```
; ##################################
;       1Kpong! - 08/08/2005
;   version del pong clasico en 1kb
;   para MSX1 - codigo asMSX v0.11
;       Phoenix Software 1989
;
;  a mi niñachica, por soportar con
;  estoica paciencia mi pinpongpique
;
; este es el primer juego completo en
; ensamblador para MSX q "termino"...
; un poquito de poofavor a mis muchos
; errores. thks
; ##################################

   .bios
   .basic
   .org 0xC800
   .start main

; ------------------------------------------ CODE AREA
; ------------ INICIALIZACIONES
; --- definicion graficos, sprites, colores, etc (una vez)
main:
   ld a,rg_colborde
   ld [BDRCLR],a                    ; color borde
   xor a
   ld [CLIKSW],a                    ; click off
   inc a
   call CHGMOD                      ; screen 1
   ld bc,0xe001
   call WRTVDP                      ; vdp(226)=0 sprites 8*8

   xor a                            ; inicializa vram - bestial
   ld hl,tGraficos
   ld bc,14436
   call FILVRM
   ld a,rg_colfondo                 ; color "rolando guarros" (general)
   ld hl,tColores
   ld bc,32
   call FILVRM

   ld a,rg_punvacio                 ; color punto vacio
   ld hl,tColores+poff/8
   call VPOKE
   ld a,rg_punlleno                 ; color punto lleno
   ld hl,tColores+pon/8
   call VPOKE

   ld hl,datgra                     ; define graficos
   ld de,tGraficos+bhorizontal*8
   ld bc,8*3
   call LDIRVM
   ld hl,datpun
   ld de,tGraficos+poff*8
   ld bc,8
   push bc
   call LDIRVM
   ld hl,datpun+8
   ld de,tGraficos+pon*8
   pop bc
```

```
   call LDIRVM

   ld bc,5*8                        ; define sprites
   ld de,tSprites
   ld hl,datspr
   call LDIRVM

; --- inicializa pantalla, paletas, pelota, etc (con cada partida)
maingame:
   ld sp,0xe000                     ; reubica situacion pila
   call DISSCR                      ; desactiva pantalla

   ld a,bhorizontal                 ; dibuja bordes horizontales
   ld hl,tPatrones+32*1
   ld bc,32
   push af
   push bc
   call FILVRM
   pop bc
   pop af
   ld hl,tPatrones+32*22
   call FILVRM

   ld a,bizquierda                  ; dibuja bordes verticales
   ld b,20
   ld hl,tPatrones+15+32*2
   ld de,32
   push bc
   push hl
   call dibBorVert
   pop hl
   pop bc
   inc hl
   inc a
   call dibBorVert

   xor a
   ld [rebxpartida],a               ; reinicia rebotes x partida

   call iniciaMarcadores            ; reinicia marcadores
   call ENASCR                      ; activa pantalla

   call esperatecla                 ; reinicia pelota / espera tecla

; --- bucle partida
iniGame:
   call muevePaletas                ; ver desc. detallada de cada rutina alli
   call muevePelota
   call actuSprites
   jr iniGame

; ------------- SUBRUTINAS
; --- fin partida; nueva partida
tenemosGanador:
   ld hl,marcadores                 ; como la victoria es por 3-0
   ld a,[marcadores+1]              ; sumo marcadores y si = 3 entonces
   add a,[hl]                       ; un jugador (quien sea) gano la partida
   cp lavicky
   ret nz
@@gana:
   ld b,30                          ; efecto chorra "victoria"
@@bgana:
```

```
   call cambiaBordes
   djnz @@bgana
nuevoJuego:
   pop af                          ; extrae dir retorno de la pila
   jr maingame                     ; para volver al menu principal desde aqui

; --- comprueba choque pelota con paletas
; entrada:      b -> 1 paleta izquierda / 2 paleta derecha
;               ix -> puntero atributos paleta (sprite superior)
testChoque:
   ld a,[sprsBuff+33]              ; CX pelota
   djnz @@choqueD
   cp cxminpelota                  ; CX minima pelota
   ret nc
   jr @@testCY
@@choqueD:
   cp cxmaxpelota                  ; CX maxima pelota
   ret c
frontonON:                         ; salto aqui para modo fronton ON
;   jr choquePaleta
   nop                             ; 2xNOP para cambio de modo
   nop                             ; modo normal, dos jugadores
@@testCY:
   ld c,4                          ; pixels margen sup e inferior
   ld a,[ix+0]                     ; CY paleta - parte superior
   sbc a,c
   ld d,a
   ld a,[ix+12]                    ; CY paleta - parte inf
   add a,c
   ld e,a
   ld a,[sprsBuff+32]              ; CY pelota
   cp d
   jr c,marcaPunto                 ; CY pelota esta fuera del rango CY paleta
   cp e                            ; han marcado un punto
   jr nc,marcaPunto
   jr choquePaleta                 ; CY pelota esta dentro del rango CY paleta

; --- uno de los jugadores obtuvo un punto
marcaPunto:
   xor a
   cp b
   ld a,1                          ; a = 1, sgte punto contra jugador 2
   jr nz,puntoJ1
; --- marca punto jugador 2
   ld de,cpuntosJ2                 ; dir vram marcador J2
   ld iy,marcadores+1              ; marcador J2
   ld ix,marcadores                ; marcador J1
   jr marcaPuntos
; --- marca punto jugador 1
puntoJ1:
   neg                             ; a = -1, sgte punto contra jugador 1
   ld de,cpuntosJ1                 ; dir vram marcador J1
   ld iy,marcadores                ; marcador J1
   ld ix,marcadores+1              ; marcador J2
marcaPuntos:
   ld [espuntode],a                ; jugador que anoto (sgte saque en contra)
   xor a
   cp [ix+0]                       ; comprueba si el contrario tiene puntos
   call nz,iniciaMarcadores        ; y se los quita (si tuviera)
   inc [iy+0]
   ld b,3                          ; para dibujar los puntos actuales partimos
```

```
   ld hl,cmarcadores+9              ; de 3 (1-1-1 victoria)
@@bPuntos:
   ld a,[iy+0]                      ; marcadores (J1) o marcadores+1 (J2)
   cp b
   call z,dibMarcadores             ; rutina q dibuja los marcadores
   dec hl
   dec hl
   dec hl
   djnz @@bPuntos
   ld hl,cpoint                     ; sonido de punto
   call PLAY
   halt

; --- pausa hasta q se pulse una tecla
; "normalmente" llamada cdo se marca un punto
; ... por eso desde aqui se puede comprobar
; primero si existe un ganador, aunq no sea
; lo mas correcto del mundomundial :)
esperatecla:
   call tenemosGanador              ; comprueba si existe un ganador
   call iniciaPelota                ; reinicia posicion y datos pelota
   call KILBUF                      ; vacia el buffer de teclado
   halt
@@jartadesperar:                    ; espera hasta q se pulse cq tecla
   call CHSNS
   jr z,@@jartadesperar
   ret

; --- comprueba la parte de la paleta con la q choco la pelota
; entrada:      a  -> CY pelota
;               ix -> puntero atributos paleta (CY,CX..)
choquePaleta:
   push af
   ld a,[movpaleta]                 ; ver NOTAS (1) y (2)
   ld d,a
   pop af
   ld e,0
   cp [ix+12]                       ; CY borde inf paletas
   jr nc,@@sigue
   inc e
   cp [ix+8]                        ; CY centro inf paletas
   jr nc,@@sigue
   inc e
   cp [ix+4]                        ; CY centro sup paletas
   jr nc,@@sigue
   inc e
   ; e  -> 0=parte inf / 1=centro inf / 2=centro sup / 3=parte sup

; --- efectos asociados al choque de la pelota con las paletas
@@sigue:
   inc ix
   push ix
   push de
   push ix

   pop hl                           ; comienza efectos parpadeo + retroceso
   ld bc,0x040f                     ; b=4  - nº sprites paleta
@@bresta:                           ; c=15 - color parpadeo paleta
   dec [hl]
   dec [hl]                         ; realmente la paleta del J2 avanza,
   inc hl                           ; en lugar de retroceder. ver NOTA (2)
```

```
   inc hl
   ld [hl],c
   inc hl
   inc hl
   djnz @@bresta
   call actuSprites                ; actualiza sprites YAYAYA! para visualizar
                                   ; el efecto ahora mizmo. ver NOTA (2)
   ld iy,tabPelota                 ; tabla atributos pelota
   inc [iy+0]                      ; incrementa rebotes por jugada
   ld a,[iy+3]                     ; cambia el angulo X pelota
   neg
   ld [iy+3],a

   ld hl,rebxpartida               ; incrementa rebotes por partida
   inc [hl]

   pop de                          ; comprueba parte de la paleta con la
   xor a                           ; que choco la pelota (e) y reajuste
   cp e                            ; ultrabasico de los angulos
   jr z,bordesPaletas
   inc a
   cp e
   jr z,cenInfPaletas
   inc a
   cp e
   jr z,cenSupPaletas
   inc a
   cp e
   jr z,bordesPaletas

sigueChoque:
   pop hl                          ; termina efectos parpadeo + retroceso
   ld bc,0x040b                    ; b=4  - nº sprites paleta
@@bsuma:                           ; c=11 - color normal paleta
   inc [hl]
   inc [hl]                        ; realmente la paleta de J2 retrocede
   inc hl
   inc hl
   ld [hl],c
   inc hl
   inc hl
   djnz @@bsuma

   ld hl,cpong                     ; sonido del rebote
   call PLAY
   ret

; --- cambia angulo Y pelota segun choque con paleta
; y direccion del ultimo movimiento de cq paleta
; ver NOTAS (1) y (4)
; entrada:      d  -> dir ultimo mov de paleta, J1 o J2
;               iy -> tabla atributos pelota
; choque con los bordes de la paleta
bordesPaletas:                     ; choque con bordes sup e inf paletas
   ld a,d
   cp movarriba                    ; ult dir paletas fue arriba
   jr z,@@arribaBorde
   cp movabajo                     ; ult dir paletas fue abajo
   jr nz,sigueChoque

@@abajoBorde:                      ; si ult dir fue abajo y choca con bordes
   ld a,[iy+4]                     ; angulo Y = angulo Y + 1
```

```
   bit 7,a                          ; siempre sera positivo, para
   jr z,@@finAbajo                  ; rebote de la pelota hacia abajo
   neg
@@finAbajo:
   inc a
   ld [iy+4],a
   jr sigueChoque

@@arribaBorde:                      ; si ult dir fue arriba y choca con bordes
   ld a,[iy+4]                      ; angulo Y = -angulo Y - 1
   bit 7,a                          ; siempre sera negativo, para
   jr nz,@@finArriba                ; rebote de la pelota hacia arriba
   neg
@@finArriba:
   dec a
   jr sigueChoque

; choque con el centro superior de la paleta (al reves q la sgte)
cenSupPaletas:
   ld a,d
   cp movarriba                     ; ult dir paletas fue hacia arriba
   jr z,negativo
   cp movabajo                      ; ult dir paletas fue hacia abajo
   jr nz,sigueChoque

positivo:
   ld a,[iy+4]                      ; angulo Y = + angulo Y
   bit 7,a                          ; pelota rebotara hacia abajo
   jr z,@@volver
   neg
   jr @@volver

; choque con el centro inferior de la paleta (al reves q la anterior)
cenInfPaletas:
   ld a,d
   cp movarriba                     ; ult dir paletas fue hacia arriba
   jr z,positivo
   cp movabajo                      ; ult dir paletas fue hacia abajo
   jr nz,sigueChoque

negativo:
   ld a,[iy+4]                      ; angulo Y = - angulo Y
   bit 7,a                          ; pelota rebotara hacia arriba
   jr nz,@@volver
   neg
@@volver:
   ld [iy+4],a
   jr sigueChoque

; --- mueve paletas jugadores
muevePaletas:
   xor a
   ld [movpaleta],a                 ; dir ultimo mov efectuado (0=ninguno)
   call GTSTCK                      ; paleta J1
   ld de,sprsBuff                   ; atributos sprites paleta J1
   ld hl,movpaleta
   call nz,arriba
   ld a,1
   call GTSTCK                      ; paleta J2
   ld de,sprsBuff+16                ; atributos sprites paleta J2
   ld hl,movpaleta
```

```
   ret z
arriba:
   cp movarriba
   jr nz,abajo
   ld [hl],a                        ; dir ultimo mov = 1 -> arriba
   ld a,[de]                        ; pos CY actual paleta
   cp cyminpaleta                   ; para comprobar si llego al minimo
   ret z
   dec a
   dec a
   ld [de],a                        ; actualiza CY paleta (resta)
   ret
abajo:
   cp movabajo
   ret nz
   ld [hl],a                        ; dir ultimo mov = 5 -> abajo
   ld a,[de]                        ; pos CY actual paleta
   cp cymaxpaleta                   ; para ver si llego al maximo
   ret z
   inc a
   inc a
   ld [de],a                        ; actualiza CY paleta (suma)
   ret

; --- mueve pelota y paletas "sincronizadamente"... e un disir :)
muevePelota:
   ; actualizar CY paleta 1
   ld hl,sprsBuff
   call sincroPaletas
   ; actualizar CY paleta 2
   ld hl,sprsBuff+16
   call sincroPaletas

   ; comprueba choque pelota con las paletas
   ld b,1                           ; paleta J1 - izquierda
   ld ix,sprsBuff                   ; atributos paleta J1
   call testChoque
   ld b,2                           ; paleta J2 - derecha
   ld ix,sprsBuff+16                ; atributos paleta J2
   call testChoque

   ; incrementa dificultad / recalculos necesarios
   ld ix,tabPelota
   call reajustaDificultad

   ; actualizar angulo Y pelota / comprueba choque bordes
   ld hl,tabPelota+4                ; angulo Y
   ld a,[sprsBuff+32]               ; CY pelota
   add a,[hl]
   cp cyminpelota                   ; chequea limites verticales
   jr c,cambiaCY                    ; para cambiar direccion
   cp cymaxpelota
   jr nc,cambiaCY
   ld [sprsBuff+32],a               ; actualiza CY pelota

   ; actualizar angulo X pelota (sin choque paletas ni punto)
   ld hl,tabPelota+3                ; angulo X
   ld a,[sprsBuff+33]               ; CX pelota
   add a,[hl]
   ld [sprsBuff+33],a               ; actualiza CX pelota
   ret
```

```
   ; cambiar CY pelota; choque con bordes verticales
   ; entrada:   a -> CY pelota
cambiaCY:
   ld a,[hl]
   neg
   ld [hl],a
   call cambiaBordes               ; rutinilla cambio color bordes
   ld hl,cping
   call PLAY
   ret

   ; actualizar CY todas las partes de la paleta
   ; entrada:   hl -> atributos paleta (sprite superior)
sincroPaletas:
   ld b,4
   ld a,[hl]
@@bsincro:
   ld [hl],a
   add a,8
   inc hl
   inc hl
   inc hl
   inc hl
   djnz @@bsincro
   ret

; --- aumenta el nivel de dificultad (velocidad y ocultacion temporal)
; ver NOTA (3)
reajustaDificultad:
   ld a,[ix+0]                     ; nº de rebotes por jugada
   cp masvelocidad                 ; comprueba si cambia la velocidad
   jr nz,ocultacion
   xor a
   ld [ix+0],a                     ; resetea rebotes por jugada

   ld a,[ix+1]                     ; velocidad X
   cp maxvelocidad                 ; maxima velocidad
   ret nc
   inc [ix+1]                      ; incrementa velocidad X

   ld a,r                          ; pelin aleatoriedad
   and 00000001b
   ld [ix+2],a                     ; incrementa (o no) velocidad Y

   ld a,[ix+2]                     ; velocidad Y
   bit 7,[ix+4]                    ; angulo Y
   jr z,@@sumaAY
   neg
@@sumaAY:
   add [ix+4]                      ; (+-)velocidad + angulo
   ld [ix+4],a                     ; guarda nuevo angulo

   ld a,[ix+1]                     ; velocidad X
   bit 7,[ix+3]                    ; angulo X
   jr z,@@sumaAX
   neg
@@sumaAX:
   add [ix+3]
   ld [ix+3],a
   ret
```

```
ocultacion:
   cp ocultapelota                  ; comprueba si oculta la pelota
   ret nz
   inc [ix+0]                       ; incrementa con un rebote el contador de
   ld a,[ix+5]                      ; rebotes x jugada, para que no oculte
   ld [ix+6],a                      ; en esta iteraccion/velocidad la pelota
   ret                              ; reinicia tiempo de ocultacion

; --- reinicia posicion y datos pelota
iniciaPelota:
   ld hl,ctabPelota                 ; atributos pelota y paletas
   ld de,tabPelota
   ld bc,43
   ldir
   ld a,[espuntode]                 ; ultimo punto de J1 o J2
   ld [tabPelota+3],a

   ld a,r                           ; pos vertical sesudo aleatoria pelota
   and 11100000b
   add a,64
   ld [sprsBuff+32],a
   ld a,inicxpelota                 ; pos horizontal fija
   ld [sprsBuff+33],a

; --- actualiza sprites en pantalla
actuSprites:
   ld a,[tabPelota+6]               ; intervalo ocultacion pelota
   or a
   ld a,rg_inicopel                 ; color normal pelota (visible)
   jr z,@@actuSprs                  ; si intervalo=0, no oculta pelota
   ld hl,tabPelota+6                ; decrementa intervalo ocultacion
   dec [hl]
   ld a,rg_foncopel                 ; color fondo pelota (oculta)
@@actuSprs:
   ld hl,sprsBuff+35                ; byte color pelota
   ld [hl],a
   ld hl,sprsBuff
   ld de,tPatSprs
   ld bc,9*4
   halt
   call LDIRVM                      ; actualiza sprites
   ret

; --- reinicia marcadores
iniciaMarcadores:
   exx
   ld hl,cmarcadores                ; marcador 0-0-0
   ld de,cpuntosJ1                  ; dir vram marcador J1
   push hl
   call dibMarcadores               ; rutina para dibujar los marcadores
   pop hl
   ld de,cpuntosJ2                  ; dir vram marcador J2
   call dibMarcadores
   xor a
   ld [marcadores],a                ; resetea nº puntos J1 y J2
   ld [marcadores+1],a
   exx
   ret

; --- minirutina para dibujar un marcador
; entrada:      hl -> dir ram lectura
```

```
;               de -> dir vram escritura
dibMarcadores:
   push bc
   ld bc,3
   call LDIRVM
   pop bc
   ret

; --- mini rutina para dibujar los bordes verticales
; entrada:      hl -> dir vram escritura
;               de -> inc para sgte dir vram escritura
;               a -> chr a escribir
;               b -> nº de caracteres a escribir
dibBorVert:
   call VPOKE
   add hl,de
   djnz dibBorVert
   ret

; --- cambia color bordes tras choque pelota
cambiaBordes:
   exx
   ld a,rg_copborde                 ; color parpadeo
   call bordes
bordesAmarillos:
   exx
   ld a,rg_colfondo                 ; color normal
bordes:
   ld hl,tColores+bhorizontal/8     ; dir vram escritura
   halt
   call VPOKE
   halt
   exx
   ret

; ------------------------------------------- CTES AREA
; --- ram / vram / bios
BDRCLR       equ 0xf3eb             ; color del borde
CLIKSW       equ 0xf3db             ; click off

tGraficos    equ 0                  ; 2048 bytes
tPatrones    equ 6144               ; 768 bytes
tPatSprs     equ 6912               ; 1024 bytes
tColores     equ 8192               ; 32 bytes = 8 chrs * 1 byte color
tSprites     equ 14336              ; 2048 bytes

cpuntosJ1    equ 6144+2
cpuntosJ2    equ 6144+27+32*23

PLAY         equ 0x73e5             ; rutina PLAY basic
VPOKE        equ 0x4d               ; rutina VPOKE basic

; --- juego
bhorizontal  equ 65                 ; chr borde horizontal sup / inf
bizquierda   equ 66                 ; chr borde vertical izq
bderecha     equ 67                 ; chr borde vertical der
poff         equ 72                 ; chr punto vacio
pon          equ 80                 ; chr punto ganado

movarriba    equ 1                  ; cursor arriba
movabajo     equ 5                  ; cursor abajo
```

```
cyminpaleta  equ 14                ; CY minima paletas
cymaxpaleta  equ 144               ; CY maxima paletas

inicxpelota  equ 112               ; pos CX inicio pelota
cyminpelota  equ 11                ; CY minima (choque bordes) pelota
cymaxpelota  equ 172               ; CY maxima (choque bordes) pelota
cxminpelota  equ 10                ; CX minima (choque paletas) pelota
cxmaxpelota  equ 239               ; CX maxima (choque paletas) pelota

; colorido a lo "rolando guarros" :)
rg_colborde  equ 0x08              ; color borde pantalla
rg_colfondo  equ 0xb6              ; color fondo terreno juego
rg_punvacio  equ 0x98              ; color punto vacio
rg_punlleno  equ 0xf8              ; color punto lleno
rg_copborde  equ 0xf6              ; color parpadeo bordes
rg_inicopel  equ 0x0f              ; color inicio pelota
rg_foncopel  equ 0x06              ; color ocultacion pelota
rg_inicopal  equ 0x0b              ; color inicio paleta

lavicky      equ 3                 ; puntos de victoria

; DOS unidades de diferencia entre rebotes para inc velocidad
; y rebotes para ocultacion de la pelota, pq al ocultar la pelota
; se suma UNA unidad a rebotes para inc de velocidad. ver NOTA (3)
maxvelocidad equ 3                 ; maxima velocidad
masvelocidad equ 6                 ; num rebotes por jugada para inc velocidad pelota
ocultapelota equ 3                 ; num rebotes por jugada para ocultacion pelota

; --- graficos
datspr:
   db 0x3c,0x7e,0x7e,0x7e,0x7e,0x7e,0x7e,0x7e          ; paleta arriba
   db 0x7e,0x7e,0x7e,0x7e,0x7e,0x7e,0x7e,0x3c          ; paleta abajo
   db 0x7e,0x7e,0x7e,0x7e,0x7e,0x7e,0x7e,0x7e          ; centro1
   db 0x7e,0x7e,0x7e,0x7e,0x7e,0x7e,0x7e,0x7e          ; centro2
   db 0x0,0x3c,0x7e,0x7e,0x7e,0x7e,0x3c,0x0            ; pelota

datgra:
   db 0x0,0x0,0x7C,0xfe,0xfe,0x7c,0x0,0x0              ; borde vertical
   db 0x1,0x3,0x3,0x3,0x3,0x3,0x1,0x0                  ; borde superior
   db 0x80,0xc0,0xc0,0xc0,0xc0, 0xc0,0x80,0x0          ; borde inferior

datpun:
   db 0x18,0x24,0x42,0x81,0x81,0x42,0x24,0x18          ; punto vacio
   db 0x18,0x7e,0x7e,0xff,0xff,0x7e,0x7e,0x18          ; punto lleno

; --- sonidos
cping:
   db $22,"T255O6G-",$22,0
cpong:
   db $22,"T200O5C-",$22,0
cpoint:
   db $22,"O3EDC",$22,0

; --- marcadores
cmarcadores:                       ; posibles marcadores J1,J2
   db poff, poff, poff             ; 0-0-0
   db pon, poff, poff              ; 1-0-0
   db pon, pon, poff               ; 1-1-0
   db pon, pon, pon                ; 1-1-1 = victoria

; pelota y paletas
; *** no separar / no desordenar ***
```

```
ctabPelota:
   db 0                          ; rebotes x jugada         +0
   db 0,0                        ; velocidad X, Y           +1,+2
   db -1,-1                      ; angulos X, Y             +3,+4
   db 35,0                       ; ocultacion               +5,+6
cinisprs:
   db 128,4,0,rg_inicopal        ; paleta J1                +0...
   db 128+8,4,2,rg_inicopal      ;                          +4...
   db 128+8*2,4,3,rg_inicopal    ;                          +8...
   db 128+8*3,4,1,rg_inicopal    ;                          +12...
   db 32,244,0,rg_inicopal       ; paleta J2                +16...
   db 32+8,244,2,rg_inicopal     ;                          +20...
   db 32+8*2,244,3,rg_inicopal   ;                          +24...
   db 32+8*3,244,1,rg_inicopal   ;                          +28...
   db 96,inicxpelota,4,rg_inicopel  ; pelota                +32...
; *** no separar / no desordenar ***

; ------------------------------------------ VAR AREA
espuntode:   db -1               ; ultimo punto, 1=jugador2 / -1=jugador1

rebxpartida: ds 1                ; rebotes por partida
marcadores:  ds 2                ; marcador jugador1, marcador jugador2
movpaleta:   ds 1                ; ultima dir movimiento ultima paleta movida

; *** no separar / no desordenar / no situar nada debajo ***
tabPelota:
             ds 1                ; rebotes por jugada; para inc dificultad
             ds 2                ; velocidad X, Y
             ds 2                ; angulos X, Y
             ds 2                ; iteraciones ocultacion; para inc dificultad
sprsBuff:
             ds 36               ; tabla de atributos de Sprites en ram, 36 bytes
; *** no separar / no desordenar / no situar nada debajo ***

/*
   NOTAS:

   (1) el ultimo movimiento de cualquier paleta
   influye en el angulo de devolucion de la pelota
   tras un choque con cualquiera de las dos paletas,
   pero solo de un modo muy basico q deberia mejorarse;
   ademas deberia guardarse el movpaleta de cada pala.
   como queda ahora el mov de la paleta del J2 podria
   influir en el angulo de devolucion de la pelota cdo
   choca con la paleta del J1, lo q no tie mucha logica
   pero resulta curioso&pasable :)

   (2) nota generica para algo q no me parece correcto
   pero q, mas o menos, funciona bien... o produce efectos
   *curiosos* :)

   (3) el contador de rebotes por jugada se utiliza
   para incrementar la velocidad de la pelota y para
   ocultarla temporalmente; mas correcto seria utilizar
   un contador de rebotes por jugada para cada funcion.
   pero como el contador es comun hay q tener cuidado al
   ajustar los limites de cada funcion. el num de rebotes
   para incrementar la velocidad debe ser mayor q el num
   de rebotes para ocultar la pelota, min en 2 unidades

   (4) la idea era q el rebote de la pelota fuese controlable
   -hasta cierto punto- y q incluso se pudiera hacer
```

disminuir la velocidad de la pelota (q aumenta con el num
de rebotes) acertandola con las partes centrales de las
paletas... pero gueno: todo se ha quedado en idea -y en ese
codigo tan enrevesado- por falta habilidad y espacio para
desarrollarlo mejor... sorry :(

tb se quedan en el tintero:
- modo "panico" (fronton 1J) con contador nº rebotes totales
mostrados al final del juego. el contador existe pero no
se muestra nada al final del juego
- ordenacion de los datos de los colores para permitir
un cambio de colorido del juego lo mas sencillo posible
- mejoras sustanciales en las rutinas de incremento de dificultad,
ocultacion y cambio de angulos segun choque con partes de las
paletas
- un monton de bugs sin arreglar, SURE!!!

EXTRAS:

activar modo "panico" (fronton un jugador):

```
10 bload"kpong.bin"
20 poke &HC8D1,&H18
30 poke &HC8D2,&H6C
40 poke &HCBE4,5
50 poke &HCBE8,5
60 poke &HCBEC,5
70 poke &HCBF0,5
100 defusr=&Hc800:a=usr(0)
```

cambiar esquema de colores:

```
-otro modelo un tanto psicodelico, en verdes,
amarillo y blanco-
10 BLOAD"kpong.bin"
20 REM color-parpadeo-paleta
30 POKE &HC962,&HF
40 REM color-normal-pelota
50 POKE &HCBF5,&HF
60 POKE &HCAF8,&HF
70 REM color-oculta-pelota
80 POKE &HCB00,&HC
90 REM color-parpadeo-bordes
100 POKE &HCB40,&HFC
110 REM color-normal-bordes
120 POKE &HCB46,&HBC
130 REM color-normal-paleta
140 POKE &HCBD5,&HB
150 POKE &HCBD9,&HB
160 POKE &HCBDD,&HB
170 POKE &HCBE1,&HB
180 POKE &HCBE5,&HB
190 POKE &HCBE9,&HB
200 POKE &HCBED,&HB
210 POKE &HCBF1,&HB
220 POKE &HC996,&HB
230 REM color-terreno-juego
240 POKE &HC81E,&HBC
250 REM color-punto-vacio
260 POKE &HC829,&HC2
270 REM color-punto-lleno
280 POKE &HC831,&HF2
```

```
   290 REM color-bordes-pantalla
   300 POKE &HC801,&H2
   310 DEFUSR=&HC800:A=USR(0)

   -esquema bte agradable: azules, blanco y gris-
   10 BLOAD"kpong.bin"
   20 REM color-parpadeo-paleta
   30 POKE &HC962,&HF
   40 REM color-normal-pelota
   50 POKE &HCBF5,&HE
   60 POKE &HCAF8,&HE
   70 REM color-oculta-pelota
   80 POKE &HCB00,&H4
   90 REM color-parpadeo-bordes
   100 POKE &HCB40,&HF4
   110 REM color-normal-bordes
   120 POKE &HCB46,&H74
   130 REM color-normal-paleta
   140 POKE &HCBD5,&HE
   150 POKE &HCBD9,&HE
   160 POKE &HCBDD,&HE
   170 POKE &HCBE1,&HE
   180 POKE &HCBE5,&HE
   190 POKE &HCBE9,&HE
   200 POKE &HCBED,&HE
   210 POKE &HCBF1,&HE
   220 POKE &HC996,&HE
   230 REM color-terreno-juego
   240 POKE &HC81E,&H74
   250 REM color-punto-vacio
   260 POKE &HC829,&H45
   270 REM color-punto-lleno
   280 POKE &HC831,&HE5
   290 REM color-bordes-pantalla
   300 POKE &HC801,&H5
   310 DEFUSR=&HC800:A=USR(0)
*/
```