**Introducción**

NWO *(New World Order)* es un juego desarrollado para Amstrad CPC 464 con la librería CPCtelera, en el se da lugar la lucha del bien contra el mal para salvar la humanidad de una invasión reptiliana. El personaje principal, Foyone, deberá intentar rescatar la máxima cantidad de aliados humanos de las manos de los enemigos reptilianos, los cuales pretenden hacer de la humanidad esclavos.

Foyone dispone de una afilada y potente espada con la cuál puede acabar con todos esos asquerosos reptiles, pero tendrá que hacerlo desde un punto donde los enemigos no estén mirando, de lo contrario este morirá, debido a que los reptilianos son bastante veloces.

Cada vez que Foyone consiga acabar con un enemigo este obtendrá 200 puntos para su *score*, cuando consiga rescatar a un aliado obtendrá 100 puntos. De lo contrario cuando un enemigo consiga capturar y comerse un aliado te restará 20 puntos a tu marcador, parece poco pero estos reptilianos son demasiado listos. Si matas por error a un aliado el score se reseteara. El score es el objetivo secundario del juego, siendo el primero la diversión.

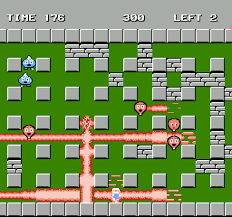
**Ficha técnica**

* Lenguaje utilizado: C con la librería CPCtelera y sus herramientas.
* Modo gráfico utilizado: Modo 0 – 160x200 en 16 colores
* Tamaño del ejecutable .CDT: 29,3 KB
* Software para el dibujado de sprites: GIMP
* Paleta utilizada de 16 colores (en orden):

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ***Firmware Number*** | ***Hardware Number*** | ***Colour Name*** | ***R%*** | ***G%*** | ***B%*** | **Hexadecimal** | **RGB values** |
| 1 | 44h (or 50h) | Blue | 0 | 0 | 50 | #000080 | 0/0/128 |
| 0 | 54h | Black | 0 | 0 | 0 | #000000 | 0/0/0 |
| 3 | 5Ch | Red | 50 | 0 | 0 | #800000 | 128/0/0 |
| 12 | 5Eh | Yellow | 50 | 50 | 0 | #808000 | 128/128/0 |
| 13 | 40h (or 41h) | White | 50 | 50 | 50 | #808080 | 128/128/128 |
| 15 | 4Eh | Orange | 100 | 50 | 0 | #FF8000 | 255/128/0 |
| 26 | 4Bh | Bright White | 100 | 100 | 100 | #FFFFFF | 255/255/255 |
| 7 | 45h (or 48h) | Purple | 100 | 0 | 50 | #FF0080 | 255/0/128 |
| 24 | 4Ah | Bright Yellow | 100 | 100 | 0 | #FFFF00 | 255/255/0 |
| 6 | 4Ch | Bright Red | 100 | 0 | 0 | #FF0000 | 255/0/0 |
| 16 | 47h | Pink | 100 | 50 | 50 | #FF8080 | 255/128/128 |
| 20 | 53h | Bright Cyan | 0 | 100 | 100 | #00FFFF | 0/255/255 |
| 9 | 56h | Green | 0 | 50 | 0 | #008000 | 0/128/0 |
| 10 | 46h | Cyan | 0 | 50 | 50 | #008080 | 0/128/128 |
| 21 | 5Ah | Lime | 50 | 100 | 0 | #80FF00 | 128/255/0 |
| 22 | 59h | Pastel Green | 50 | 100 | 50 | #80FF80 | 128/255/128 |

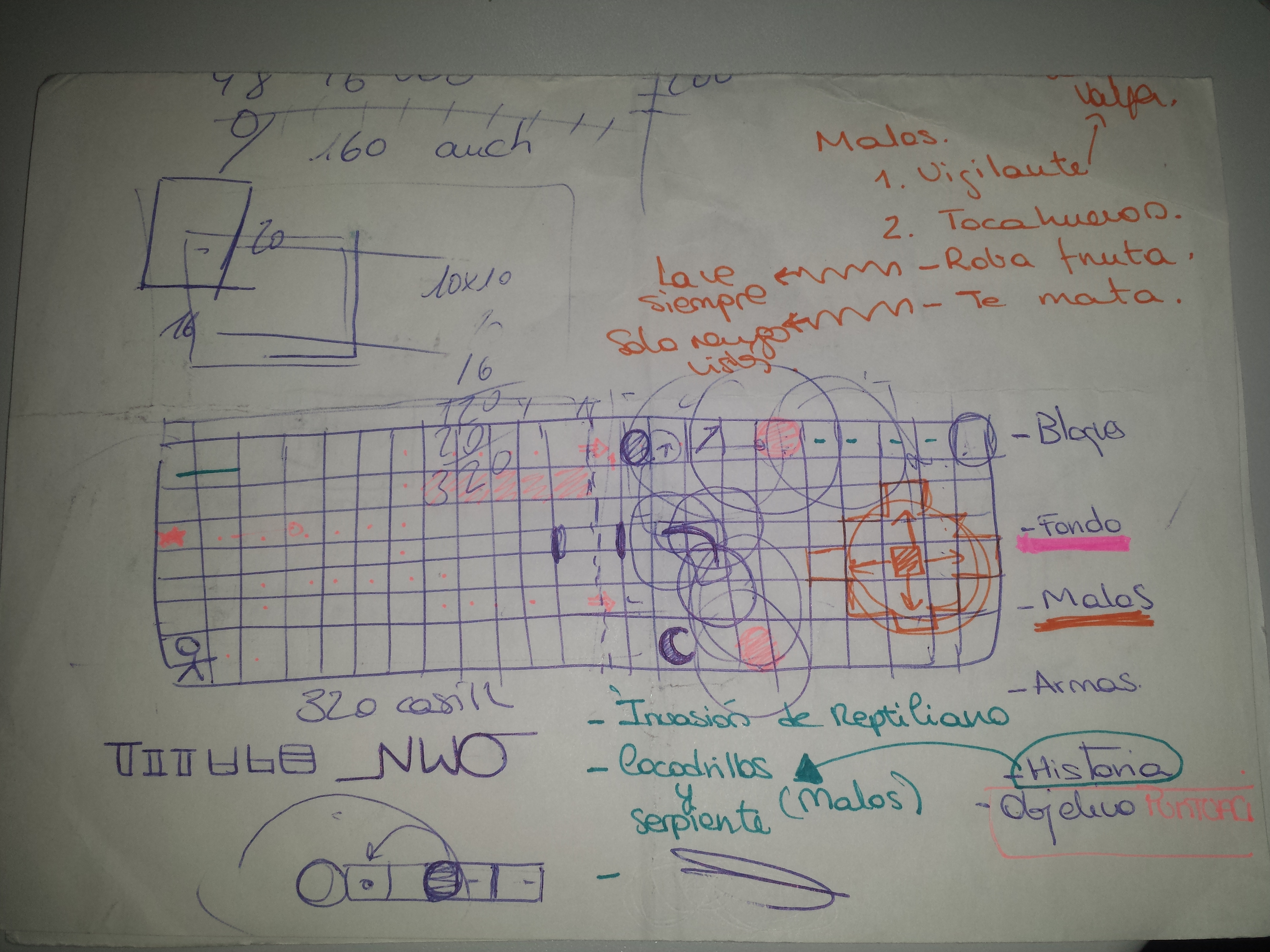
**¿Cómo se hizo?**

Después de una larga reunión de equipo con una gran tormenta de ideas sobre que hacer y no sacar nada en claro decidimos observar los juegos existentes para Amstrad CPC 464. Debido a las limitaciones de la máquina y nuestro escaso conocimiento en desarrollo de videojuegos decidimos empezar por una idea sencilla, algo similar el mítico Bomberman.

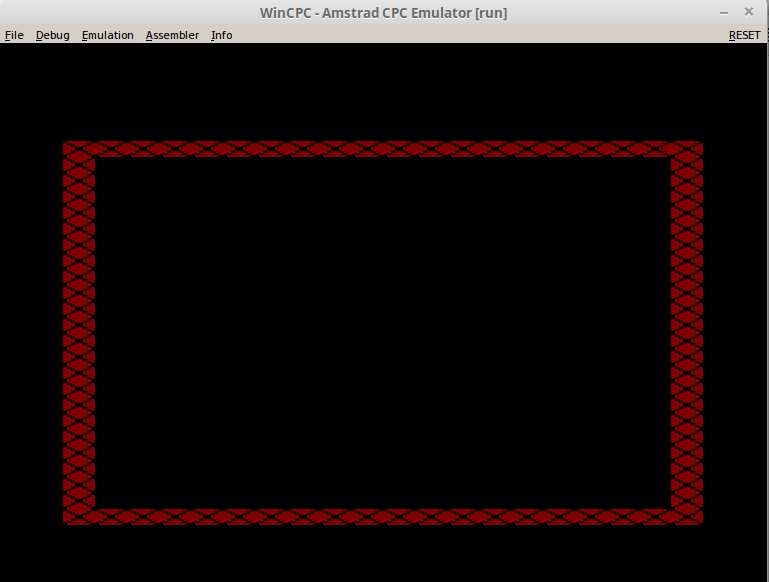


Aunque comenzamos con esta idea en mente, nuestro producto final no se parecería en nada a este más que en el tipo de mapa. Tras las primeras pruebas con CPCtelera de manera individual, comenzamos a desarrollar la idea en más profundidad teniendo en cuenta nuestros conocimientos y las limitaciones de la máquina, con un único objetivo llegar a tiempo para la #CPCRetroDev 2015.

La primera vez que nos pusimos a trabajar sobre la idea, cogimos un papel y empezamos a desarrollar diferentes conceptos, como el mapa, nombre, historia, enemigos, objetivos.. etc este documento se puede observar a continuación.



Decidimos que comenzaríamos el desarrollo con un mapa dividido en casillas de 8x8 pixeles (4x8 bytes), también empezamos a pensar que necesitaríamos parte de pantalla para mostrar puntuación y/o vidas, por lo que empezamos a hacer cálculos contando con que dividiendo toda la pantalla del CPC 464 en 8x8 píxeles obteníamos 25 filas y 20 columnas. Empezamos desechando la primera fila para obtener espacio para score y/o puntuación y también decidimos desechar dos filas y dos columnas más para mostrar un borde alrededor de nuestro mapa.



Teniendo la idea clara de que íbamos a disponer de 22 filas y 18 columnas hábiles reales comenzamos a desarrollar un mapa de prueba sobre el que trabajar con unos personajes temporales. También diseñamos un sprite 8x8 que representaría nuestros bloques sólidos.







Teniendo el mapa y un personaje principal empezamos a desarrollar las colisiones, el movimiento básico del personaje, y ajustando cosas referentes a la programación, como definición de entidades. Tras esto, creamos una base para los enemigos y sus sensores los cuáles serán utilizados para recabar información del mapa, les proporcionara una visión frontal. Además fuimos diseñando los sprites finales con GIMP.



Después de organizar mucho código y limpiarlo, proseguimos con el desarrollo del movimiento de los enemigos, una de las tareas más complicadas y necesarias. Tras intentar, sin éxito, diseñar un movimiento mínimamente lógico, decidimos pasar al pathfinding para poder apoyarnos en este para realizar nuestro movimiento “básico”. Por movimiento “ilógico” es que cuando los enemigos no observan nada en su sensor, los enemigos se movían en bucle (hacia un lado, hacia el otro y vuelta a empezar), movimiento el cuál puede ser lógico para un enemigo tipo vigilante pero no para lo que necesitábamos en un primer momento, un enemigo que fuera capaz de explorar el mapa sin pasar por el mismo sitio continuamente.

Desarrollamos un pathfinding especial basándonos en A\*, máquina de estados para los enemigos y unos razonamientos basados en objetivos. Al mismo tiempo, desarrollábamos más sprites para las animaciones y definir un ataque básico para el enemigo.



Con todo esto desarrollado, y con el tiempo apretándonos empezamos a meter los aliados a nuestro juego, la capacidad del personaje principal de matar a los enemigos, la muerte de nuestro personaje principal el funcionamiento del marcador y añadimos un segundo tipo de enemigo el cual tendría un comportamiento diferente en la toma de decisiones.

*ENEMIGO 1 ENEMIGO 2*

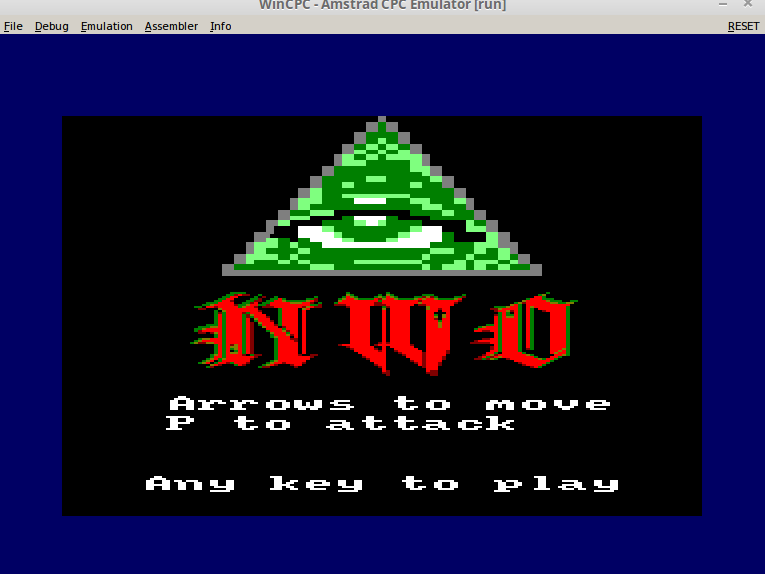




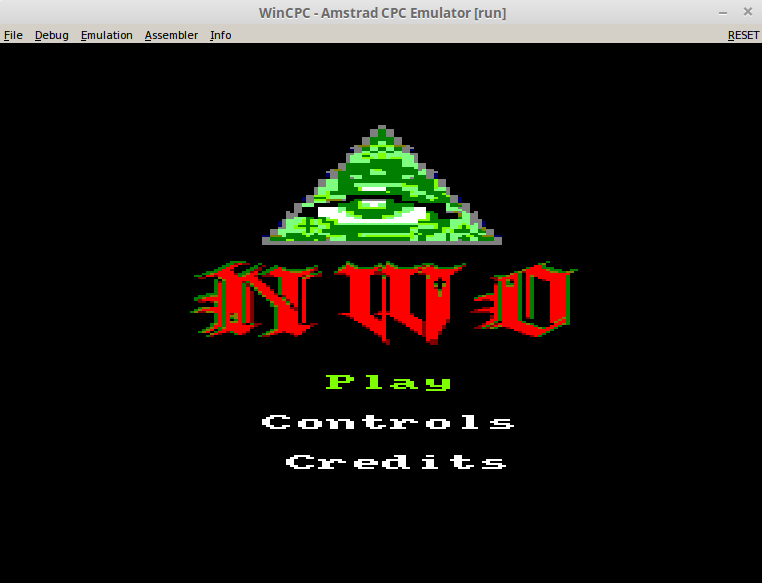
Una vez realizado el pathfinding nos apoyamos en este para realizar el movimiento básico de nuestros enemigos, dividimos el mapa en 24 zonas imaginarias sobre las cuáles se iteraria. También añadimos una pantalla principal provisional para representar el gameover y mostrar los controles.

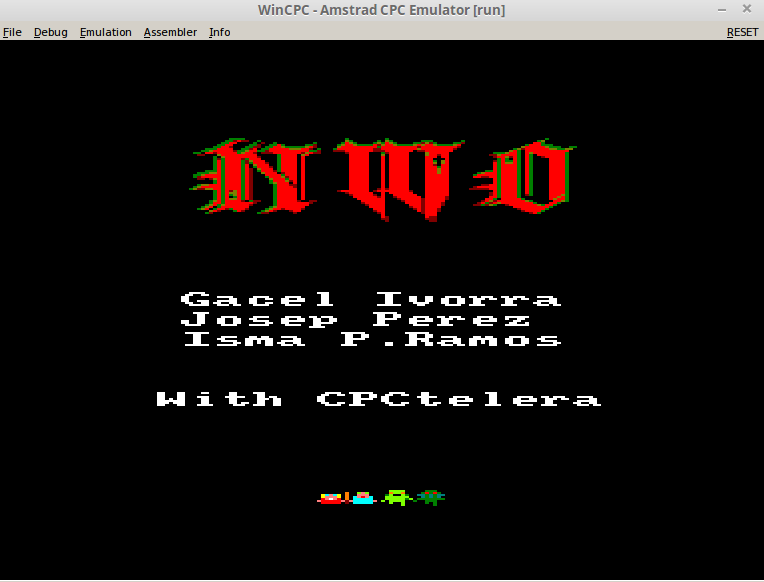


Con todo esto realizado y muchos bugs que resolver nos pusimos a solucionar diferentes bugs y realizando la generación de las posiciones donde se situarán los aliados en un primer instante de manera aleatoria y probamos con diferentes cantidades de enemigos la jugabilidad y estabilidad. Volvimos a realizar una nueva pantalla principal.



Tras un gran número de pruebas, ajustamos un número razonable de enemigos que se generarían en un primer instante, además de una escala de como iría aumentando este número según estos fueran asesinados. Continuamos depurando diferentes bugs, calibrando el bucle principal del juego y desarrollando un menú con su menú para observar los controles y los créditos.





Después de esto nos empezamos a encontrar con problemas de memoria, por lo que tomamos la decisión de empezar por hacer el logo de menor tamaño y optimizar el código para ahorrarnos algo de memoria consiguiendo aproximadamente 2Kbytes de memoria, ajustamos el número de aliados, su frecuencia de aparición, las decisiones de los enemigos que priorizarían a estos y ajustar el bucle principal del juego. Para terminar, arreglamos los últimos bugs que encontramos, y con muchas ideas en la cabeza, que por falta de tiempo y de experiencia no pudimos implementar, nos encontramos con la primera versión final de NWO.



