Diseño del juego:
Va a seguir el estilo apocalíptico del enunciado

Patrones usados:

- Factory para la instanciación de Entidades
- Strategy para el movimiento y para eliminar Entidades
- Type Object para la clasificación de Infectados
- Visitor para decidir qué sucede ante una colisión
- Bridge entre Entidad y EntidadGrafica

Pseudocodigo

Movimiento del jugador:

El jugador solo se puede mover horizontalmente, y tiene su posición vertical fijada.
Cada actualización se ejecuta el método update() en Jugador

```
metodo update():
    x ← e.getEntidadGrafica().getX()
    if (usuarioPresionoTecla(flecha_izquierda):
      velocidadX = -1
    else if (usuarioPresionoTecla(flecha_derecha):
      velocidadX = 1

    nuevaX = x + velocidadX

    // se compensa por el ancho de la imagen para
    // no dejar medio sprite fuera de la ventana
    if (nuevaX < 0):
      nuevaX = 0 + e.getEntidadGrafica().getAncho()

    // se compensa por el ancho de la imagen para
    // no dejar medio sprite fuera de la ventana
    else if (nuevaX > limite_horizontal):
       nuevaX = limite_horizontal - getEntidadGrafica().getAncho()

    e.getEntidadGrafica().setX(nuevaX)
fin update()
```

Movimiento de infectados:

La velocidad vertical [dy] de los infectados es la que cuenta (no se mueven horizontalmente)
Los infectados tienen una posición [x] e [y]

Cada actualización se ejecuta el método update() en cada entidad Infectado:

```
metodo update():
        y ← e.getEntidadGrafica().getPosY()
        nuevaY = y + velocidadY

        if (nuevaY < limite_inferior):
                nuevaY = 100 // algun valor por defecto, o su posicion inicial

        e.getEntidadGrafica().setPosY(nuevaY)
fin update()
```

Pseudocodigo de generación de mapa:

```
Nivel nivel
int nivelActual
listaEntidades l
InfectadosFactory infectadosF
ProyectilesFactory proyectilesF
PowerUpsFactory powerUpF
int infectadosVivos

void generarMapa()

    nivel <- new Nivel
    nivelActual <- 0
    infectadosVivos <- 0

    l <- new Lista

    //Crea las factories
    infectadosF <- new FactoryInfectados
    proyectilesF <- new FactoryProyectiles
    powerUpF <- new FactoryPowerUps


    // Crea al jugador en determinada posicion en el
    // medio de la parte inferior de la pantalla
    Jugador j <- crearJugador(posX,posY)
    listaEntidades.add(j)

    cargarNivel()

fin generarMapa()
```

```
//Carga el nivel, spawneando la primer oleada
void cargarNivel()

    nivel <- new Nivel(nivelActual +1)
    nivelActual++

    // Solo contempla la creacion de la primer oleada
    // En deleteInfectados cuando reduzca el contador
      // a 0 infectadosVivos cargue la segunda oleada o en su
defecto
    // cargue otro nivel

    infectadosVivos <- nivel.getCantInfectados()
    int mayorInf <- infectadosVivos * nivel.getPorcentajeTipos()

    int posX <- random(tamañoPantalla)
    int posY <- -10 // Alto de los infectados = 10

    for (i <- 0; i < infectadosVivos(); i++){

      if (posX > tamañoPantalla)
           posX <- posX - tamañoPantalla

      // Spawnea de manera aleatoria un TipoA o TipoB
      if (random(1) = 1 && mayorInf != 0)
           infectadosF.make("TipoA", posX, posY)
           mayorInf <- MayorInf - 1
      else infectadosF.make("TipoB", posX, posY)

      // Separo por filas las hordas
      if (random(5) > 3){
           posX <- random(tamaño pantalla)
           posY <- posY - 10
      }

      posX <- posX + 10

    }

    nivel.setCantInfectados(0) // Para no cargar devuelta la
primera tanda

fin cargarNivel()
```

```
void cargarSegundaOleada(){

    infectadosVivos <- nivel.getCantInfectados2()
    int mayorInf <- infectadosVivos * nivel.getPorcentajeTipos()

    int posX <- random(tamañoPantalla)
    int posY <- -10 // Alto de los infectados = 10

    for (i <- 0; i < infectadosVivos(); i++){

      if (posX > tamañoPantalla)
           posX <- posX - tamañoPantalla

      // Spawnea de manera aleatoria un TipoA o TipoB
      if (random(1) = 1 && mayorInf != 0)
           infectadosF.make("TipoB", posX, posY)
           mayorInf <- MayorInf - 1
      else infectadosF.make("TipoA", posX, posY)

      // Separo por filas las hordas
      if (random(5) > 3){
           posX <- random(tamaño pantalla)
           posY <- posY - 10
      }

      posX <- posX + 10

    }

    // Para no cargar devuelta la primera tanda
    nivel.setCantInfectados(0)


fin cargarSegundaOleada()
```