Entrada por teclado haciendo uso de la INT 21h

Servicio 01h – entrada bloqueante con eco y atención a Ctrl-Break
El servicio 01h de la INT 21h espera a que se pulse una tecla y la escribe en la pantalla (lectura con eco) comprobando si es Ctrl-Break en cuyo caso dispara la interrupción de break de teclado que suele finalizar el programa en curso. El código ASCII del carácter se devuelve en el registro AL.

argumentos: ah = 01h **valores devueltos:** al = código ASCII leído

Servicio 08h – entrada bloqueante sin eco y atención a Ctrl-Break
El servicio 08h de la INT 21h espera a que se pulse una tecla sin eco y comprobando si es Ctrl-Break en cuyo caso dispara la interrupción de break de teclado. El código ASCII del carácter se devuelve en el registro AL.

argumentos: ah = 08h **valores devueltos:** al = código ASCII leído

Servicio 07h – entrada bloqueante sin eco y omisión de Ctrl-Break
El servicio 07h de la INT 21h espera a que se pulse una tecla sin eco y omitiendo un posible Ctrl-Break. El código ASCII del carácter se devuelve en el registro AL. argumentos: ah = 07h valores devueltos: al = código ASCII leído Servicio 08h – entrada bloqueante sin eco y atención a Ctrl-Break El servicio 08h de la INT 21h espera a que se pulse una tecla sin eco y comprobando si es Ctrl-Break en cuyo caso dispara la interrupción de break de teclado. El código ASCII del carácter se devuelve en el registro AL.

argumentos: ah = 08h **valores devueltos:** al = código ASCII leído

Funciones de entrada/salida de cadenas de caracteres de la INT 21h
Una cadena de caracteres (string en inglés) es una secuencia ordenada de longitud arbitraria de códigos alfanuméricos. El sistema de codificación puede ser ASCII, EBCDIC o UNICODE.

Físicamente las cadenas se pueden almacenar colocando cada código en posiciones consecutivas de memoria o enlazando carácter a carácter. El primer sistema es el más habitual y eficiente. Para diferenciar una cadena de otra, podemos reservar espacio en memoria de tamaño fijo, pero este método es ineficiente ya que desaprovecha espacio de almacenamiento. Habitualmente, lo más eficiente es manejar espacios de almacenamiento de longitud variable utilizando alguno de estos métodos:

- Indicando el comienzo y final de cada cadena mediante un carácter separador
- Indicando la longitud de la cadena antes de la misma
- Indicando el final de la cadena mediante un carácter terminador

En el caso de los servicios de entrada/salida de cadenas de la interrupción del sistema operativo INT 21h se utiliza el almacenamiento consecutivo en memoria de los códigos

alfanuméricos y se permite la longitud arbitraria de la cadena gracias al carácter terminador ‘enter’ para entrada de cadena y ‘$’ para salida de cadena.

Servicio 0Ah – entrada de cadena de caracteres
El servicio 0Ah de la INT 21h lee una secuencia de caracteres desde el teclado y los almacena en memoria hasta que se pulse la tecla ‘enter’. El área de memoria es una estructura de datos con 3 campos: el primer byte es el tamaño máximo permitido de la cadena incluido el código de la tecla ‘enter’, el segundo byte recibirá el número de caracteres leídos sin contar el ‘enter’ y el tercer campo es el buffer de memoria en el que se almacenará la cadena incluyendo el ‘enter’. Evidentemente, el tamaño del buffer deberá coincidir con el valor del primer campo. El puntero del área de memoria se pasa como parámetro en DS:DX.

argumentos	**valores devueltos**
ah = 0Ah	- número de caracteres leídos
ds:dx = puntero del área de memoria	- cadena

Ejemplo:

```
.DATA CADENA DB 11,?,11 DUP(?)
;RESERVA PARA UNA CADENA DE 10 CARACTERES

.CODE
MOV AX, @DATA
MOV DS, AX

LEA DX, CADENA
MOV AH, 0AH
INT 21H
```

Funciones avanzadas del teclado utilizando INT 16H

La interrupción 16h ayuda a manejar funciones avanzadas del teclado, recordemos que las teclado se hallan distribuido, por teclas alfanuméricas, de control, extendidas y de función.

Leer un carácter

Para realizar esta operación se utilizan los servicios 00 y 10.
El servicio 00 utiliza se utiliza para el manejo de sólo 83 teclas.

Registro	Valor	Retorna

Registro	Valor	Tipo de tecla
AH	Código de rastreo	Ascii normal
AL	Carácter ascii	
AH	Código de rastreo	Función extendida
AL	00	

El servicio 10 utiliza se utiliza para el manejo de sólo 101 teclas, acepta teclas de función extendida, teclado ampliado, teclas de control duplicadas.

Registro	Valor	Retorna

Registro	Valor	Tipo de tecla
AH	Código de rastreo	Ascii normal
AL	Carácter ascii	
AH	Código de rastreo	Función extendida
AL	00 o E0	

Para determinar si un usuario ha presionado una tecla de función extendida utilizar:

```
MOV AH,00
INT 16H
CMP AL, 00
JE COMPARA_RASTREO
...
```

```
MOV AH,10H
INT 16H
CMP AL, 00
JE COMPARA_RASTREO
CMP AL, E0H
JE COMPARA_RASTREO
...
```

Algunos códigos de rastreo:

Tecla	Rastreo	Ascii
Supr	53	00
Insert	52	00
End	4f	00
Flech abajo	50	00
Flecha Arrib	48	00
Flecha Izq	4B	00
Flecha Der	4D	00
Inicio	47	00
PgDn	51	00
PgUp	49	00

TABLA ASCII

Caracteres ASCII de control

00	NULL	(carácter nulo)
01	SOH	(inicio encabezado)
02	STX	(inicio texto)
03	ETX	(fin de texto)
04	EOT	(fin transmisión)
05	ENQ	(consulta)
06	ACK	(reconocimiento)
07	BEL	(timbre)
08	BS	(retroceso)
09	HT	(tab horizontal)
10	LF	(nueva línea)
11	VT	(tab vertical)
12	FF	(nueva página)
13	CR	(retorno de carro)
14	SO	(desplaza afuera)
15	SI	(desplaza adentro)
16	DLE	(esc.vínculo datos)
17	DC1	(control disp. 1)
18	DC2	(control disp. 2)
19	DC3	(control disp. 3)
20	DC4	(control disp. 4)
21	NAK	(conf. negativa)
22	SYN	(inactividad sínc)
23	ETB	(fin bloque trans)
24	CAN	(cancelar)
25	EM	(fin del medio)
26	SUB	(sustitución)
27	ESC	(escape)
28	FS	(sep. archivos)
29	GS	(sep. grupos)
30	RS	(sep. registros)
31	US	(sep. unidades)
127	DEL	(suprimir)

Caracteres ASCII imprimibles

32	espacio	64	@	96	`
33	!	65	A	97	a
34	"	66	B	98	b
35	#	67	C	99	c
36	$	68	D	100	d
37	%	69	E	101	e
38	&	70	F	102	f
39	'	71	G	103	g
40	(	72	H	104	h
41	)	73	I	105	i
42	*	74	J	106	j
43	+	75	K	107	k
44	,	76	L	108	l
45	-	77	M	109	m
46	.	78	N	110	n
47	/	79	O	111	o
48	0	80	P	112	p
49	1	81	Q	113	q
50	2	82	R	114	r
51	3	83	S	115	s
52	4	84	T	116	t
53	5	85	U	117	u
54	6	86	V	118	v
55	7	87	W	119	w
56	8	88	X	120	x
57	9	89	Y	121	y
58	:	90	Z	122	z
59	;	91	[	123	{
60	<	92	\	124	|
61	=	93	]	125	}
62	>	94	^	126	~
63	?	95	_		

ASCII extendido

128	Ç	160	á	192	└	224	Ó
129	ü	161	í	193	┴	225	ß
130	é	162	ó	194	┬	226	Ô
131	â	163	ú	195	├	227	Ò
132	ä	164	ñ	196	─	228	õ
133	à	165	Ñ	197	┼	229	Õ
134	å	166	ª	198	ã	230	µ
135	ç	167	º	199	Ã	231	þ
136	ê	168	¿	200	╚	232	Þ
137	ë	169	®	201	╔	233	Ú
138	è	170	¬	202	╩	234	Û
139	ï	171	½	203	╦	235	Ù
140	î	172	¼	204	╠	236	ý
141	ì	173	¡	205	═	237	Ý
142	Ä	174	«	206	╬	238	¯
143	Å	175	»	207	¤	239	´
144	É	176	░	208	ð	240	≡
145	æ	177	▒	209	Ð	241	±
146	Æ	178	▓	210	Ê	242	‗
147	ô	179	│	211	Ë	243	¾
148	ö	180	┤	212	È	244	¶
149	ò	181	Á	213	ı	245	§
150	û	182	Â	214	Í	246	÷
151	ù	183	À	215	Î	247	¸
152	ÿ	184	©	216	Ï	248	°
153	Ö	185	╣	217	┘	249	¨
154	Ü	186	║	218	┌	250	·
155	ø	187	╗	219	█	251	¹
156	£	188	╝	220	▄	252	³
157	Ø	189	¢	221	¦	253	²
158	×	190	¥	222	Ì	254	■
159	ƒ	191	┐	223	▀	255	nbsp

```
;-------------------------------------------------------------------------------
;Rastreo de la tecla DE FUNCION F12, USANDO INT 16H, SERVICIO 10
;-------------------------------------------------------------------------------

;Definición del Stack
STACKSG SEGMENT PARA STACK 'STACK'
  DB 20 DUP (0)
STACKSG ENDS

;Definición de áreas de trabajo
;µrea de Datos
DATASG SEGMENT PARA 'DATA'
  MEN DB 'Hola .....$'
DATASG ENDS

;Area de código
CODESG SEGMENT PARA 'CODE'
PRINCI PROC FAR
  ASSUME SS:STACKSG, DS:DATASG, CS:CODESG
  ;Protocolo
  PUSH DS
  SUB AX,AX
  PUSH AX
  MOV AX,SEG DATASG
  MOV DS,AX
  ;Inicia programa
  CALL limpia
  CALL PREINI

  RET
PRINCI ENDP

;Código de Procedimientos
LIMPIA PROC NEAR
      PUSH AX
      PUSH DX
      MOV AX,0600h
      MOV BH,71h
      MOV CX,0000h
      MOV DX,184Fh
      INT 10h
      POP DX
      POP AX
LIMPIA ENDP
```

```
PREINI PROC NEAR
      MOV AH,10H
      INT 16H
      CMP AL,00H
      JE RASTREA
      CMP AL,0E0H
      je RASTREA
      JMP SAL1
RASTREA:
      CMP AH,86h          ;Código de rastreo de la tecla F12
      JNE SAL1
      MOV AH,02
      MOV BH,00
      MOV DX,0c27h
      INT 10H
      LEA DX,MEN          ;Lee el mensaje
      MOV AH,09
      INT 21H
  SAL1: RET
PREINI ENDP

CODESG ENDS
END PRINCI
END
```