**Inteligencia Artificial para Videojuegos**

***Práctica 5: aliados y enemigos de combate***



*Grupo 07*

*Alejandro Ortega Álvarez*

*Borja Cano Álvarez*

Problema a resolver

* La propuesta es rellenar en Unity a partir de un prototipo dado de un shooter en tercera persona ciertas características para darle más jugabilidad al simulador.



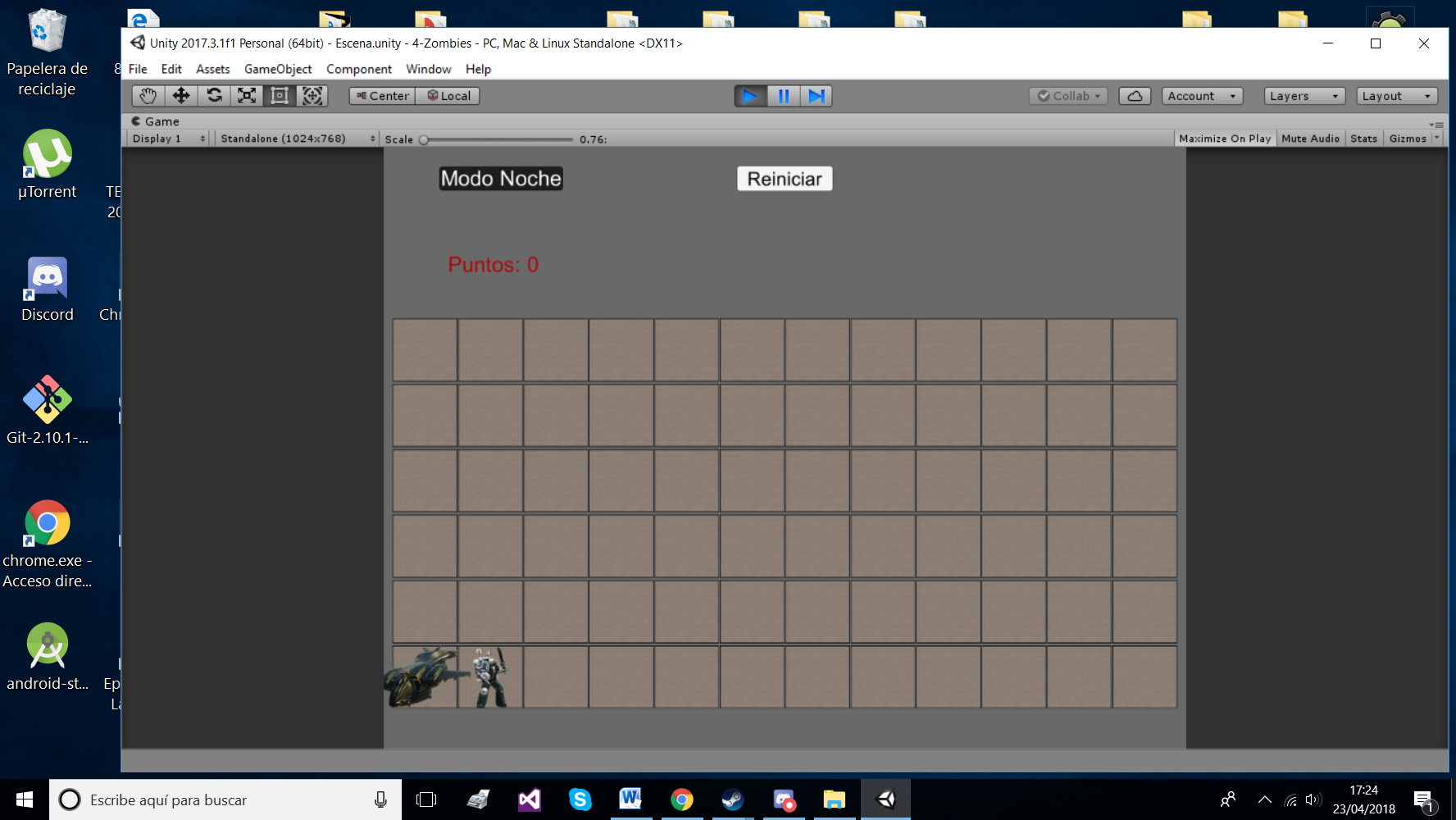
* Para ello se piden los siguientes requisitos:

1. Observar y comprender los árboles de comportamiento de los soldados
2. Crear varias entidades nuevas de fantasmas, entidades que vuelan por el escenario con un comportamiento de patrullas aleatorias por el mapa, sin perseguir al jugador.
3. Los fantasmas no mueren, pero se teletransportan a una posición aleatoria del mapa cuando se les dispara.
4. Los fantasmas hacen daño cuando atraviesan a los soldados.
5. Los soldados enemigos tienen como prioridad disparar a los fantasmas antes que disparar al jugador.

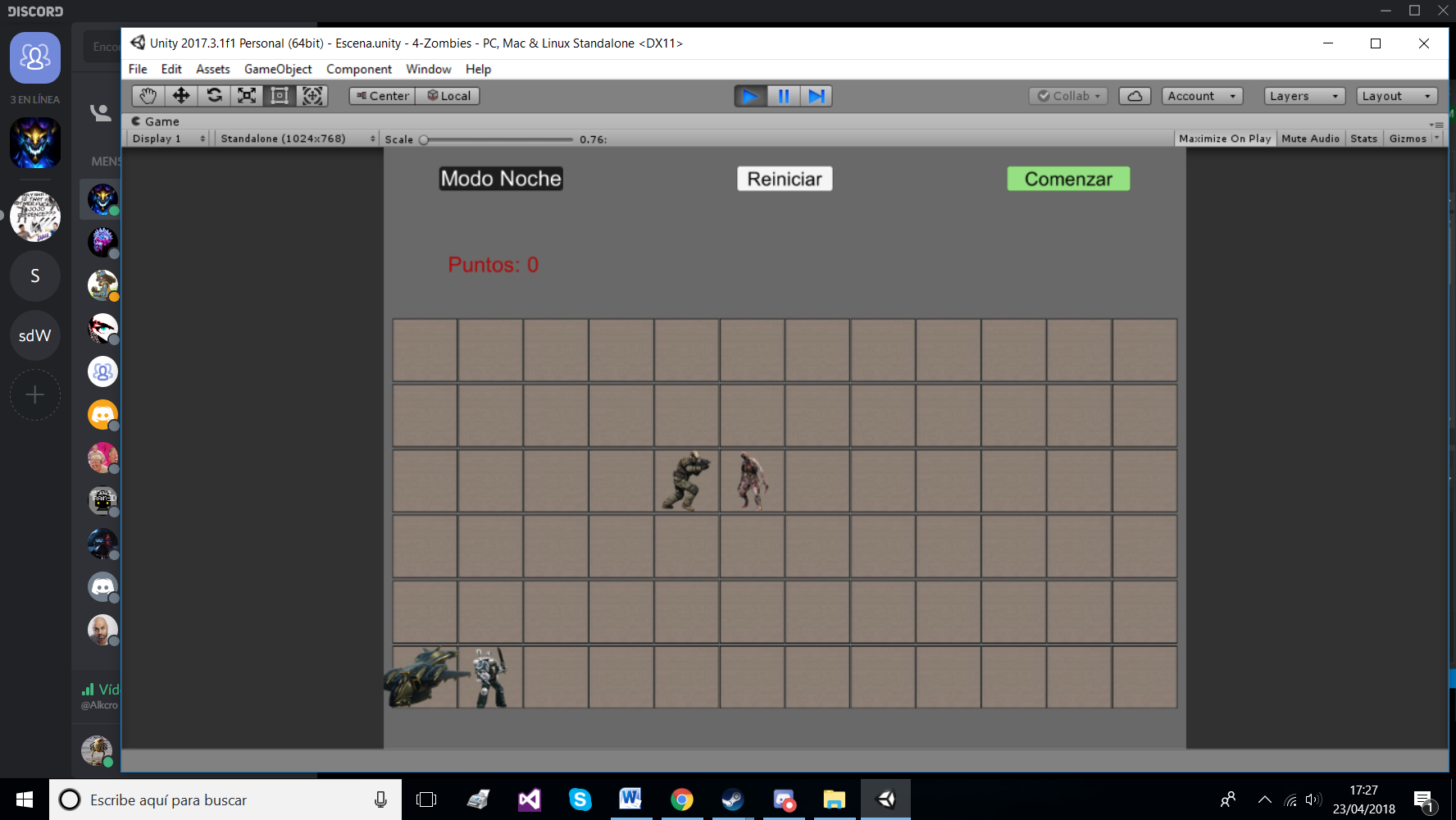
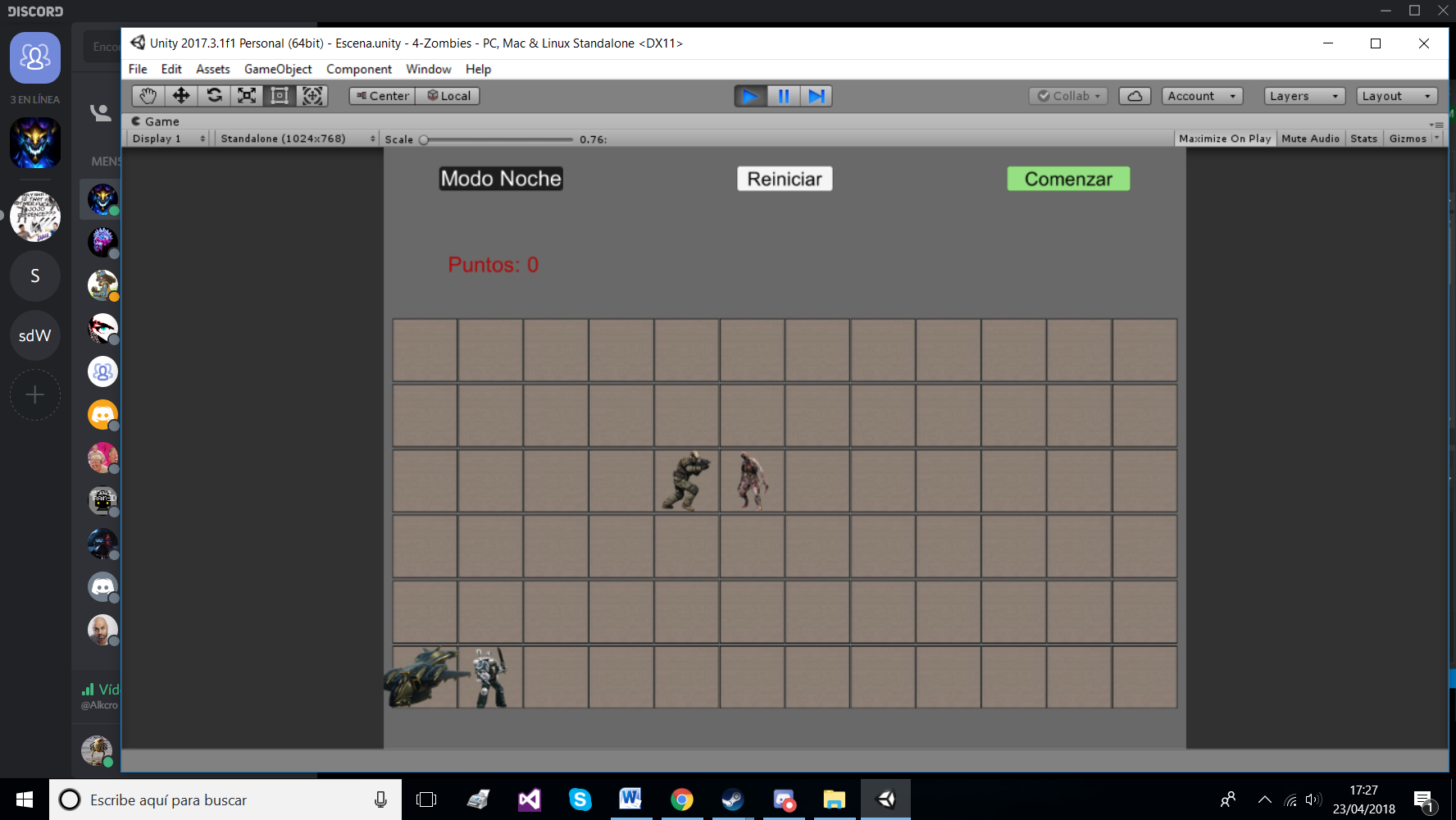
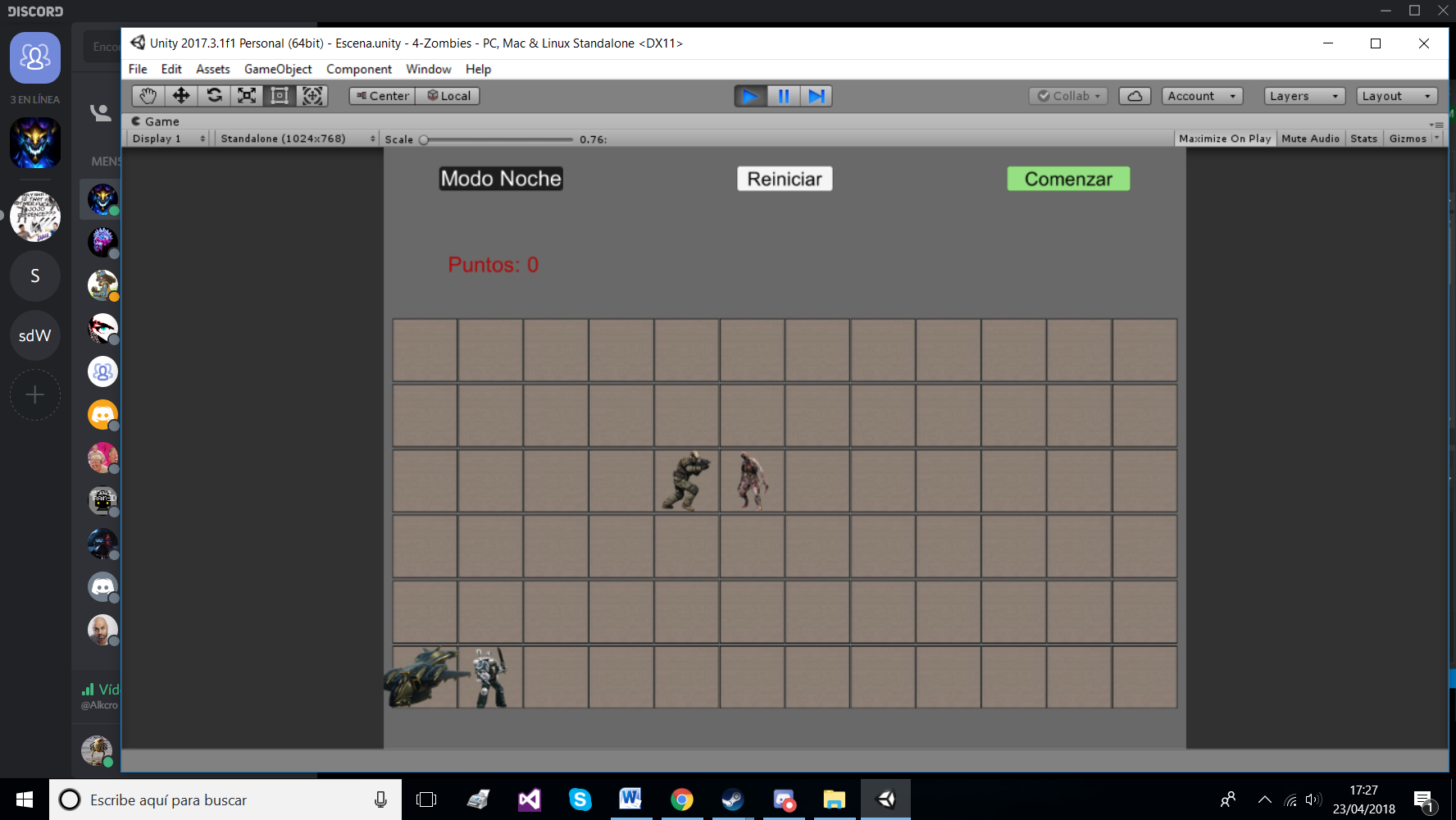
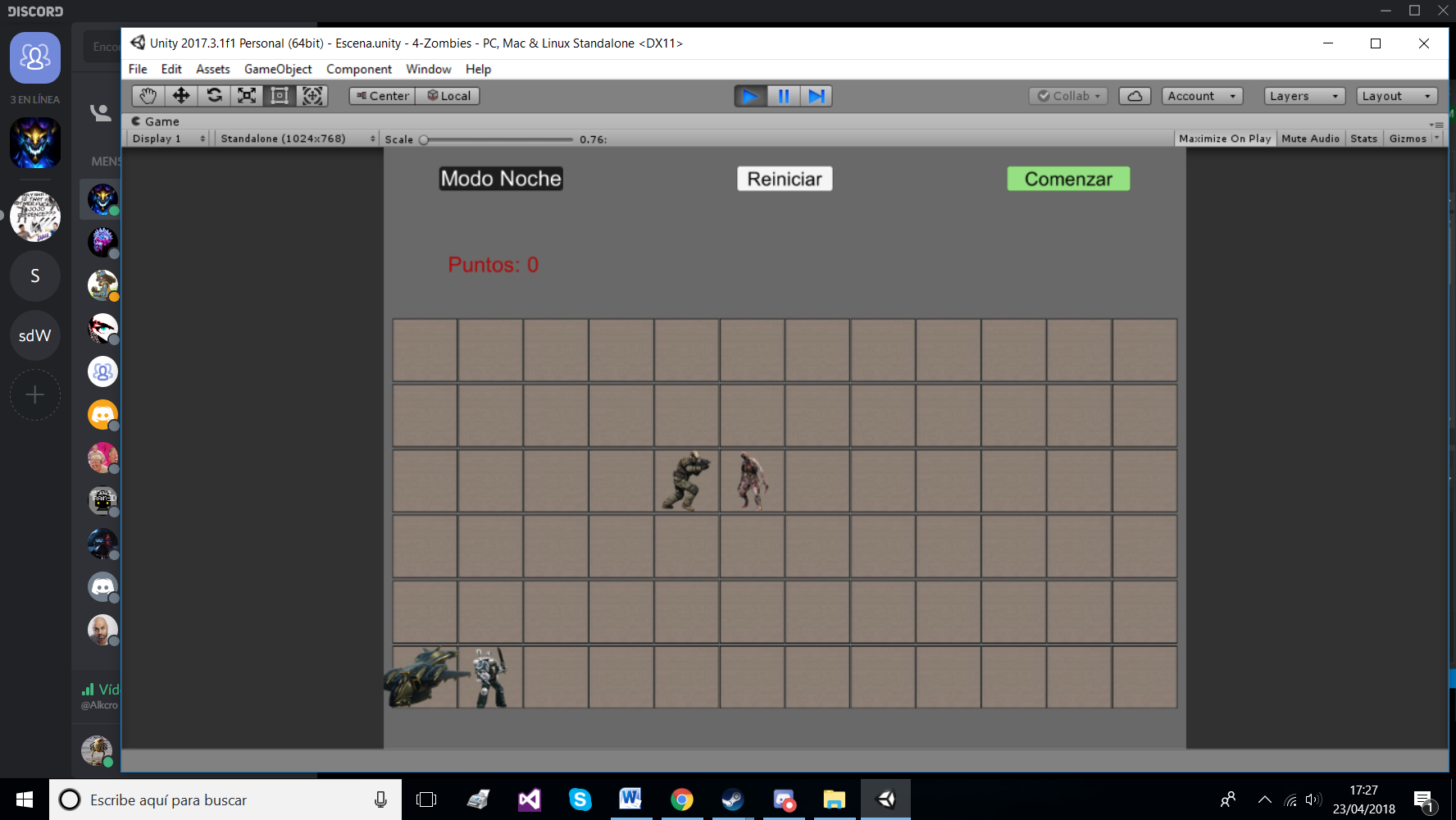
El juego

Repasamos el funcionamiento final del simulador.

Ésta es la vista nada más iniciar el juego:



Al presionar cualquier casilla (excepto las que están ocupadas por la nave y el agente) aparece un aliado y si se vuelve a hacer click en su lugar aparece un zombie.

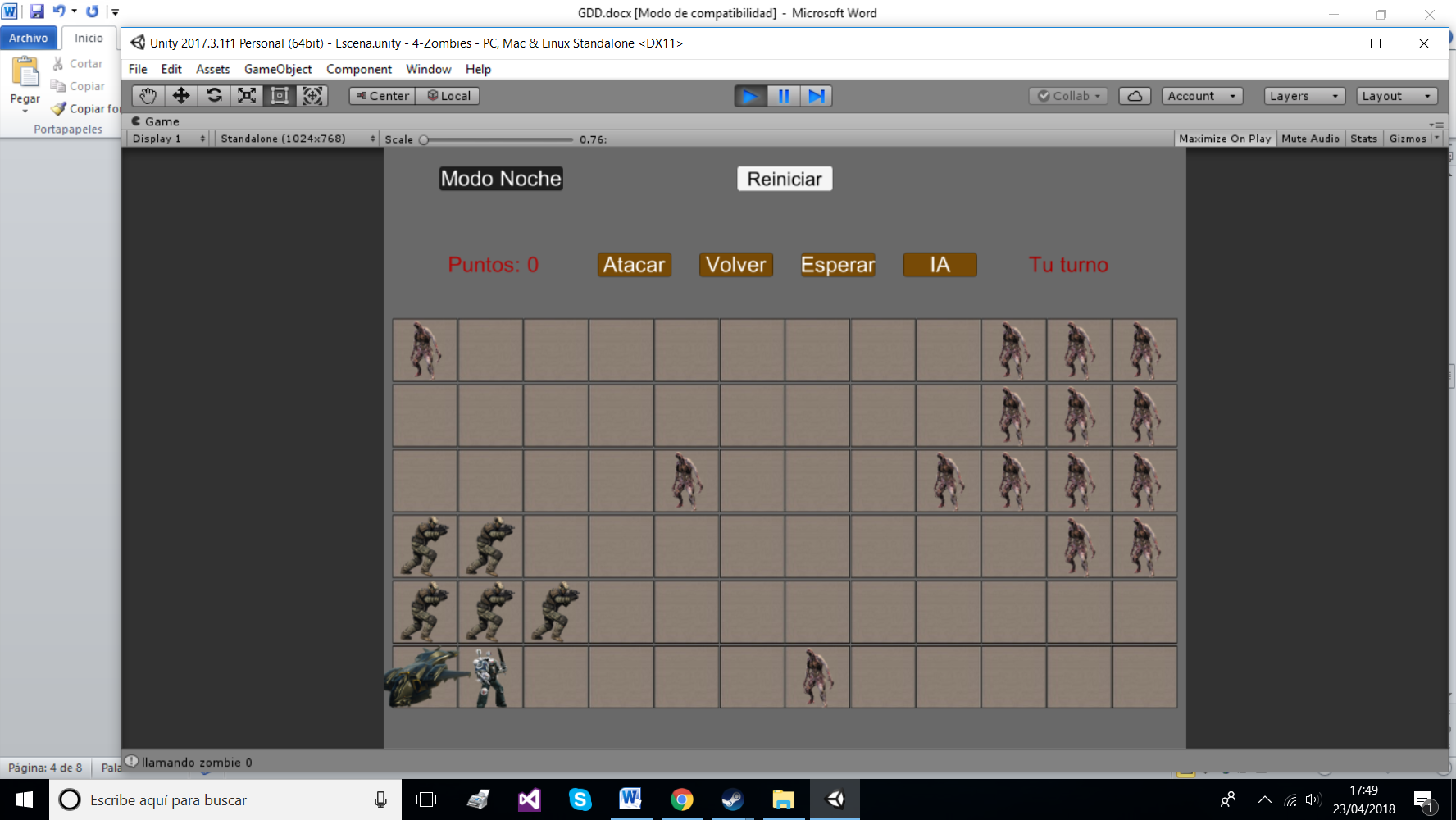


Una vez haya al menos un zombie en el mapa, se puede comenzar la batalla. La batalla se desarrolla por turnos, el turno de los zombies y el turno del agente. El primer turno se decide de forma aleatoria.

En cualquier momento de la partida se puede reiniciar o cambiar entre modo noche/día, pero no se puede clickear una casilla para añadir entidades si se ha comenzado ya la batalla.

**El turno del agente**

En el turno del agente se muestra el siguiente menú de acciones:



El usuario decide qué acción hace el agente o si lo decide la IA por él.

Tal como en el enunciado de la práctica, las acciones para cada turno son:

1. **Atacar** al zombie más cercano (avanza una casilla hacia él y hay batalla si entra en la misma casilla).
2. **Retroceder** una casilla hacia la nave.
3. **Esperar** y permanecer en el sitio.

**El turno de los enemigos**

Los zombies sólo buscan atrapar a los agentes y acabar con ellos. La única acción que realizan por lo tanto es avanzar una casilla hacia el agente más cercano o aliado, y atacar si entra en la misma casilla. Cada zombie tiene su turno independiente y espera a que haya terminado el turno del zombie anterior.

Los aliados **no** entran en los turnos, se quedan en la posición que estén para el resto de la partida.

**Los combates**

Los combates sólo tienen lugar cuando un zombie entra en la casilla de algún soldado o el héroe en la casilla de uno (o varios) zombies. En los combates se decide, mediante probabilidades, quién vive y quién muere en el conflicto.

Las probabilidades dependen de dos factores:

* La **situación**: cantidad de enemigos vs. cantidad de aliados. Es más probable que gane el héroe si dispone de más apoyo por parte de los aliados.
* La **destreza**: cantidad de aliados vs. visibilidad (si es o no modo noche).

Dependiendo de estos factores, puede suceder que muera el atacante o el atacado. Si hay varios atacados en una misma casilla se hacen los combates por separado.

**Puntuación**

Cada resultado de un combate se ve reflejado en la puntuación de la siguiente forma:

|  |  |
| --- | --- |
| **Si un aliado mata un zombie** | 1 punto |
| **Si el agente mata un zombie** | 5 puntos |
| **Si muere un aliado** | -10 puntos |
| **Si muere el héroe** | -50 puntos |

La IA del agente debe decidir la acción dependiendo tanto de los puntos como de la situación.

\*La puntuación máxima que se puede conseguir con 20 zombies es de 100 puntos. (Victoria arrasante)

\*La puntuación mínima con 5 aliados es de -100 puntos. (Fracaso total)

**Cuándo acaba la partida**

La partida acaba si muere el agente, mueren todos los zombies o se retira el agente (consigue llegar a la nave antes de que le acorralen los zombies).

Implementación

* El Script de GameManager se encarga de gestionar todo el juego antes de comenzar la partida en sí. Crea el tablero, todas las entidades (tanto las que ya están como las que se van colocando), controla el modo noche y el reinicio.
* El Script de GUI se encarga de la gestión de turnos, puntuación, eliminación de entidades por combate y cálculo de la destreza y la situación. Es decir, controla todo el GamePlay en cuanto se da a comenzar.
* Los Scripts de OnClick, como su nombre indica, realizan determinada respuesta dependiendo de qué entidad se haya clicado como se indicó más atrás (clickeado de casilla-aliado-enemigo).
* TurnoPlayer decide qué acción realizará el jugador, ya sea por IA o por usuario, y la lleva a cabo.
* TurnoEnemy controla el comportamiento de cada enemigo en el mapa, busca el objetivo más cercano y mueve una casilla hacia él cuando es su turno.

Algoritmo

Tanto la toma de decisiones de la IA del jugador como el desarrollo de los combates dependen de un algoritmo basado en probabilidades.

Tal como se ha indicado, existen dos variables globales a lo largo de la batalla que son la Situación y la Destreza, que condicionan el cálculo de las probabilidades.

La **situación** puede ser: mucho enemigo, mucho aliado o una situación neutral.

La **destreza** la condiciona tanto la visibilidad (modo noche) como el número de aliados. Puede ser buena, mala o regular.

Tanto destreza como situación son variables también calculadas con probabilidades.

**La IA del agente**

Lo primero que hace el héroe es calcular la destreza y la situación, y a partir de ello calcula la probabilidad que tiene de realizar cada acción.

El cálculo de probabilidades de realizar las acciones es la siguiente (cada vez que hay una toma de decisión de la IA del agente salen por Debug los porcentajes):

|  |  |  |  |
| --- | --- | --- | --- |
| Situación  Destreza | Mucho Zombie | Neutral | Mucho Aliado |
| Mala | 10-30-60% | 30-40-30% | 70-10-20% |
| Regular | 20-30-50% | 50-30-20% | 80-10-10% |
| Buena | 30-30-40% | 70-20-10% | 90-10-0% |

**Atacar-Retroceder-Esperar**

\*Extra: la toma de decisiones hemos decidido que también dependa de los puntos. El héroe se llena de valor si ya ha conseguido matar a varios enemigos, por lo que si los puntos de la partida son positivos, tendrá un 10% más de posibilidad de atacar.

**El combate**

De forma semejante, se calcula la probabilidad de que ganen los soldados o los enemigos cuando surge el conflicto. Las probabilidades en este caso dependen exclusivamente del número de apoyo de aliados y del número de enemigos, y una pequeña bonificación para los enemigos si hay mala visibilidad.

Explicación más detallada abajo en el apartado de **Resultados**.

**Problemas**

-Principal problema el no saber dónde estaban las cosas. Los elementos del prototipo que nos dabas estaban almacenados en demasiadas carpetas (algunas con nombres muy parecidos o idénticos) y era complicado buscar la ruta si no era fuera de Unity. Aun habiendo encontrado el archivo que buscábamos, era muy difícil detectar el elemento que queríamos tocar, por lo que ha habido apartados que directamente no hemos conseguido lograr por no saber dónde se modificaba.

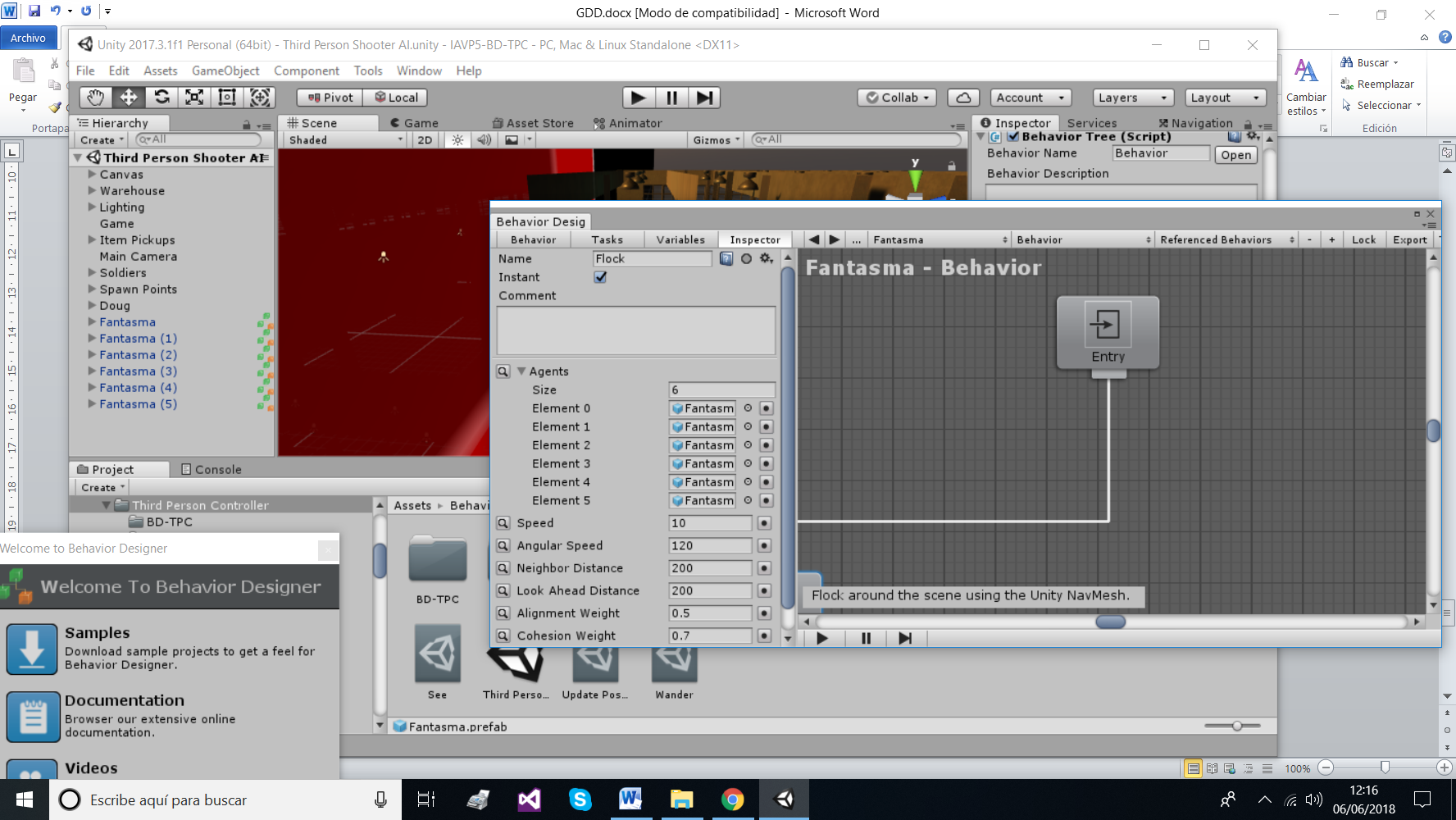
-Demasiado tiempo empleado para 4 modificaciones que tiene la práctica, casi todo el tiempo empleado en tratar de comprender cómo iba y buscar los elementos que necesitáramos. Y sin el tutorial que nos mandaste hubiéramos tenido que gastar casi el doble de tiempo más.

-En el caso de los fantasmas, se podía pasar el prefab de un proyecto a otro pero no con las partes necesarias (sólo me aparecía el cuerpo y tuve que meter a mano los brazos, la cabeza y las texturas).

-Hicimos perfectamente el pequeño detalle de que “flotaran” los fantasmas, con el seno en el script Flock para mover la variable que posicionaba la distancia con el suelo, pero al pasarlo al otro proyecto se perdió y no conseguimos volver a encontrar la variable.

-Dificultad a la hora de tocar los Scripts que ya venían, por ser demasiado extensos y no conseguir encontrar nada, al final la solución para las implementaciones era crear otro script auxiliar que accediera al que correspondiera.

-Nos pasaban cosas extrañas con el comportamiento de los fantasmas (completamente el mismo que en el otro lado); cuando conseguíamos que funcionara correctamente, nos poníamos a tocar otra cosa y, sin haber tocado el comportamiento, dejaban de funcionar, y hacía cosas raras como que los fantasmas se activasen al pasar un soldado enemigo cerca. Además, en la ventana del comportamiento, cada vez que reiniciamos Unity, se pierden las referencias a los fantasmas y hay que volverlas a poner cada vez que arrancamos la práctica, es un poco rollo.



En el resultado final, habiendo tocado las distancias del objetivo, con los vecinos y demás hemos conseguido controlar el comportamiento salvo porque a veces se quedan parados unos breves segundos, no nos gusta pero creemos que es porque trata de buscar una ruta imposible y hasta que no le llegue otra no continúa, y eso no sabemos cómo arreglarlo ya que es de la propia implementación del algoritmo.

-El único apartado que nos quedó por hacer fue el de que los soldados disparasen a los fantasmas. Le proporcionamos un nuevo comportamiento a su árbol, más prioritario que el de disparar al jugador que fuera exactamente igual pero disparando a los fantasmas y sin tener que ser dañado para activarse. Pero nos daba referencia nula al objeto pese a haberlo cambiado y tuvimos que quitarlo para que no diera errores. Además, fue un lío ya que tratamos de crearlo a partir de una copiar del comportamiento que ya tenía de disparar al soldado, pero al modificarlo se modificaban los 2 comportamientos y no había forma de crearlo desde 0 ya que había que añadir muchas cosas, al final después de unas cuantas pruebas tuvimos que dejarlo como estaba.

-También problemas con la carga de Unity. Al principio no tanto, pero después de varios cambios al darle al play a la simulación se quedaba pillado un rato bastante largo debido al peso del proyecto, en una ocasión tuvimos que forzar el cerrado de la aplicación y es bastante fastidio si no nos damos cuenta de guardar los cambios.

**Resultados**

-En el resultado final tenemos el juego base con el añadido de los fantasmas casi a la perfección. Se mueven con el comportamiento indicado, si atraviesan a los soldados les causa daño (a los soldados enemigos también, les hemos añadido el mismo componente) y si se les dispara a los fantasmas se teletransportan a un punto aleatorio del mapa.

Referencias y apoyos del desarrollo de la práctica:

* *Apuntes del Campus Virtual y enlaces de los prototipos*

*Repositorio de Github:* https://github.com/alex97ortega/IA.git