**作业E** 等级：\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

学号：\_1929409021\_\_\_ 姓名:\_\_\_胡志彬\_\_

[F1]在汇编语言中，设计子程序时，常用传递参数的方法是寄存器传递\_和\_\_堆栈传递\_。

[F2]在汇编语言中，设计子程序时，经常把局部变量安排在\_寄存器\_和\_堆栈帧\_\_。

[F3]主程序与子程序的约定包括参数传递方式\_、\_ 返回值约定\_\_和\_寄存器保存规则\_\_等。

[F4]按照转移到子程序时是否设置CS，调用指令可以分为\_段内调用\_和\_段间调用\_两类。

[F5]按照表示子程序入口地址的方法，调用指令可以分为\_直接调用\_和\_间接调用两类。

[F6]按照返回主程序时是否设置CS，返回指令分为\_段内返回\_和\_段间返回两类。

[F7]如果利用堆栈传递参数，可以采用\_调用者平衡或\_被调用者平衡两种平衡堆栈的方法。

[S1]以下关于调用和返回指令的叙述，正确的是？A

（A）通常子程序的返回指令应该与主程序的调用指令配对使用。

（B）直接调用，应当直接返回；间接调用应当间接返回。

（C）直接调用，应当采用不带立即数的返回指令。

（D）间接调用，应当采用带立即数的返回指令。

[S2]以下关于汇编语言设计子程序的叙述，正确的是？AB

（A）需要考虑传递参数的方法。

（B）需要考虑局部变量的安排。

（C）无需考虑堆栈的平衡。

（D）无需考虑返回的方式。

[S3]以下关于设计子程序的叙述，错误的是？D

（A）为了空间效率，应当把在程序中反复出现的某个程序片段设计成子程序。

（B）为了开发效率，应当把具有通用性的程序片段，设计成子程序。

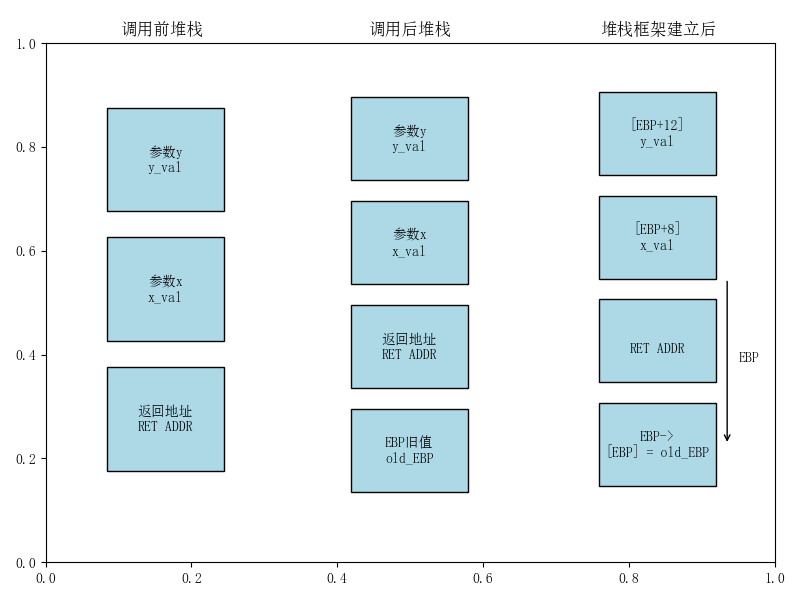
（C）为了模块化，应当把具有相对独立功能的程序，设计成子程序。

（D）为了开发效率，不应当编写子程序说明。

[Q1]请简要说明保持堆栈平衡的重要性。

防止堆栈溢出/错位，确保程序正确返回

[Q2]请画出3.5节的例11中调用函数cf341期间堆栈的变化示意图。



[R1]采用速度最大化的编译选项，编译生成如下函数的目标代码，请说明每条汇编格式指令的作用（可以利用高级语言语句或表达式来描述）。

char \* my(char \*dst, char value, unsigned int count)

{

char \*start = dst;

while (count--)

\*dst++ = value;

return(start);

}

MOV start,dst ; 保存起始地址

TEST count,count ; 检查计数器

JZ end ; 计数器为0则跳转

MOV [dst],value ; 存储值

INC dst ; 指针递增

DEC count ; 计数器递减

JNZ loop ; 循环

[R2]基于VC2010环境，设有如下源程序：

#include <stdio.h>

int varx = 6;

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

int main()

{

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

\_asm {

MOV EAX, 6

CALL SUBR1

MOV var\_a, EAX

;

PUSH DWORD PTR 5

LEA EAX, buff

PUSH EAX

CALL SUBR2

ADD ESP, 8

MOV var\_b, EAX

;

LEA EAX, var\_c

PUSH EAX

MOV EAX, varx

PUSH EAX

CALL SUBR3

}

printf("D=%d,E=%d,F=%d\n", var\_a, var\_b, var\_c);

return 0;

\_asm {

SUBR1:

ADD EAX, 7

RET

;

SUBR2:

PUSH EBP

MOV EBP, ESP

MOV ECX, [EBP+12]

MOV EDX, [EBP+8]

XOR EAX, EAX

SUBR2A:

ADD EAX, [EDX]

ADD EDX, 4

LOOP SUBR2A

POP EBP

RET

;

SUBR3:

PUSH EBP

MOV EBP, ESP

MOV EAX, [EBP+8]

MOV ECX, [EBP+12]

ADD EAX, 8

MOV [ECX], EAX

POP EBP

RET 8

}

}

（1）请说明三个子程序传递入口参数的方法。

（2）请说明三个子程序的功能。

（3）请说明子程序SUB2和SUB3平衡堆栈的方法。

（4）假设不上机编辑、运行，请给出这个程序运行的结果。

(1)

SUBR1: 寄存器(EAX)传递

SUBR2: 堆栈传递(数组指针和长度)

SUBR3: 堆栈传递(值和指针)

(2)

SUBR1: 对输入值加7

SUBR2: 计算数组元素和

SUBR3: 对输入值加8并存储

(3)

SUB2: 调用者平衡(ADD ESP,8)

SUB3: 被调用者平衡(RET 8)

(4) 运行结果：D=13,E=15,F=14