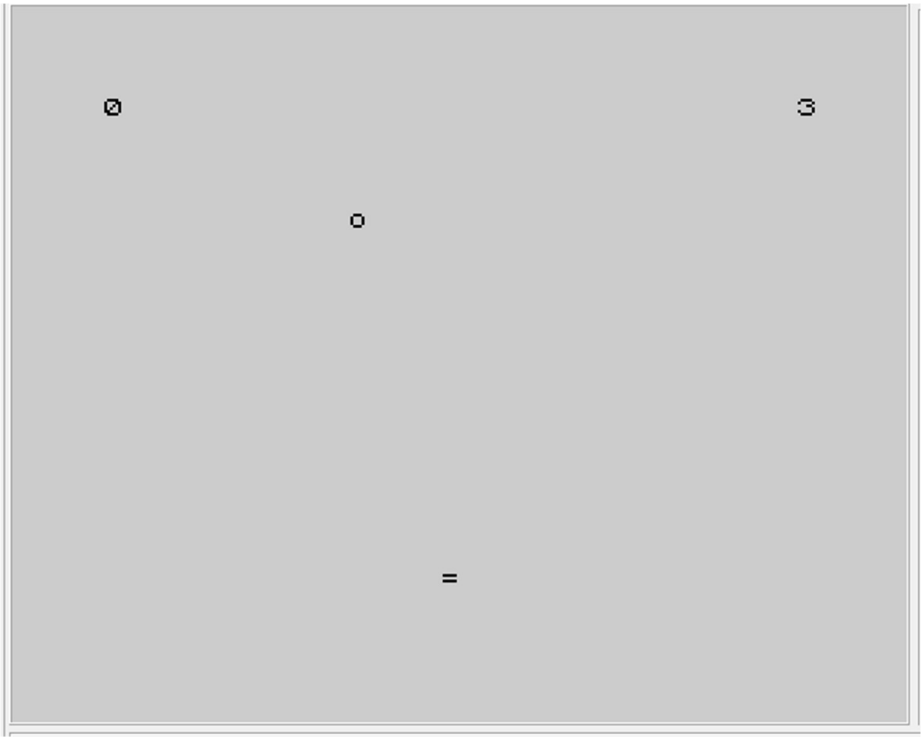
PROYECTO 2: PROGRAMACIÓN ENSAMBLADOR

Inspirado en el clásico “Pong”, se desea implementar un juego para el ZX Spectrum en el que una pelota rebote por las paredes de la pantalla, y el jugador tenga que impedir que caiga al suelo moviendo una pala a lo largo de la pared inferior.



ESPECIFICACIONES:

- Se trabajará en modo de baja resolución: pantalla de 24x32 caracteres (cada carácter formado por 8 octetos).
- La pala se situará en la fila 21 y podrá moverse horizontalmente con las teclas ‘I’ y ‘P’ para poder impedir que se caiga la pelota.
- El juego comenzará apareciendo el nombre de éste (“Vertical Pong”) en el centro de la pantalla.
- Los límites del campo son las filas 1 y 20 y las columnas 0 y 31.

- En la parte superior izquierda de la pantalla (línea 0) aparecerá un marcador con el número de veces que el jugador ha devuelto la bola (rebotes/puntos). En la parte superior derecha de la pantalla aparecerá un marcador con el número de vidas restantes.
- Al iniciarse el juego, la pelota aparecerá siempre en la fila 10.
- El juego termina cuando la bola toque la pared inferior de la pantalla 3 veces (hay 3 “vidas”).
- Al finalizar el juego, se mostrará un mensaje de “Game Over”.
- Cuando el usuario pulse cualquier tecla del teclado, comenzará una nueva partida.

Puntuación de la práctica:

1) **(1 punto)** Inicialización. Aparición inicial del título. Puesta a 0 del contador de impactos y puesta a 3 del contador de vidas.

2) **(2 puntos)** Rutina de movimiento de la pala.

3) **(2 puntos)** Rutina de movimiento de la pelota. La pelota partirá de una columna aleatoria de la pantalla.

4) **(2 puntos)** Rebotes de la pelota. Detectar los rebotes en las paredes laterales y superior y cambiar el sentido de la marcha. Detectar el rebote en la pala y cambiar el sentido de la marcha.

5) **(1 punto)** Actualizar el marcador de puntos cada vez que la pelota rebote en la pala.

6) **(1 punto)** Actualizar el marcador de vidas cada vez que la pelota toca la parte inferior de la pantalla (no consigue pararla la pala) y volver a comenzar. Terminar la partida cuando se acaban las vidas.

7) **(1 punto)** Claridad y limpieza del código, inclusión de comentarios que aclaren su funcionamiento.

Normativa de realización, entrega y evaluación de la práctica:

- **La práctica se realizará y entregará en grupos de 3. La calificación obtenida será idéntica para todos los miembros del grupo.**
- **El apartado 7 sólo puntuará si hay algún otro apartado puntuado.**
- **Cada apartado se puntúa como un todo, esto es, sólo si el apartado al completo está bien realizado.**
- **La práctica se realizará únicamente mediante el software Zeus.**
- **La entrega deberá hacerse mediante Blackboard antes del 20 de mayo de 2019 a las 23:59 horas (hora peninsular en España).**
- **La entrega se compondrá de un único fichero que contendrá el circuito en formato Zeus (extensión .ASM).**
- **El nombre del fichero será el nombre y primer apellido de los alumnos que lo entregan, escritos en letras mayúsculas y separados mediante un guión bajo '_'. Ejemplo: JOSE_FERNANDEZ_ISABEL_MARTINEZ_ LAURA_PALMER.ASM**
- **Cualquier sospecha de copia entre dos o más prácticas derivará en la calificación de 0 para todos los alumnos involucrados.**

ANEXO

- Para pintar en la pantalla utilizaremos la llamada al sistema PRINT (call PRINT). Para utilizar dicha función se debe:

  1.- Crear dos etiquetas con la dirección de las funciones CHANOPEN y PRINT.

```
CHANOPEN                 equ 5633
PRINT                   equ 8252
```

  2.- Llamar a CHANOPEN con el parámetro 2, que indica, utilizar la pantalla principal. Una vez.

```
ld a, 2
call CHANOPEN
```

  3.- Cada vez que vayas a imprimir por pantalla, rellenar los parámetros de la función y llamar a `PRINT.`

```
ld de, parametros
ld bc, 4
call PRINT
```

  donde parámetros:

```
parametros  defb 22, fila, columna, "caracter a pintar"
```

- Para incluir aleatoriedad, puede utilizarse el valor del registro R, ya que es el refresco de la memoria dinámica y va cambiando rápidamente.

- Una máscara (aplicar una instrucción and con un valor determinado) nos puede servir para quedarnos con ciertos bits de un registro (poniendo a cero los otros).

- El código ASCII del 0 es el 48