

# SOLUCION EJERCICIO 2

## DEBU2

### ENUNCIADO

Realizar un programa para almacenar en el siguiente segmento de memoria, los siguientes datos:

12233344445555666667777778888888999999999,

y guardar el programas en memoria auxiliar (disco duro) y luego volver a ejecutarlo.

## PRIMERA PARTE DEL EJERCICIO

Realizar un programa para almacenar en el siguiente segmento de memoria, los siguientes datos:

12233344445555666667777778888888999  
999999

# PRIMERA PARTE DEL EJERCICIO

## SEGMENTOS Y DESPLAZAMIENTOS

- ◆ MOSTRAMOS LOS SEGMENTOS DE LA MEMORIA CON LA INSTRUCCION D

```
C:\>debug  
-d  
06B0:0100 C3 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 00 . . .  
06B0:0110 00 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 00 . . .  
06B0:0120 00 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 00 . . .  
06B0:0130 00 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 00 . . .  
06B0:0140 00 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 00 . . .  
06B0:0150 00 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 00 . . .  
06B0:0160 00 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 00 . . .  
06B0:0170 00 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 00 . . .
```

AL DECIR SIGUIENTE SEGMENTO SUMAMOS 1 A NUESTRO SEGMENTO ACTUAL  
(SEGMENTO ACTUAL + 1 = SIGUIENTE SEGMENTO)

EN ESTE CASO EL SEGMENTO ACTUAL ES 06B0

EL SIGUIENTE SEGMENTO SERA  $06B0 + 1 = 06B1$  (LA SUMA DEBE SER EN HEX)

```
C:\>debug -d  
NUESTRO SEGMENTO ACTUAL ESTA UBICADO EN 06B0 (VARIA EN CADA MAQUINA)  
06B0:0100 C3 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 00  
06B0:0110 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 00  
06B0:0120 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 00  
06B0:0130 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 00  
06B0:0140 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 00  
06B0:0150 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 00  
06B0:0160 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 00  
06B0:0170 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 00
```

## PRIMERA PARTE DEL EJERCICIO

CUALQUIER DATO SIEMPRE SE ALMACENA EN EL DS (DATA SEGMENT, SEGMENTO DE DATOS)

PARA VER EN QUE DIRECCION ESTA EL DATA SEGMENT (DS) USAMOS “R”

LA INSTRUCCIÓN ‘R’ NOS MUESTRA LAS DIRECCIONES DE LOS SEGMENTOS Y LOS REGISTROS

```
R
AX=0000 BX=0000 CX=0000 DX=0000 SP=FFFE BP=0000 SI=0000 DI=0000
DS=06B0 ES=06B0 SS=06B0 CS=06B0 IP=0100 NV UP EI PL NZ NA PO NC
06B0:0100 C3
RET
S
```

PODEMOS VER QUE EL DS (SEGMENTO DE DATOS) ESTA EN LA DIRECCION ACTUAL **06B0**

## PRIMERA PARTE DEL EJERCICIO

ENTONCES PARA ALMACENAR EN EL SIGUIENTE SEGMENTO DE MEMORIA NUESTROS DATOS DEBEMOS DIRECCIONAR EL SEGMENTO DE DATOS (DS) AL SIGUIENTE SEGMENTO

SIGUIENTE SEGMENTO **06B0 + 1 = 06B1**

PARA CAMBIAR LA DIRECCION DEL ‘DS’ USAMOS “R DS”

LA INSTRUCCIÓN ‘R DS’ NOS MUESTRA LA DIRECCION ACTUAL DEL SEGMENTO DE DATOS ‘DS’ ESPERANDO LA DIRECCION A LA CUAL QUEREMOS CAMBIARLO

ENTONCES PARA NUESTRO EJERCICIO INSERTAMOS LA DIRECCION DEL SIGUIENTE SEGMENTO **06B1**

```
-R DS  
DS 06B0 :06B1  
-R  
AX=0000 BX=0000 CX=0000 DX=0000 SP=FFFE BP=0000 SI=0000 DI=0000  
DS=06B1 ES=06B0 SS=06B0 CS=06B0 IP=0100 NV UP EI PL NZ NA PO NC  
06B0:0100 C3           RET
```

VOLVEMOS A EJECUTAR LA INSTRUCCION ‘R’

PODEMOS VER QUE EL DS (SEGMENTO DE DATOS) AHORA ESTA EN LA DIRECCION SIGUIENTE **06B1**

## PRIMERA PARTE DEL EJERCICIO

AHORA ESCRIBIMOS NUESTRO PROGRAMA PARA GENERAR Y ALMACENAR

12233444555566667777788888899999999 EN EL SIGUIENTE SEGMENTO DE MEMORIA

NOTA: NUESTRO PROGRAMA LO ESCRIBIREMOS EN EL SEGMENTO DE MEMORIA ACTUAL EN EL DESPLAZAMIENTO 0100

PARA ESO EJECUTAMOS 'A 06B0:0100' Y EMPEZAMOS A ESCRIBIR NUESTRO PROGRAMA

```
-a 06b0:0100  
06B0:0100 mov cl,9
```

EXPLICANDO  
EL  
PROGRAMA

MOV	CL, 09	MOVEMOS EL VALOR DE CL A 9 (NOS SERVIRA PARA EL PRIMER LOOP)
MOV	BL, 01	MOVEMOS EL VALOR 01 A BL (VALOR QUE NOS AYUDARA A CONTROLAR EL SEGUNDO LOOP) Y TAMBIEN VALOR QUE SE ALMACENARA EN LA MEMORIA
MOV	SI, 0000	MOVEMOS 0000 A SI (SI ES EL REGISTRO QUE APUNTA A DIRECCIONES DE DS (DATA SEGMENT) EN ESTE CASO APUNTARA A LA DIRECCION 0000)
MOV	BH, CL	GUARDAMOS EL VALOR DE CL ACTUAL ES DECIR EL NRO DE REPETICION DEL PRIMER LOOP
MOV	CL, BL	AQUI DAMOS EL VALOR DE BL -> (1,2,3.....) A CL (PARA EL SEGUNDO LOOP)
ADD	BL, 30	SUMAMOS 30 A BL (BL -> 31) VALOR ASCCI DE '1','2','3',.....
MOV	[SI], BL	ALMACENAMOS BL EN EL ESPACIO DE MEMORIA EN LA DIRECCION SI EN DS(DATA SEGMENT)
INC	SI	INCREMENTAMOS SI ES DECIR PASAMOS A LA SIGUIENTE DIRECCION DE ESPACIO DE MEMORIA
LOOP	010E	SEGUNDO LOOP QUE GENERARA Y ALMACENARA DATOS DE BL (1,22,333,4444,....) EL VALOR DEBE SER EL DESPLAZAMIENTO DE LA INSTRUCCION (MOV [SI],BL) EN MI CASO ES 00E
SUB	BL, 30	RESTAMOS 30 A BL (BL -> 1) VALOR PARA EL SEGUNDO LOOP (1,2,3,4.....)
INC	BL	INCREMENTAMOS BL (1,2,3,4.....)
MOV	CL, BH	REGRESAMOS EL VALOR DE CL INICIAL DEL PRIMER LOOP QUE ESTABA GUARDADO EN BH
LOOP	0107	PRIMER LOOP QUE HARA 9 REPETICIONES, EL VALOR DEBE SER EL DESPLAZAMIENTO DE LA INSTRUCCION (MOV BH,CL EN MI CASO ES 0107)
INT	20	

'[SI]' ES DIFERENTE DE 'SI'

- '[SI]' ES EL **ESPACIO EN LA MEMORIA** (EN EL DS(DATA SEGMENT)) EN LA DIRECCION SI
- 'SI' ES UN REGISTRO (UNA VARIABLE QUE ALMACENA UNA DIRECCION DE ESPACIO DE MEMORIA)

## PRIMERA PARTE DEL EJERCICIO

EJECUTAMOS NUESTRO PROGRAMA CON LA INSTRUCCION 'G'

VEAMOS LOS DATOS EN EL SIGUIENTE SEGMENTO DE MEMORIA **06B1** EN EL DESPLAZAMIENTO 0000  
CON LA INSTRUCCION D 06B1:0000

```
-g

Program terminated (0000)
-d 06b1:0000
06B1 0000 31 32 32 33 33 33 34 34-34 34 35 35 35 35 35 36 1223334444555556
06B1 0010 36 36 36 36 36 37 37 37-37 37 37 37 37 38 38 38 38 6666677777778888
06B1 0020 38 38 38 38 39 39 39 39-39 39 39 39 39 00 00 00 00 88889999999999...
06B1 0030 05 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 00 ...
06B1 0040 CD 21 CB 00 00 00 00 00 00-00 00 00 00 00 20 20 20 20 .†.....
06B1 0050 20 20 20 20 20 20 20 20-00 00 00 00 00 00 20 20 20 20 .....
06B1 0060 20 20 20 20 20 20 20 20-00 00 00 00 00 00 00 00 00 00 .....
06B1 0070 00 00 0D 00 00 00 00 00 00-00 00 00 00 00 00 00 00 00 .....
```

DEZPLAZAMIENTO

DIRECCION DE SEGMENTO

## SEGUNDA PARTE DEL EJERCICIO

y guardar el programas en memoria auxiliar  
(disco duro) y luego  
volver a ejecutarlo.

## SEGUNDA PARTE DEL EJERCICIO

ALMACENAR EL PROGRAMA EN EL DISCO DURO, PRIMERO OBTENEMOS EL TAMAÑO DEL PROGRAMA

06B0:0100	MOV	CL,09
06B0:0102	MOV	BL,01
06B0:0104	MOV	SI,0000
06B0:0107	MOV	BH,CL
06B0:0109	MOV	CL,BL
06B0:010B	ADD	BL,30
06B0:010E	MOV	[SI],BL
06B0:0110	INC	SI
06B0:0111	LOOP	010E
06B0:0113	SUB	BL,30
06B0:0116	INC	BL
06B0:0118	MOV	CL,BH
06B0:011A	LOOP	0107
06B0:011C	INT	20

TAMAÑO EN BYTES SERA LOS DESPLAZAMIENTOS  
(EN EL DESPLAZAMIENTO 0102 ESTARAN DOS BYTES ES DECIR 0102 Y 0103  
(EN EL DESPLAZAMIENTO 011C ESTARAN DOS BYTES ES DECIR 11C Y 11D

VEMOS EL TAMAÑO DE NUESTRO PROGRAMA  
EN ESTE CASO SERIA ( $1C + 1 = 1D$ , QUE SERIAN  
**29 BYTES**)

TOMAMOS 1C PORQUE NUESTRO  
DESPLAZAMIENTO EMPIEZA EN 0100 Y  
NUESTRO PROGRAMA OCUPA HASTA 011C

## SEGUNDA PARTE DEL EJERCICIO

UNA VEZ TENIENDO EL TAMAÑO DEL PROGRAMA PROCEDEMOS A ALMACENARLO EN EL DISCO DURO

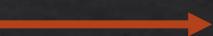
```
n ejer2  
r cx  
cx 0000 :29  
w  
riting 0029 bytes  
q
```

DAMOS UN NOMBRE A NUESTRO PROGRAMA ‘EJER2’ CON LA INSTRUCCION ‘N’  
EJECUTAMOS ‘R CX’ QUE ESPERARA POR UN VALOR,DAMOS EL TAMAÑO DEL  
PROGRAMA A CX (29)  
CON LA INSTRUCCION W GRABAMOS EL PROGRAMA EN EL DISCO DURO  
VEMOS QUE SE GRABO EL PROGRAMA CORRECTAMENTE  
SALIMOS DE DEBUG

## SEGUNDA PARTE DEL EJERCICIO

PARA VERIFICAR QUE EL PROGRAMA HAYA SIDO GRABADO EJECUTAMOS EL COMANDO DIR EN DOS

VEMOS QUE  
NUESTRO  
PROGRAMA  
SE GRABO



```
.          <DIR>      30-08-2023 14:09
..
DEBUG     ASM      230,208 14-05-2016 10:32
DEBUG     COM      20,788 14-05-2016 10:32
DEBUG     TXT      27,998 12-04-2016 11:44
DEBUGTBL   INC      46,360 10-05-2016 10:44
DEBUGX    COM      26,164 14-05-2016 10:32
DPMICL32  ASM      3,130 14-04-2016  9:05
DPMICL32  EXE      863 14-04-2016  9:05
EJER2      41 30-08-2023 14:09
FPTOSTR   INC      7,908 28-11-2014 14:28
HISTORY   TXT      17,574 14-05-2016 10:04
INSTR     KEY      5,740 10-03-2014 11:09
INSTR     ORD      576 25-01-2014 10:59
INSTR     SET      9,309 11-03-2016 19:21
MAKE      BAT      1,320 04-02-2014 19:26
MAKEC     BAT      335 20-01-2014 14:34
MKTABLES  C      31,542 26-02-2014 13:50
PROG1      22 29-08-2023 20:33
PROG1      COM      22 29-08-2023 20:32
README    TXT      7,095 10-05-2016 10:44
 19 File(s)      436,995 Bytes.
 2 Dir(s)       262,111,744 Bytes free.
```

C:\>

## SEGUNDA PARTE DEL EJERCICIO

PARA VOLVER A EJECUTAR EL PROGRAMA, ESCRIBIMOS EL COMANDO ‘DEBUG NOMPROG’ EN ESTE CASO ‘DEBUG EJER2’

C:\>DEBUG EJER2		
-U 100 L 1D		
06B0:0100 B109	MDV	CL,09
06B0:0102 B301	MDV	BL,01
06B0:0104 BE0000	MDV	SI,0000
06B0:0107 88CF	MDV	BH,CL
06B0:0109 88D9	MDV	CL,BL
06B0:010B 80C330	ADD	BL,30
06B0:010E 881C	MDV	[SI],BL
06B0:0110 46	INC	SI
06B0:0111 E2FB	LOOPW	010E
06B0:0113 80EB30	SUB	BL,30
06B0:0116 FEC3	INC	BL
06B0:0118 88F9	MDV	CL,BH
06B0:011A E2EB	LOOPW	0107
06B0:011C CD20	INT	20
-G		

UNA VEZ DENTRO DE DEBUG  
DESENSAMBLAMOS EL  
PROGRAMA CON LA INSTRUCCION  
*‘U NRODESPLAZAMIENTO L TAMAÑODELPROGRAMA’*  
EN NUESTRO CASO ‘U 100 L 1D’

UNA VEZ DESENSAMBLADO  
VEMOS TODO EL CODIGO DEL  
PROGRAMA  
LUEGO EJECUTAMOS LA  
INSTRUCCION G

# FIN

RECUERDA PARA TENER EXITO EN LA PROGRAMACION HAY 3 REGLAS PRACTICA, PRACTICA Y PRACTICA