

Ejercicio 1:

El valor de la variable FACT_DATO_1 es el valor del factorial de DATO_1.

```

1 ; *****
2 ; CALCULA EL PRODUCTO DEL FA
3 ; ENCUENTRAN EN LAS POSICION
4 ; DATOS. EL VALOR DE CADA NU
5 ; SE ALMACENA EN DOS PALABRA
6 ; PALABRA EL MENOS SIGNIFICA
7 ; SIGNIFICATIVO. SE UTILIZA
8 ; *****
9
10 ; DEFINICION DEL SEGMENTO DE
11
12 DATOS SEGMENT
13
14 DATO_1 DB 4
15 DATO_2 DB 5
16
17 DATOS ENDS
18
19 ; DEFINICION DEL SEGMENTO DE PILA
20

```

DOSBox 0.70, Cpu Cycles: 3000, Frameskip: 0, Program: TD

File View Run Breakpoints Data Options Window Help

[-] CPU 80486

Address	Instruction	Comment	Register	Value
4810:001B	MOV FACT_DATO_1, AX	ALMACENA EL R	ax	012C
4810:001F	MOV CL, DATO_2		bx	9C3C
4810:0023	CALL FACTOR		cx	1712
4810:0026	MOV BX, FACT_DATO_1		dx	013C
4810:002B	MUL BX ; EN AX ESTA EL RESULTADO DE		si	EE3E
4810:002D	MOV RESULT, AX		di	0005
4810:0031	MOV RESULT+2, DX		bp	0100
4810:0036	MOV AX, 4C00H		sp	0106
4810:0039	INT 21H		ds	1A01
4810:003B	MOV AX, 1		es	EED4
4810:003E	XOR CH, CH		ss	012C
4810:0040	CMP CX, 0		cs	0000
4810:0043	JE FIN		ip	0000

F1-Help F2-Bkpt F3-Mod F4-Here F5-Zoom F6-Next F7-Trace F8-Step F9-Run F10-Menu

4! * 5! = 2880 = 0B40h, resultado correcto.

FACT_DATO_1 = 0018h, que es el factorial de 4.

Escribo en DATO_1 el número 4 y en DATO_2 el número 5.

```

1 ; FIN DE LAS INICIALIZACIONES
2
3 ; COMIENZO DEL PROGRAMA
4 ; MOV CL, DATO_1
5 ; CALL FACTOR
6 ; MOV FACT_DATO_1, AX
7 ; MOV CL, DATO_2
8 ; CALL FACTOR
9 ; MOV BX, FACT_DATO_1
10 ; MUL BX
11
12 MOV CL, DATO_1
13 CALL FACTOR
14 MOV FACT_DATO_1, AX
15
16 ; ALMACENA EL RESULTADO
17 MOV RESULT, AX
18 MOV RESULT+2, DX
19

```

DOSBox 0.70, Cpu Cycles: 3000, Frameskip: 0, Program: TD

File View Run Breakpoints Data Options Window Help

[-] CPU 80486

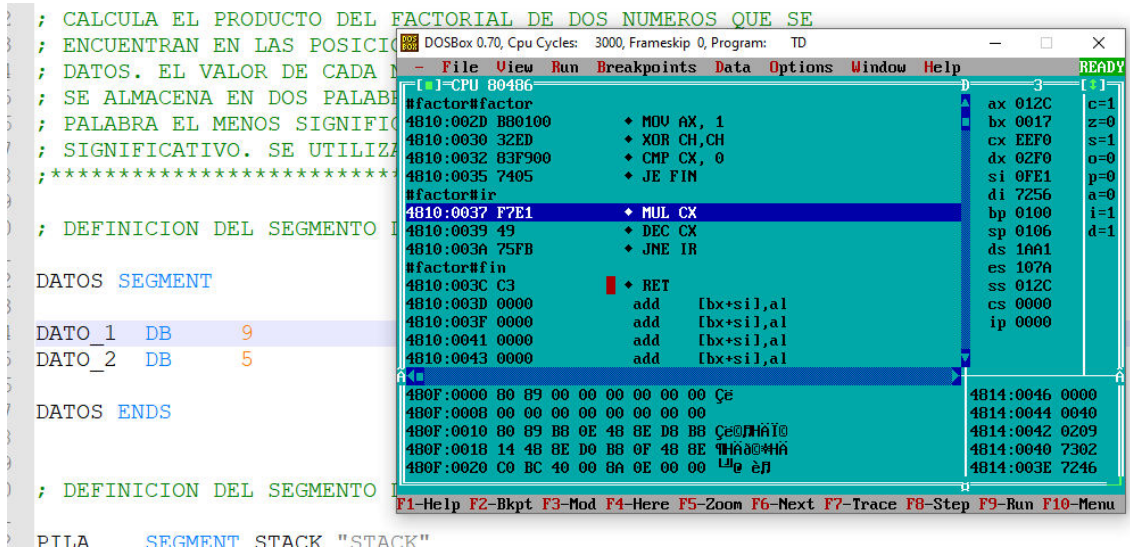
Address	Instruction	Comment	Register	Value
4810:002D	MOV AX, 1		ax	012C
4810:0030	XOR CH, CH		bx	0031
4810:0032	CMP CX, 0		cx	00CE
4810:0035	JE FIN		dx	0926
4810:0037	MUL CX		si	EE4A
4810:0039	DEC CX		di	009E
4810:003A	JNE IR		bp	0100
4810:003C	RET		sp	0106
4810:003D	add [bx+si], al		ds	1A01
4810:003F	add [bx+si], al		es	EE34
4810:0041	add [bx+si], al		ss	012C
4810:0043	add [bx+si], al		cs	0000

F1-Help F2-Bkpt F3-Mod F4-Here F5-Zoom F6-Next F7-Trace F8-Step F9-Run F10-Menu

8! = 40320 = 9D80h, resultado correcto.

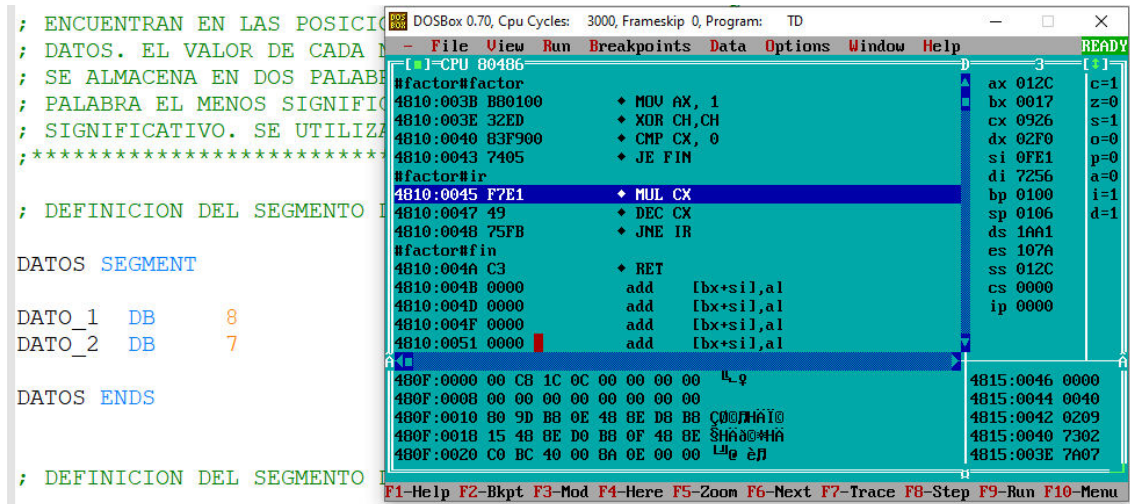
FACT_DATO_1 = 9D80h, ya que he usado DATO_1 = 8.

He añadido las 3 líneas de la captura para en vez de calcular el producto de los dos factoriales calcular solo un factorial.


$$9! = 362880 = 058980h.$$

Pero me sale el número 8980h, por lo que vemos que falta el Byte 05. Podemos comprobar que 8980h es el resultado truncado. Una solución sería manejar la multiplicación no con MUL sino una más compleja, con registros más grandes o hacerla manualmente usando más Bytes de memoria.

FACT_DATO_1 = 8980h, ya que he usado DATO_1 = 9, y el resultado de FACT_DATO_1 tiene también un overflow, de los cuales se queda con el resultado truncado a dos Bytes.


$$8! * 7! = 203212800 = \text{C1CC800h}.$$

Pero me sale el número C800h, que como en el cálculo anterior es el resultado truncado a los dos últimos Bytes. Una solución sería manejar la multiplicación no con MUL sino una más compleja, con registros más grandes o hacerla manualmente usando más Bytes de memoria.

FACT DATO1 = 9D80h.

Ejercicio 2:

```

;MOV CL, DATO_1
;CALL FACTOR
;MOV FACT_DATO_1, AX
;MOV CL, DATO_2
;CALL FACTOR
;MOV BX, FACT_DATO_1
;MUL BX

```

```

;MOV CL, DATO_1
;CALL FACTOR
;MOV FACT_DATO_1, AX

```

```

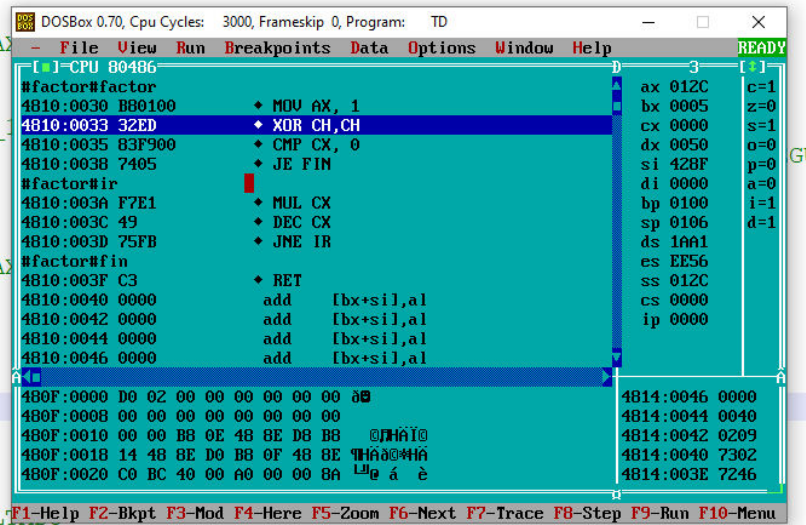
MOV AL, DATO_1
MOV BL, DATO_2
MUL BX
MOV CX, AX
CALL FACTOR

```

```

; ALMACENA EL RESULTADO

```



$(2 * 3)! = 720 = 02D0h$, resultado correcto.

He añadido las instrucciones necesarias para hacer el factorial de la multiplicación en vez de la multiplicación de los factoriales, da el resultado correcto.

```

; ENCUESTRAN EN LAS POSICIONES DE MEMORIA 0 Y 1 DEL SEGMENTO DE
; DATOS. EL VALOR DE CADA NUMERO DEBE SER INFERIOR A 9. EL RESULTADO
; SE ALMACENA EN DOS PALABRAS.
; PALABRA EL MENOS SIGNIFICATIVO. SE UTILIZA LA PALABRA DEL MENOS
; SIGNIFICATIVO. SE UTILIZA LA PALABRA DEL MENOS SIGNIFICATIVO.
; *****

```

```

; DEFINICION DEL SEGMENTO

```

```

DATOS SEGMENT

```

```

DATO_1 DB 2

```

```

DATO_2 DB 4

```

```

DATOS ENDS

```

```

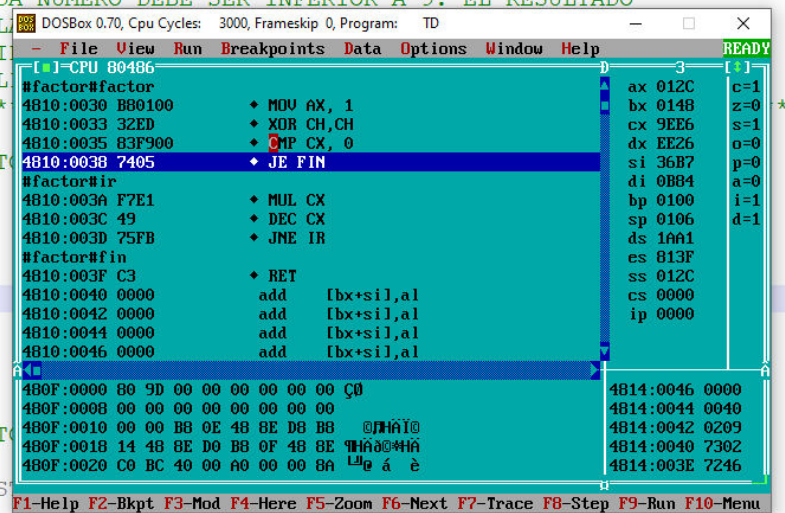
; DEFINICION DEL SEGMENTO

```

```

PILA SEGMENT STACK "STACK"
DB 40H DUP (0)

```



$(2 * 4)! = 40320 = 9D80h$.

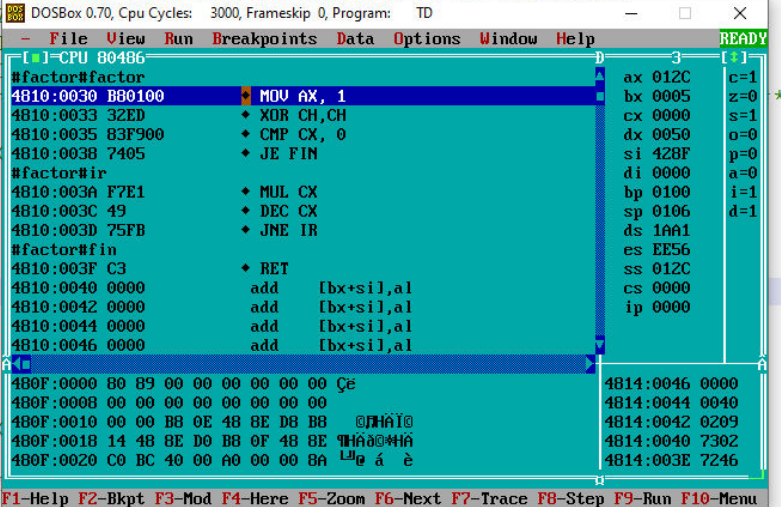
El resultado es correcto.


```

; DATOS. EL VALOR DE CADA NUMERO DEBE SER INFERIOR A 9. EL RESULTADO
; SE ALMACENA EN DOS PALABRAS. EL VALOR DE CADA PALABRA EL MENOS SIGNIFICATIVO. SE UTILIZA EL VALOR MAS SIGNIFICATIVO. SE UTILIZA EL VALOR MAS SIGNIFICATIVO.
; *****
; DEFINICION DEL SEGMENTO DATOS
DATOS SEGMENT
    DATO_1 DB 3
    DATO_2 DB 3
DATOS ENDS

; DEFINICION DEL SEGMENTO PILA
PILA SEGMENT STACK "STACK"
    DB 40H DUP (0)
PILA ENDS

```



$(3 * 3)! = 362880 = 058980h$.

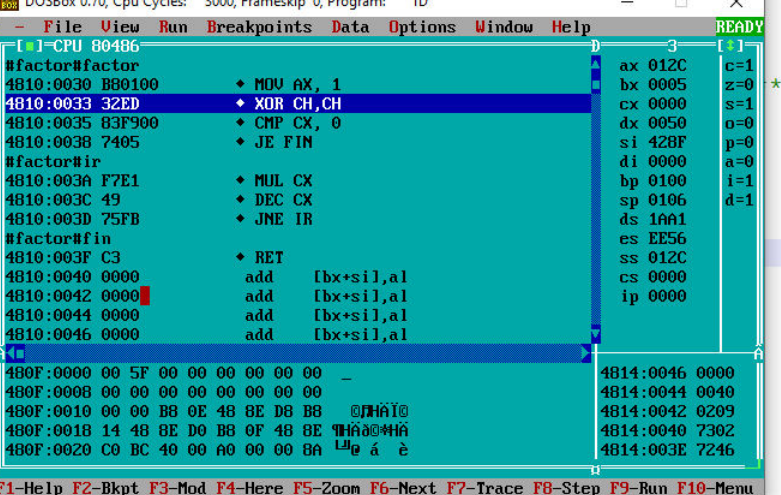
Pero me sale el número 8980h, por lo que vemos que falta el Byte 05. Podemos comprobar que 8980h es el resultado truncado. Una solución sería manejar la multiplicación no con MUL sino una más compleja, con registros más grandes o hacerla manualmente usando más Bytes de memoria.

```

; DATOS. EL VALOR DE CADA NUMERO DEBE SER INFERIOR A 9. EL RESULTADO
; SE ALMACENA EN DOS PALABRAS. EL VALOR DE CADA PALABRA EL MENOS SIGNIFICATIVO. SE UTILIZA EL VALOR MAS SIGNIFICATIVO. SE UTILIZA EL VALOR MAS SIGNIFICATIVO.
; *****
; DEFINICION DEL SEGMENTO DATOS
DATOS SEGMENT
    DATO_1 DB 5
    DATO_2 DB 2
DATOS ENDS

; DEFINICION DEL SEGMENTO PILA
PILA SEGMENT STACK "STACK"
    DB 40H DUP (0)
PILA ENDS

```



$(5 * 2)! = 3628800 = 375F00h$.

Pero nos sale el resultado 5F00h, igual que en el cálculo anterior es el resultado correcto pero truncado.