BL] （H）XCHG AL, [EBX+4\*ECX]

（I）MOV DI, [1234H] （J）SUB ESI, [2\*ECX+4\*EDX]

[R1]如下指令中目的操作数采用寄存器寻址方式，请以注释形式说明源操作数的寻址方式：

MOV EBX, [1234H] ;#直接寻址

MOV EBX, 1234H ;#立即数寻址

MOV EDX, EBX ;#寄存器寻址

MOV EDX, [EBX] ;#寄存器间接寻址

MOV EDX, [EBX+1234H] ;#通用寻址

[R2]设寄存器ECX的内容是4，寄存器ESI的内容是1230H。请以注释形式给出如下每条指令中存储器操作数的有效地址：

MOV AL, [ESI-5] ;EA=122BH

MOV AX, [ESI+ECX\*4] ;EA=1240H

MOV EAX, [ESI+ECX\*8+100H] ;EA=1350H

MOV [ESI\*8], AL ;EA=9180H

MOV [2000H+ECX], ECX ;EA=2004H

[R3]设寄存器ECX的内容是100H，寄存器EDX的内容是1234H。请给出执行如下每条指令后，寄存器EAX的值：

LEA EAX, [EDX+3] ;EAX=00001237H

LEA EAX, [ECX\*8] ;EAX=00000800H

LEA EAX, [ECX+ECX\*4] ;EAX=00000500H

LEA EAX, [ECX+EDX\*2-5] ;EAX=00002563H

LEA EAX, [5678H] ;EAX=00005678H

MOV EAX, [5678H] ;EAX=00005678H

[R4]假设VC2010环境，请给出如下控制台程序的运行结果：

#include <stdio.h>

int varx = 6;

char dstr[] = "abcde";

int buff[5]= {1,2,3,4,5};

int main()

{

int var\_a, var\_b, var\_c;

\_asm {

MOV EAX, varx

ADD EAX, 4

MOV var\_a, EAX

;

LEA EBX, dstr

MOV AL, [EBX+2]

MOV var\_b, EAX

;

LEA ESI, buff

MOV ECX, 1

MOV AX, [ESI+ECX\*2+5]

MOV var\_c, EAX

}

printf("A=%d,B=%d,C=%d\n", var\_a, var\_b, var\_c);

return 0;

}

运行结果: A=10,B=99,C=768