```
# Fantasma.s
# Estructura que representa el personaje enemigo del juego.
#
# Autores: Ka Fung & Christopher Gomez
# Fecha:   10-ene-2022

        .data

        .text

# Funcion: Crea un Fantasma con su posicion y color.
# Entrada:  $a0:  Coordenada x.
#           $a1:  Coordenada y.
#           $a2:  Color del fantasma.
# Salida:   $v0:  Fantasma (negativo si no se pudo crear).
#          ($v0): Direccion del Bitmap Display.
#         4($v0): Color del Fantasma.
#         8($v0): Color de la capa de fondo.
#        12($v0): Dir. de movimiento del fantasma.
#                 (0: Izquierda, 1: Arriba, 2: Abajo, 3: Derecha)
# Planificacion de registros:
# $s0: Color del fantasma.
# $t0: Coordenada x del fantasma.
# $t1: Auxiliar.
Fantasma_crear:
    # Prologo
    sw   $fp,   ($sp)
    sw   $ra, -4($sp)
    sw   $s0, -8($sp)
    move $fp,    $sp
    addi $sp,    $sp, -12

    move $s0, $a2

    # Obtiene direccion del fantasma
    jal coord_a_dir_bitmap
    move $t0, $v0

    # Reserva memoria para el fantasma
    li $v0, 9
    li $a0, 13
    syscall
    bltz $v0, Fantasma_crear_fin

    # Inicializa el fantasma
    sw $t0,   ($v0)       # Direccion
    sw $s0,  4($v0)       # Color del fantasma
    lw $t1, colorComida
    sw $t1,  8($v0)       # Color capa de fondo
    li $t1,  3
    sb $t1, 12($v0)       # Direccion de movimiento

Fantasma_crear_fin:
    # Epilogo
    move $sp,    $fp
    lw   $fp,   ($sp)
    lw   $ra, -4($sp)
    lw   $s0, -8($sp)

    jr $ra

# Funcion: Reinicia el fantasma con su posicion inicial.
# Entrada:  $a0:  Fantasma
#           $a1:  Coordenada x
#           $a2:  Coordenada y
# Planificacion de registros:
# $s0: Fantasma.
# $s1: Dir. del fantasma actual en el Bitmap Display
# $t0: Auxiliar.
# $t1: Dir. del Bitmap Display del fantasma.
Fantasma_reiniciar:
    # Prologo
    sw   $fp,    ($sp)
    sw   $ra,  -4($sp)
    sw   $s0,  -8($sp)
    sw   $s1, -12($sp)
    sw   $s2, -16($sp)
    move $fp,    $sp
    addi $sp,    $sp, -20

    move $s0,   $a0
    lw   $s1,  ($s0)
    lw   $s2, 8($s0)

    # Obtiene y actualiza direccion del fantasma
    move $a0, $a1
    move $a1, $a2
    jal  coord_a_dir_bitmap
    sw   $v0, ($s0)

    # Actualiza el color del fondo en la dir. nueva del fantasma
    lw $t0, colorComida
    sw $t0, 8($s0)

    # Si en la anterior partida, Pac-Man fue comido
    # Pintar de color del fondo donde estaba el fantasma
```

```
    lb   $t1, fueComido
    beqz $t1, Fantasma_reiniciar_actualizar_dir

    beq $s2, $t0, Fantasma_reiniciar_pintar_fondo_anterior
    lw  $t0, colorFondo
    Fantasma_reiniciar_pintar_fondo_anterior:
        sw  $s2, ($s1)

        # Actualiza el color del fondo en la dir. nueva del fantasma
        lw  $t0, ($v0)
        lw  $t1, colorComida
        beq $t0, $t1, Fantasma_reiniciar_pintar_fondo_actual

        # Evita tomar como fondo algun color de fantasma
        lw  $t0, colorFondo
    Fantasma_reiniciar_pintar_fondo_actual:
        sw  $t0, 8($s0)
    Fantasma_reiniciar_actualizar_dir:
        # Pinta el fantasma en la nueva direccion
        lw $t0, 4($s0)  # Color del fantasma
        sw $t0,  ($v0)

Fantasma_reiniciar_fin:
    # Epilogo
    move $sp,     $fp
    lw   $fp,    ($sp)
    lw   $ra,  -4($sp)
    lw   $s0,  -8($sp)
    lw   $s1, -12($sp)
    lw   $s2, -16($sp)
    jr $ra

# Funcion: Se encarga del movimiento del fantasma por el tablero, tomando
#          en cuenta las intersecciones y colisiones.
# Entrada: $a0: Fantasma.
# Planificacion de registros:
# $s0: Fantasma
# $t0: Auxiliar
Fantasma_mover:
    # Prologo
    sw   $fp,    ($sp)
    sw   $ra,  -4($sp)
    sw   $s0,  -8($sp)
    move $fp,     $sp
    addi $sp,     $sp, -12

    # Guarda fantasma
    move $s0, $a0

    # Verifica si se encuentra en una interseccion
    jal Fantasma_chequear_interseccion
    beqz $v0, Fantasma_mover_revisar_sig

    # Escoge direccion aleatoria si es una interseccion
    move $a0, $s0
    jal  Fantasma_cambiar_dir

    j Fantasma_mover_ejecutar

    Fantasma_mover_revisar_sig:
        # Guarda la direccion del Bitmap Display del fantasma
        lw $a0, ($s0)

        # Revisa la direccion actual del fantasma
        lb   $t0, 12($s0)
        beqz $t0, Fantasma_mover_revisar_sig_izq
        beq  $t0, 1, Fantasma_mover_revisar_sig_arriba
        beq  $t0, 2, Fantasma_mover_revisar_sig_abajo

        # Revisa siguiente direccion derecha
            jal obtener_dir_derecha
            j   Fantasma_mover_chequear_colision
        Fantasma_mover_revisar_sig_izq:
            jal obtener_dir_izquierda
            j   Fantasma_mover_chequear_colision

        Fantasma_mover_revisar_sig_arriba:
            jal obtener_dir_arriba
            j   Fantasma_mover_chequear_colision
        Fantasma_mover_revisar_sig_abajo:
            jal obtener_dir_abajo
    Fantasma_mover_chequear_colision:
        move $a0, $v0
        jal  Fantasma_es_camino
        beqz $v0, Fantasma_mover_ejecutar

        # Si hay alguna colision, se busca una nueva dir. aleatoria
        move $a0, $s0
        jal  Fantasma_cambiar_dir

    Fantasma_mover_ejecutar:
        move $a0, $s0
        jal  Fantasma_ejecutar_mov
Fantasma_mover_fin:
    # Epilogo
```

```
    move $sp,    $fp
    lw   $fp,    ($sp)
    lw   $ra,  -4($sp)
    lw   $s0,  -8($sp)

    jr $ra

# Funcion: Mueve el pixel del fantasma a la direccion especificada
#          en la estructura.
# Entrada:  $a0:  Fantasma.
# Planificacion de registros:
# $s0: Fantasma.
# $s1: Direccion del Bitmap Display siguiente del fantasma.
# $t0: Auxiliar.
# $t1: Color de Pac-Man / Fondo
Fantasma_ejecutar_mov:
    # Prologo
    sw   $fp,    ($sp)
    sw   $ra,  -4($sp)
    sw   $s0,  -8($sp)
    sw   $s1, -12($sp)
    move $fp,     $sp
    addi $sp,     $sp, -16

    move $s0,  $a0      # Fantasma
    lw   $a0, ($s0)     # Dir. del fantasma

    # Revisa la direccion actual del fantasma
    lb   $t0, 12($s0)
    beqz $t0, Fantasma_ejecutar_mov_izq
    beq  $t0, 1, Fantasma_ejecutar_mov_arriba
    beq  $t0, 2, Fantasma_ejecutar_mov_abajo

    # Direccion derecha
    jal obtener_dir_derecha
    j   Fantasma_ejecutar_mov_verif_portal
Fantasma_ejecutar_mov_izq:
    jal obtener_dir_izquierda
    j   Fantasma_ejecutar_mov_verif_portal

Fantasma_ejecutar_mov_arriba:
    jal obtener_dir_arriba
    j   Fantasma_ejecutar_mov_verif_portal
Fantasma_ejecutar_mov_abajo:
    jal obtener_dir_abajo
Fantasma_ejecutar_mov_verif_portal:
    # Guarda la direccion siguiente del fantasma
    move $s1, $v0

    # Verifica si hay un portal
    lw  $t0, colorPortal
    lw  $t1, ($v0)
    bne $t0,  $t1,  Fantasma_ejecutar_mov_pintar

    # Portal 6 (0, 18)
    li  $a0, 0
    li  $a1, 18
    jal coord_a_dir_bitmap
    beq $v0, $s1, Fantasma_ejecutar_mov_portal_der

    # Portal 6 (0, 17)
    add $t1, $v0, 128
    beq $t1, $s1, Fantasma_ejecutar_mov_portal_der

    # De otra forma, se trata del portal 5 (31, 17) o (31, 18)
    # Mueve el Fantasma al portal izquierdo
    add $s1, $s1, -120     # DIRSIGUIENTE = DIRACTUAL - 30*4
    j   Fantasma_ejecutar_mov_pintar

    Fantasma_ejecutar_mov_portal_der:
        # Mueve el Fantasma al portal derecho
        add $s1, $s1, 120  # DIRSIGUIENTE = DIRACTUAL + 30*4

Fantasma_ejecutar_mov_pintar:
    # Verificamos que la siguiente posicion es camino
    # Si el camino no es valido, no se mueve el fantasma
    move $a0, $s1
    jal  Fantasma_es_camino
    bnez $v0, Fantasma_ejecutar_mov_fin

    # En cambio, se mueve fantasma
    # Pinta (x, y) del color de la capa de fondo del fantasma
    lw $t0,  ($s0)  # Dir. actual del fantasma
    lw $t1, 8($s0)  # Color del fondo
    sw $t1,  ($t0)

    # Actualiza fondo del fantasma
    lw $t0,  ($s1)  # Color del fondo de la dir. siguiente
    sw $t0, 8($s0)

    # Actualiza nueva direccion del fantasma
    sw $s1, ($s0)

    # Revisa si el fantasma se ha comido a Pac-Man
    lw  $t0, ($s1)
    lw  $t1, colorPacman
    bne $t0, $t1, Fantasma_ejecutar_mov_pintar_nuevo
```

```
        lw $t0, colorFondo
        sw $t0, 8($s0)

        li $t1, 1
        sb $t1, fueComido

Fantasma_ejecutar_mov_pintar_nuevo:
    # Pinta el fantasma en la nueva direccion
    lw $t1, 4($s0)  # Color del fantasma
    sw $t1,  ($s1)
Fantasma_ejecutar_mov_fin:
    # Epilogo
    move $sp,     $fp
    lw   $fp,    ($sp)
    lw   $ra,  -4($sp)
    lw   $s0,  -8($sp)
    lw   $s1, -12($sp)

    jr $ra

# Funcion: Verifica si un fantasma se encuentra en una interseccion.
# Entrada: $a0: Fantasma.
# Salida:  $v0: 1 si se encuentra en una interseccion,
#               0 de otra forma.
# Planificacion de registros:
# $s0: Direccion del Bitmap Display
# $s3: colorPared
# $t0: Color del pixel que se chequea
# $t1: Booleanos auxiliares
# $t2: Booleanos auxiliares
Fantasma_chequear_interseccion:
# Prologo
sw   $fp,    ($sp)
sw   $ra,  -4($sp)
sw   $s0,  -8($sp)
sw   $s1, -12($sp)
move $fp,     $sp
addi $sp,     $sp, -16

# Guarda la direccion del fantasma
lw $s0, ($a0)

# Guarda los colores a comparar
lw $s1, colorPared

# La condicion de interseccion consiste en chequear que
# las adyacencias sean caminos y la diagonal una pared

# Verifica si (x+1, y+1) es pared / Noreste
move $a0, $s0
jal obtener_dir_noreste
lw  $t0, ($v0)
bne $t0,  $s1, Fantasma_chequear_interseccion_der_abajo

# Verifica si (x, y+1) es camino / Arriba
move $a0, $s0
jal  obtener_dir_arriba

move $a0, $v0
jal  Fantasma_es_camino
bnez $v0, Fantasma_chequear_interseccion_der_abajo

# Verifica si (x+1, y) es camino / Derecha
move $a0, $s0
jal  obtener_dir_derecha

move $a0, $v0
jal  Fantasma_es_camino
bnez $v0, Fantasma_chequear_interseccion_abajo_izq

j Fantasma_chequear_interseccion_retornar_cierto
Fantasma_chequear_interseccion_der_abajo:
    # Verifica si (x+1, y-1) es pared / Sureste
    move $a0, $s0
    jal obtener_dir_sureste
    lw  $t0, ($v0)
    bne $t0,  $s1, Fantasma_chequear_interseccion_abajo_izq

    # Verifica si (x+1, y) es camino / Derecha
    move $a0, $s0
    jal  obtener_dir_derecha

    move $a0, $v0
    jal  Fantasma_es_camino
    bnez $v0, Fantasma_chequear_interseccion_abajo_izq

    # Verifica si (x, y-1) es camino / Abajo
    move $a0, $s0
    jal  obtener_dir_abajo

    move $a0, $v0
    jal  Fantasma_es_camino
    bnez $t1, Fantasma_chequear_interseccion_izq_arriba

    j Fantasma_chequear_interseccion_retornar_cierto
Fantasma_chequear_interseccion_abajo_izq:
```

```
    # Verifica si (x-1, y-1) es pared / Suroeste
    move $a0,  $s0
    jal  obtener_dir_suroeste
    lw   $t0, ($v0)
    bne  $t0,  $s1, Fantasma_chequear_interseccion_izq_arriba

    # Verifica si (x, y-1) es camino / Abajo
    move $a0, $s0
    jal  obtener_dir_abajo

    move $a0, $v0
    jal  Fantasma_es_camino
    bnez $v0, Fantasma_chequear_interseccion_izq_arriba

    # Verifica si (x-1, y) es camino / Izquierda
    move $a0, $s0
    jal  obtener_dir_izquierda

    move $a0, $v0
    jal  Fantasma_es_camino
    bnez $v0, Fantasma_chequear_interseccion_retornar_falso

    j Fantasma_chequear_interseccion_retornar_cierto
Fantasma_chequear_interseccion_izq_arriba:
    # Verifica si (x-1, y+1) es pared / Noroeste
    move $a0,  $s0
    jal  obtener_dir_noroeste
    lw   $t0, ($v0)
    bne  $t0,  $s1, Fantasma_chequear_interseccion_retornar_falso

    # Verifica si (x-1, y) es camino / Izquierda
    move $a0, $s0
    jal  obtener_dir_izquierda

    move $a0, $v0
    jal  Fantasma_es_camino
    bnez $v0, Fantasma_chequear_interseccion_retornar_falso

    # Verifica si (x, y+1) es camino / Arriba
    move $a0, $s0
    jal  obtener_dir_arriba

    move $a0, $v0
    jal  Fantasma_es_camino
    bnez $v0, Fantasma_chequear_interseccion_retornar_falso

    j Fantasma_chequear_interseccion_retornar_cierto
Fantasma_chequear_interseccion_retornar_falso:
    li $v0, 0
    j  Fantasma_chequear_interseccion_fin
Fantasma_chequear_interseccion_retornar_cierto:
    li $v0, 1
Fantasma_chequear_interseccion_fin:
    # Epilogo
    move $sp,     $fp
    lw   $fp,    ($sp)
    lw   $ra,  -4($sp)
    lw   $s0,  -8($sp)
    lw   $s1, -12($sp)

    jr $ra

# Funcion: Escoge aleatoriamente una direccion valida para el siguiente
#          movimiento del fantasma. Solo retorna la direccion contraria
#          a la actual si es la unica disponible
# Entrada: $a0: Fantasma.
# Salida:  $v0: Dir. aleatoria valida de movimiento del fantasma.
#               (0: Izquierda, 1: Arriba, 2: Abajo, 3: Derecha)
# Planificacion de registros:
# $s0: Fantasma
# $s1: Direccion del Bitmap Display del fantasma
# $s2: Direccion opuesta al movimiento del fantasma
# $s3: Contador de direcciones disponibles / Auxiliar
# $s4: Dir. inicial de la pila donde se guarda dir. de movimientos validos
# $t0: Auxiliar
Fantasma_cambiar_dir:
    # Prologo
    sw   $fp,    ($sp)
    sw   $ra,  -4($sp)
    sw   $s0,  -8($sp)
    sw   $s1, -12($sp)
    sw   $s2, -16($sp)
    sw   $s3, -20($sp)
    sw   $s4, -24($sp)
    move $fp,     $sp
    addi $sp,     $sp, -28

    move $s0,    $a0        # Fantasma
    lw   $s1,   ($a0)       # Dir. del fantasma
    lw   $s2, 12($a0)       # Dir. opuesta al movimiento del fantasma
    andi $s2,    $s2, 0xF
    li   $t0,    3
    sub  $s2,    $t0, $s2

    # Si es una interseccion, chequea caminos disponibles
    # No tomara en cuenta la direccion opuesta como disponible
    move $s3, $zero         # Contador de dir. disponibles
```

```
la $s4, ($sp)

# Arriba (salta la direccion si es la opuesta)
beq $s2, 1, Fantasma_cambiar_dir_chequear_derecha

# Verificamos si la dir. de arriba es camino
move $a0, $s1
jal  obtener_dir_arriba
move $a0, $v0
jal  Fantasma_es_camino

bnez $v0, Fantasma_cambiar_dir_chequear_derecha

# Empila las direcciones disponibles para moverse
li  $t0,  1
sw  $t0, ($sp)
add $s3,  $s3, 1
add $sp,  $sp, -4
Fantasma_cambiar_dir_chequear_derecha:
    beq $s2, 3, Fantasma_cambiar_dir_chequear_abajo

    # Verifica si la dir. derecha es camino
    move $a0, $s1
    jal  obtener_dir_derecha
    move $a0, $v0
    jal  Fantasma_es_camino

    bnez $v0, Fantasma_cambiar_dir_chequear_abajo

    li  $t0,  3
    sw  $t0, ($sp)
    add $s3,  $s3,  1
    add $sp,  $sp, -4
Fantasma_cambiar_dir_chequear_abajo:
    beq $s2, 2, Fantasma_cambiar_dir_chequear_izquierda

    # Verifica si la dir. abajo es camino
    move $a0, $s1
    jal  obtener_dir_abajo
    move $a0, $v0
    jal  Fantasma_es_camino

    bnez $v0, Fantasma_cambiar_dir_chequear_izquierda

    li  $t0,  2
    sw  $t0, ($sp)
    add $s3,  $s3,  1
    add $sp,  $sp, -4
Fantasma_cambiar_dir_chequear_izquierda:
    beqz $s2, Fantasma_cambiar_dir_verificar

    # Verifica si la dir. izquierda es camino
    move $a0, $s1
    jal  obtener_dir_izquierda
    move $a0, $v0
    jal  Fantasma_es_camino

    bnez $v0, Fantasma_cambiar_dir_verificar

    move $t0, $zero
    sw   $t0, ($sp)
    add  $s3,  $s3,  1
    add  $sp,  $sp, -4
Fantasma_cambiar_dir_verificar:
        bnez $s3, Fantasma_cambiar_dir_escoger_direccion

        # Se mueve en la direccions contraria si es la unica opcion
        sw  $s2, ($sp)
        add $s3,  $s3,  1
        add $sp,  $sp, -4
    Fantasma_cambiar_dir_escoger_direccion:
        move $a0, $s3
        add  $a1, $sp,  4

        jal escoger_aleatorio
        sb  $v0, 12($s0)

        # Desempila direcciones disponibles
        move $sp, $s4

Fantasma_cambiar_dir_fin:
    # Epilogo
    move $sp,    $fp
    lw   $fp,    ($sp)
    lw   $ra,  -4($sp)
    lw   $s0,  -8($sp)
    lw   $s1, -12($sp)
    lw   $s2, -16($sp)
    lw   $s3, -20($sp)
    lw   $s4, -24($sp)

    jr $ra


# Funcion: Verifica si el pixel en la direccion es un camino
#          válido para el fantasma.
```

```
# Entrada: $a0: Direccion del Bitmap Display del fantasma.
# Salida:  $v0: 0 si el pixel es un camino.
#               1 de otra forma.
# Planificacion de registros:
# $t0: Color de comida / fondo / portal / Pac-Man.
# $t1: Color del pixel en la direccion.
# $s0: Dir. del Bitmap Display del fantasma.
Fantasma_es_camino:
    # Prologo
    sw   $fp,   ($sp)
    sw   $ra, -4($sp)
    sw   $s0, -8($sp)
    move $fp,    $sp
    addi $sp,    $sp, -12

    # Color del pixel
    lw   $t1, ($a0)
    move $s0,  $a0

    # Si es un camino (comida, fondo, portal o Pac-Man)
    move $v0, $zero
    lw   $t0, colorComida
    beq  $t1, $t0, Fantasma_es_camino_fin

    lw  $t0, colorFondo
    beq $t1, $t0, Fantasma_es_camino_fin

    lw  $t0, colorPacman
    beq $t1, $t0, Fantasma_es_camino_fin

    lw  $t0, colorPortal
    beq $t1, $t0, Fantasma_es_camino_verif_portal
    li $v0, 1
    j  Fantasma_es_camino_fin
Fantasma_es_camino_verif_portal:
    # Portal 6 (0, 18)
    li  $a0, 0
    li  $a1, 18
    jal coord_a_dir_bitmap
    beq $v0, $s0, Fantasma_es_camino_verif_portal_der

    # Portal 6 (0, 17)
    add $t1, $v0, 128
    beq $v0, $s0, Fantasma_es_camino_verif_portal_der

    # De otra forma, se trata del portal 5 (31, 17) o (31, 18)
    # Verifica recursivamente si el otro lado del portal es camino
    add $a0, $s0, -120     # DIRSIGUIENTE = DIRACTUAL - 30*4
    jal Fantasma_es_camino
    j   Fantasma_es_camino_fin

    Fantasma_es_camino_verif_portal_der:
        # Análogamente, verifica si el otro lado del portal es camino
        add $a0, $s0, 120  # DIRSIGUIENTE = DIRACTUAL + 30*4
        jal Fantasma_es_camino

Fantasma_es_camino_fin:
    # Epilogo
    move $sp,    $fp
    lw   $fp,   ($sp)
    lw   $ra, -4($sp)
    lw   $s0, -8($sp)

    jr $ra
```