Arquitectura de computadores /Interfaces y Arquitectura Hardware
SEGUNDO EXÁMEN PARCIAL

Nombre	viernes 28 octubre 2022
Código:	Duración: 1.5 horas

Máximo común divisor

En las matemáticas, se define el máximo común divisor (abreviado MCD) de dos o más números enteros al mayor número entero que los divide sin dejar residuo alguno. EL método más eficiente para calcular el MCD es el algoritmo de Euclides, que utiliza la operación de la división junto al hecho que el MCD de dos números también divide al residuo obtenido de dividir el mayor entre el más pequeño.

En la figura 1 se muestra un programa en lenguaje alto nivel que implementa el algoritmo de Euclides en dos funciones, ambas funciones (una de forma iterativa la otra de forma recursiva) permiten calcular el máximo común divisor entre dos números de datos tipo int (enteros de 32 bits). El programa en el main() hace un llamado de la función ***MCD_iterativo***(120,25); - que devuelve 5 como el MCD en éste caso -.

Se realiza la división entera entre ellos, de donde se obtienen los naturales –Cociente y residuo, entonces, si se divide 120 entre 25 da un cociente de 4 y un residuo de 20, cómo el residuo no es cero, se divide ahora el anterior divisor 25 entre el anterior residuo 20, dando ahora 1 y residuo 5. Después se divide 20 entre 5 dando un residuo de 0, lo que significa que 5 es el máximo común divisor.

```
static int temporal;    //Para no perder b
static int mcd_result;

int main(void) {
    mcd_result = MCD_iterativo(120,25);
    return 0;
}

int MCD_iterativo(int a, int b) {
    while (b != 0) {
        temporal = b;
        b = a % b;
        a = temporal;
    }
    return a;
}

int MCD_recursivo(int a, int b) {
    if (b == 0) return a;
    return MCD_recursivo(b, a % b);
}
```

Figura 1

Se pide entonces lo siguiente:

A. [05%] Escriba el segmento de datos (*.data)* del programa ensamblador x86 MASM equivalente a la declaración de las variables globales del programa alto nivel.
B. [30%]Escriba en lenguaje ensamblador x86 MASM una subrutina que permita implementar la función ***MCD_iterativo()*** usando la pila como mecanismo de paso de los dos parámetros de entrada y use el

registro EAX como parámetro de salida. Optimice su código con la intención que se obtenga la mínima cantidad de instrucciones en ensamblador.

C. [20%] Escriba el programa ensamblador que corresponda al segmento de código (.code) para cada la función ***main()*** del programa C++ de la figura1. Optimice su código con la intención que se obtenga la mínima cantidad de instrucciones en ensamblador (20 puntos).

D. [45%] En la figura 2 se muestra la implementación de la función `int` ***`MCD_recursivo`***`(int a, int b)` en lenguaje ensamblador MASM x86

```
        MCD_recursivo PROC
00F51068        push    ebp
00F51069        push    edx
00F5106A        mov     ebp, esp
00F5106C        cmp     DWORD PTR [ebp + _b$], 0
00F51070        jne     LN2
00F51072        mov     eax, DWORD PTR [ebp + _a$]
00F51075        jmp     LN1
00F51077 LN2:
00F5107A        mov     eax, DWORD PTR [ebp + _a$]
00F5107C        xor     edx,edx
00F5107F        idiv    DWORD PTR [ebp + _b$]
00F51080        push    edx
00F51083        mov     eax, DWORD PTR [ebp + _b$]
00F51084        push    eax
00F51089        call    MCD_recursivo
00F5108C        add     esp, 8
00F5108D LN1:
00F5108E        pop     edx
                pop     ebp
                ret     0
        MCD_recursivo ENDP
```

Figura 2

Con ayuda de la **tabla 1**, realice un análisis del comportamiento y uso de la pila al ejecutar completamente el programa, asuma entonces que el inicio de la función main en ensamblador es el siguiente código:

```
main   PROC
00F51024  push        ebp
00F51025  mov         ebp,esp
00F51027  push        25
00F51029  push        120
00F5102B  call        MCD_recursivo
00F51030  .
          .
          .

          xor     eax, eax
          pop     ebp
          ret
```

Y asuma también que los registros al inicio de la ejecución tienen almacenado lo siguiente:

EAX = 00F3F998 **EBX** = 010A6000 **ECX** = 00F5100F **EDX** = 00F5100F **ESI** = 00F5100F **EDI** = 00F5100F
EIP = 00F51024 **ESP** = 00F3F940 **EBP** = 00F3F94C **EFL** = 00000246

Indique los valores de las constantes **_b$ =** _____ y **_a$** =_____ para que el programa funcione correctamente y complete el programa main.

Stack					Instrucciones que escriben	Instrucciones que leen
0x00F3F8CC						
0x00F3F8D0						
0x00F3F8D4						
0x00F3F8D8						
0x00F3F8DC						
0x00F3F8E0						
0x00F3F8E4						
0x00F3F8E8						
0x00F3F8EC						
0x00F3F8F0						
0x00F3F8F4						
0x00F3F8F8						
0x00F3F8FC						
0x00F3F900						
0x00F3F904						
0x00F3F908						
0x00F3F90C						
0x00F3F910						
0x00F3F914						
0x00F3F918						
0x00F3F91C						
0x00F3F920						
0x00F3F924						
0x00F3F928						
0x00F3F92C						
0x00F3F930						
0x00F3F934						
0x00F3F938						
0x00F3F93C						
0x00F3F940						
0x00F3F944						
0x00F3F948						
0x00F3F94C						
0x00F3F950						