### **Ejercicio 6 - Memoria dinámica**

Con este ejercicio se pretende probar el funcionamiento de memoria dinámica. Se solicitan varias celdas de memoria, se liberan algunas y solicitan nuevamente otras. Se guardan las direcciones obtenidas en el DS y luego se recorren los encabezados de cada nodo del ES para verificar la correcta implementación.

#### **6.1 Traducción**

##### **Ejecutar:**

##### **>mvc 6.asm 6.bin**

|  |
| --- |
| Warning: truncado de operando en linea 32.  [0000] B5 00 E0 0E XOR ex, ex  [0001] 04 00 C0 04 MOV cx, 4  [0002] F0 00 00 05 SYS %5  [0003] 14 00 E0 01 ADD ex, 1  [0004] 0D 00 E0 0D MOV [ex], dx  [0005] 04 00 C0 08 MOV cx, 8  [0006] F0 00 00 05 SYS %5  [0007] 14 00 E0 01 ADD ex, 1  [0008] 0D 00 E0 0D MOV [ex], dx  [0009] 04 00 C0 0A MOV cx, 10  [0010] F0 00 00 05 SYS %5  [0011] 14 00 E0 01 ADD ex, 1  [0012] 0D 00 E0 0D MOV [ex], dx  [0013] 04 00 C0 05 MOV cx, 5  [0014] F0 00 00 05 SYS %5  [0015] 14 00 E0 01 ADD ex, 1  [0016] 0D 00 E0 0D MOV [ex], dx  [0017] 04 00 C0 05 MOV cx, 5  [0018] F0 00 00 05 SYS %5  [0019] 14 00 E0 01 ADD ex, 1  [0020] 0D 00 E0 0D MOV [ex], dx  [0021] 06 00 D0 03 MOV dx, [3]  [0022] F0 00 00 06 SYS %6  [0023] 06 00 D0 01 MOV dx, [1]  [0024] F0 00 00 06 SYS %6  [0025] 04 00 C0 06 MOV cx, 6  [0026] F0 00 00 05 SYS %5  [0027] 14 00 E0 01 ADD ex, 1  [0028] 0D 00 E0 0D MOV [ex], dx  [0029] 06 00 D0 04 MOV dx, [4]  [0030] F0 00 00 06 SYS %6  [0031] 14 00 E0 01 ADD ex, 1  [0032] 0C 00 EF FF MOV [ex], -1 ;marcador  [0033] B5 00 F0 0F XOR fx, fx  [0034] 64 00 E0 0D COPY: CMP ex, 13  [0035] F7 00 00 2A JNN fin  [0036] 14 00 F0 01 ADD fx, 1  [0037] 07 00 D0 0F MOV dx, [fx]  [0038] 24 00 D0 01 SUB dx, 1  [0039] 14 00 E0 01 ADD ex, 1  [0040] 0F 00 E0 0D MOV [ex], [dx]  [0041] F1 00 00 22 JMP copy  [0042] 04 00 A0 08 FIN: MOV ax, %8  [0043] 04 00 C0 0E MOV cx, 14  [0044] 04 00 D0 01 MOV dx, 1  [0045] F0 00 00 02 SYS %2  [0046] F0 00 00 0F SYS %f  [0047] FF 10 00 00 STOP |

##### 

#### **6.2 Ejecución**

##### **Ejecutar:**

##### **>mvx 6.bin**

|  |
| --- |
| [0001]:%00020001  %00020006  %0002000F  %0002001A  %00020020  %0002000F  %FFFFFFFF  %00040015  %0008000E  %0006001F  %00050025  %00050005  %0006001F  %00000000 |

##### 

### **Ejercicio 7 - Strings**

Con este ejercicio se pretende probar el trabajo con strings, además de los breakpoints, desensamblado y muestra de memoria.

#### **7.1 Traducción**

##### **Ejecutar:**

##### **>mvc 7.asm 7.bin**

|  |
| --- |
| [0000] F9 00 00 02 LDH 2  [0001] F8 00 00 00 LDL 0  [0002] 0D 00 90 09 MOV [ac], ac  [0003] 1C 00 90 01 ADD [ac], 1  [0004] F9 00 00 03 LDH 3  [0005] F8 00 00 2D LDL msg  [0006] 04 00 A8 00 MOV ax, %800  [0007] 04 00 C3 E8 MOV cx, 1000  [0008] 05 00 D0 09 MOV dx, ac  [0009] F0 00 00 04 SYS %4  [0010] 04 00 C0 01 MOV cx, 1  [0011] F9 00 00 02 LDH 2  [0012] F8 00 00 00 LDL 0  [0013] 04 00 A9 00 INI: MOV ax, %900  [0014] 04 00 C3 E8 MOV cx, 1000  [0015] 04 00 D0 00 MOV dx, 0  [0016] F0 00 00 03 SYS %3  [0017] C6 00 C0 00 SLEN cx, [0]  [0018] 64 00 C0 00 CMP cx, 0 ;Si estÃ¡ vacÃ­a...  [0019] F2 00 00 19 JZ finlee ;...termina la lectura  [0020] 07 00 D0 09 MOV dx, [ac] ;Sino muevo el HEAP a DX  [0021] 1D 00 90 0C ADD [ac], cx ;Incremento el HEAP para reservar la memoria  [0022] 1C 00 90 01 ADD [ac], 1 ;Uno mÃ¡s por el \0  [0023] DE 00 D0 00 SMOV [dx], [0] ;Agrega la palabra en el ES  [0024] F1 00 00 0D JMP ini  [0025] F8 00 00 01 FINLEE: LDL 1  [0026] 05 00 D0 09 MOV dx, ac ;Inicializa 1 para comenzar el recorrido  [0027] F8 00 00 00 LDL 0  [0028] 07 00 C0 09 MOV cx, [ac]  [0029] 24 00 C0 01 SUB cx, 1 ;Posiciona a cx en el Ãºltimo \0  [0030] 65 00 D0 0C NEXT: CMP dx, cx ;Termina cuando DX llega al Ãºltimo \0  [0031] F2 00 00 26 JZ fin  [0032] 9C 00 D0 DF MAY: AND [dx], %DF ;Pasa a MAYUSCULAS  [0033] 6C 00 D0 00 CMP [dx], 0 ;Si encuentra un \0 ...  [0034] F5 00 00 24 JNZ sig  [0035] 0C 00 D0 20 MOV [dx], %20 ;... pone un espacio para contactenar  [0036] 14 00 D0 01 SIG: ADD dx, 1  [0037] F1 00 00 1E JMP next  [0038] F0 00 00 0F FIN: SYS %F ;mostrar el ES  [0039] F8 00 00 01 LDL 1  [0040] F9 00 00 02 LDH 2  [0041] 05 00 D0 09 MOV dx, ac  [0042] 04 00 A9 00 MOV ax, %900  [0043] F0 00 00 04 SYS %4  [0044] FF 10 00 00 STOP |

##### 

#### **7.2 Ejecución**

##### **Ejecutar:**

##### **>mvx 7.bin -b -d**

Cuando el programa solicite escribir palabras escriba (para finalizar presione ENTER sin escribir caracteres):

uno

dos

tres

Cuando el debug solicite cmd, escriba el rango de memoria correspondiente a las **celdas 1 a 13 del Extra segment**. Luego continúe con la ejecución**.** Si todo está correcto, el programa terminará escribiendo “UNO DOS TRES”.

A continuación pegue el estado de los registros, las direcciones de memoria solicitadas, el contenido del rango de memoria y la impresión final por pantalla:

|  |
| --- |
| Ejecucion con el archivo dado:  Escriba palabras seguidas de ENTER (en blanco para terminar)  uno  dos  tres  Segmentation Fault.  Correccion:  Modificando el tamaño del ES a 20el error desaparece, devolviendo la siguiente salida por pantalla:  >[0000] F9 00 00 02 1: LDH 2  [0001] F8 00 00 00 2: LDL 0  [0002] 0D 00 90 09 3: MOV 106, AC  [0003] 1C 00 90 01 4: ADD 106, 1  [0004] F9 00 00 03 5: LDH 3  [0005] F8 00 00 2D 6: LDL 45  [0006] 04 00 A8 00 7: MOV AX, -2048  [0007] 04 00 C3 E8 8: MOV CX, 1000  [0008] 05 00 D0 09 9: MOV DX, AC  [0009] F0 00 00 04 10: SYS 4  Registros:  | DS = 0400006A | SS = 0400047E | ES = 0014046A | CS = 006A0000 |  | HP = FFFFFFFF | IP = 0 | SP = 66560 | BP = 66560 |  | CC = 0 | AC = 0 | AX = 0 | BX = 0 |  | CX = 0 | DX = 0 | EX = 0 | FX = 0 |  Escriba palabras seguidas de ENTER (en blanco para terminar)  uno  dos  tres  [0034] F5 00 00 24 35: JNZ 36  [0035] 0C 00 D0 20 36: MOV 1143, 32  [0036] 14 00 D0 01 37: ADD DX, 1  [0037] F1 00 00 1E 38: JMP 30  [0038] F0 00 00 0F 39: SYS 15  >[0039] F8 00 00 01 40: LDL 1  [0040] F9 00 00 02 41: LDH 2  [0041] 05 00 D0 09 42: MOV DX, AC  [0042] 04 00 A9 00 43: MOV AX, -1792  [0043] F0 00 00 04 44: SYS 4  Registros:  | DS = 0400006A | SS = 0400047E | ES = 0014046A | CS = 006A0000 |  | HP = FFFFFFFF | IP = 39 | SP = 66560 | BP = 66560 |  | CC = 1 | AC = 131072 | AX = -1792 | BX = 0 |  | CX = 131085 | DX = 131085 | EX = 0 | FX = 0 |  [0039] cmd: 1131 1144  [1131]: 0000 0055 85  [1132]: 0000 004E 78  [1133]: 0000 004F 79  [1134]: 0000 0020 32  [1135]: 0000 0044 68  [1136]: 0000 004F 79  [1137]: 0000 0053 83  [1138]: 0000 0020 32  [1139]: 0000 0054 84  [1140]: 0000 0052 82  [1141]: 0000 0045 69  [1142]: 0000 0053 83  [1143]: 0000 0000 0  [1144]: 0000 0000 0  UNO DOS TRES |

##### 

### **Ejercicio 8 - Indirección**

Con este ejercicio se pretende probar los operandos indirectos. La intención es escribir en DS, ES y SS. Al final imprime los valores de las 3 primeras celdas de ES, luego de DS y finalmente una celda del SS.

El SYS debe interpretar DX como relativo al segmento, al igual que en una indirección.

#### **8.1 Traducción**

##### **Ejecutar:**

##### **>mvc 8.asm 8.bin**

Pegar el resultado mostrado por consola:

|  |
| --- |
| Warning: truncado de operando en linea 8.  Warning: truncado de operando en linea 11.  Warning: truncado de operando en linea 13.  [0000] 04 00 A0 01 MOV ax, 1  [0001] 04 00 B0 05 MOV bx, 5  [0002] 04 00 C0 02 MOV cx, 2  [0003] 64 00 A0 07 OTRO: CMP ax, 7  [0004] F2 00 00 08 JZ sigue  [0005] 0D 00 A0 0A MOV [ax], ax  [0006] 14 00 A0 01 ADD ax, 1  [0007] F1 00 00 03 JMP otro  [0008] 4C FF C0 0A SIGUE: MUL [CX+ant], 10  [0009] 4C 00 C0 0A MUL [Cx], 10  [0010] 4C 01 C0 0A MUL [cx+sig], 10  [0011] 4C FF B0 0A MUL [bx-1], 10  [0012] 4C 00 B0 0A MUL [Bx], 10  [0013] 4C FF B0 0A MUL [BX-ant], 10  [0014] 0F 00 A0 1C MOV [ax], [cx-sig]  [0015] FC C0 00 0A PUSH [ax]  [0016] 05 00 70 06 MOV BP, SP  [0017] 1C 00 70 01 ADD [BP], 1  [0018] F8 00 00 01 LDL 1  [0019] F9 00 00 02 LDH 2  [0020] 05 00 D0 09 MOV dx, ac  [0021] F9 00 00 00 LDH 0  [0022] 0F 00 D0 09 MOV [dx], [ac]  [0023] 0F 01 D0 19 MOV [dx+sig], [ac+sig]  [0024] 0F 02 D0 29 MOV [dx+2], [ac+2]  [0025] 04 00 A0 01 MOV ax, %1  [0026] 04 00 C0 03 MOV cx, 3  [0027] F0 00 00 02 SYS %2  [0028] 04 00 D0 01 MOV dx, 1  [0029] 04 00 C0 07 MOV cx, 7  [0030] F0 00 00 02 SYS %2  [0031] 05 00 D0 07 MOV dx, bp  [0032] 04 00 C0 01 MOV cx, 1  [0033] F0 00 00 02 SYS %2  [0034] F0 00 00 0F SYS %F  [0035] FF 10 00 00 STOP  [0036] FF 10 00 00 STOP  [0037] FF 10 00 00 STOP  [0038] FF 10 00 00 STOP  [0039] FF 10 00 00 STOP |

#### **8.2 Ejecución**

##### **Ejecutar:**

**>mvx 8.bin**

|  |
| --- |
| Segmentation Fault.3 |

##### 

### **Ejercicio 9 - Fibonacci**

Con este ejercicio se pretende evaluar el manejo de la pila.

#### **9.1 Traducción**

##### **Ejecutar:**

**>mvc 9.asm 9.bin**

Pegar el resultado mostrado por consola:

|  |
| --- |
| Warning: truncado de operando en linea 20.  Warning: truncado de operando en linea 21.  Warning: truncado de operando en linea 22.  Warning: truncado de operando en linea 27.  [0000] 04 00 D0 00 MOV dx, buffer  [0001] 04 00 C0 01 MOV cx, 1  [0002] 04 00 A0 01 MOV ax, %1  [0003] FC 00 00 01 PUSH 1  [0004] FC 00 00 01 PUSH 1  [0005] FC 00 00 0A PUSH 10  [0006] FE 00 00 0E CALL fibo  [0007] 14 00 60 03 ADD sp, 3  [0008] 0C 00 00 01 MOV [buffer], 1  [0009] F0 00 00 02 SYS %2  [0010] F0 00 00 02 SYS %2  [0011] F0 00 00 0F SYS %F  [0012] FD 40 00 0A POP ax  [0013] FF 10 00 00 STOP  [0014] FC 40 00 07 FIBO: PUSH bp  [0015] 05 00 70 06 MOV bp, sp  [0016] 24 00 60 01 SUB sp, 1  [0017] 6C 02 70 00 CMP [bp+2], 0  [0018] F2 00 00 1D JZ finf  [0019] 2C 02 70 01 SUB [bp+2], 1  [0020] 0F FF 70 37 MOV [bp-1], [bp+3]  [0021] 1F FF 70 47 ADD [bp-1], [bp+4]  [0022] FC C0 0F F7 PUSH [bp-1]  [0023] FC C0 00 47 PUSH [bp+4]  [0024] FC C0 00 27 PUSH [bp+2]  [0025] FE 00 00 0E CALL fibo  [0026] 14 00 60 03 ADD sp, 3  [0027] 0F 00 0F F7 MOV [buffer], [bp-1]  [0028] F0 00 00 02 SYS %2  [0029] 05 00 60 07 FINF: MOV sp, bp  [0030] FD 40 00 07 POP bp  [0031] FF 00 00 00 RET |

#### **9.2 Ejecución**

##### **Ejecutar:**

**>mvx 9.bin -d -b**

Cuando se detenga la ejecución por breakpoint solicitar mostrar direcciones de memoria correspondientes para que muestre todo el contenido de la pila. Luego finalizar la ejecución.

A continuación pegue el estado de los registros, las direcciones de memoria solicitadas, el contenido del rango de memoria y la impresión final por pantalla:

|  |
| --- |
| Segmentation Fault. |

### **Ejercicio 10 - Hanoi**

En este ejercicio se pretende realizar una evaluación integradora.

#### **10.1 Traducción**

##### **Ejecutar:**

>mvc A.asm A.bin

|  |
| --- |
| [0000] 0C 00 40 00 MOV [step], 0  [0001] F9 00 00 02 LDH 2  [0002] F8 00 00 32 LDL 50  [0003] 05 00 D0 09 MOV dx, ac  [0004] F9 00 00 03 LDH 3  [0005] F8 00 00 82 LDL msg  [0006] DF 00 D0 09 SMOV [dx], [ac]  [0007] F9 00 00 02 LDH 2  [0008] F8 00 00 0A LDL 10  [0009] 05 00 D0 09 MOV dx, ac  [0010] F9 00 00 03 LDH 3  [0011] F8 00 00 6B LDL titulo  [0012] DF 00 D0 09 SMOV [dx], [ac]  [0013] 04 00 A8 00 MOV ax, %800  [0014] F0 00 00 04 SYS %4  [0015] 14 00 D0 28 ADD dx, 40  [0016] 04 00 A9 00 MOV ax, %900  [0017] F0 00 00 04 SYS %4  [0018] 04 00 D0 00 MOV dx, 0  [0019] 04 00 C0 01 MOV cx, 1  [0020] 04 00 A8 01 MOV ax, %801  [0021] F0 00 00 01 SYS %1  [0022] 04 00 C0 01 MOV cx, strAUX  [0023] F9 00 00 03 LDH 3  [0024] F8 00 00 A1 LDL strA  [0025] 0D 00 C0 09 MOV [cx+torreA], ac  [0026] F8 00 00 A9 LDL strB  [0027] 0D 01 C0 09 MOV [cx+torreB], ac  [0028] F8 00 00 B1 LDL strC  [0029] 0D 02 C0 09 MOV [cx+torreC], ac  [0030] FC 00 00 02 PUSH torreC ;torre auxiliar (aux)  [0031] FC 00 00 01 PUSH torreB ;torre destino (dtn)  [0032] FC 00 00 00 PUSH torreA ;torre origen (org)  [0033] FC 80 00 00 PUSH [0] ;cantidad de discos a mover  [0034] FE 00 00 25 CALL hanoi  [0035] 14 00 60 05 ADD sp, 5  [0036] FF 10 00 00 STOP  [0037] FC 40 00 07 HANOI: PUSH bp  [0038] 05 00 70 06 MOV bp, sp  [0039] FC 40 00 0C PUSH cx  [0040] 07 00 C0 27 MOV cx, [BP+discos]  [0041] 64 00 C0 00 CMP cx, 0 ;si la cantidad a mover es 0...  [0042] F2 00 00 3D JZ finh ;...no hace nada  [0043] 24 00 C0 01 SUB cx, 1  [0044] FC C0 00 47 PUSH [BP+destino] ;destion pasa a auxiliar  [0045] FC C0 00 57 PUSH [BP+aux] ;auxiliar es el nuevo destino  [0046] FC C0 00 37 PUSH [BP+origen] ;el origen se mantiene  [0047] FC 40 00 0C PUSH cx  [0048] FE 00 00 25 CALL hanoi  [0049] 14 00 60 04 ADD sp, 4  [0050] FC C0 00 47 PUSH [BP+destino] ;destino  [0051] FC C0 00 37 PUSH [BP+origen] ;origen  [0052] FE 00 00 41 CALL print  [0053] 14 00 60 02 ADD sp, 2  [0054] FC C0 00 37 PUSH [BP+origen] ;el origen pasa a auxiliar  [0055] FC C0 00 47 PUSH [BP+destino] ;el destino se mantiene  [0056] FC C0 00 57 PUSH [BP+aux] ;el auxiliar pasa a ser origen  [0057] FC 40 00 0C PUSH cx  [0058] FE 00 00 25 CALL hanoi  [0059] 14 00 60 04 ADD sp, 4  [0060] F1 00 00 3D JMP finh  [0061] FD 40 00 0C FINH: POP cx  [0062] 05 00 60 07 MOV sp, bp  [0063] FD 40 00 07 POP bp  [0064] FF 00 00 00 RET  [0065] FC 40 00 07 PRINT: PUSH bp  [0066] 05 00 70 06 MOV bp, sp  [0067] FC 40 00 0A PUSH ax  [0068] FC 40 00 0B PUSH bx  [0069] FC 40 00 0C PUSH cx  [0070] FC 40 00 0D PUSH dx  [0071] F9 00 00 03 LDH 3  [0072] F8 00 00 BE LDL paso  [0073] 05 00 D0 09 MOV dx, ac  [0074] 04 00 B0 01 MOV bx, 1  [0075] 04 00 A9 00 MOV ax, %900  [0076] F0 00 00 04 SYS %4  [0077] 04 00 D0 04 MOV dx, step  [0078] 1C 00 D0 01 ADD [dx], 1  [0079] 04 00 C0 01 MOV cx, 1  [0080] 04 00 A9 01 MOV ax, %901  [0081] F0 00 00 02 SYS %2  [0082] 07 00 D0 27 MOV dx, [bp+desde]  [0083] 14 00 D0 01 ADD dx, strAUX  [0084] 07 00 D0 0D MOV dx, [dx]  [0085] 04 00 A9 00 MOV ax, %900  [0086] 04 00 B0 01 MOV bx, 1  [0087] F0 00 00 04 SYS %4  [0088] F9 00 00 03 LDH 3  [0089] F8 00 00 B9 LDL flecha  [0090] 05 00 D0 09 MOV dx, ac  [0091] 04 00 A9 00 MOV ax, %900  [0092] 04 00 B0 01 MOV bx, 1  [0093] F0 00 00 04 SYS %4  [0094] 07 00 D0 37 MOV dx, [bp+hasta]  [0095] 14 00 D0 01 ADD dx, strAUX  [0096] 07 00 D0 0D MOV dx, [dx]  [0097] 04 00 A8 00 MOV ax, %800  [0098] 04 00 B0 01 MOV bx, 1  [0099] F0 00 00 04 SYS %4  [0100] FD 40 00 0D FINP: POP dx  [0101] FD 40 00 0C POP cx  [0102] FD 40 00 0B POP bx  [0103] FD 40 00 0A POP ax  [0104] 05 00 60 07 MOV sp, bp  [0105] FD 40 00 07 POP bp  [0106] FF 00 00 00 RET |

#### 

#### **10.2 Ejecución**

##### **Ejecutar:**

>mvx A.bin

Ingrese la cantidad de discos:3

|  |
| --- |
| Segmentation Fault. |

##### 

>mvx A.bin

Ingrese la cantidad de discos:9

|  |
| --- |
| Segmentation Fault. |

Observación:

La aclaración al comienzo de la clase nos dio lugar a darnos cuenta de que teníamos un error en las indirecciones que no pudimos encontrar y que arrastramos en el resto de los ejercicios.

La salida de 0.bin nos da:

[0016]: Prueba

[0001]:0