1. Architecture (programming language, version of CPCtelera ...)

The programming language that we have used for the development of the videogame is assembler z80.
The version of CPCtelera that we have used has been that of the development branch with the last commit: e2c6822

You can obtain this tool in the following link:

- https://github.com/lronaldo/cpctelera/tree/development

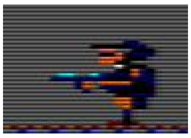
2. Problems encountered (masks, double buffer, lack of memory ...)

- Incorrect mask that prevented generating the .cdt file
- Error when implementing the double buffer and drawing the obstacles each time the protagonist was updated
- Memory problems, which forced us to reject certain changes and erase levels
- The images of the menu and manual along with the tilemap were too compressed, so we had to dispense with some screens and the tilemap replaced it with a gray background color.

3. Reference at the game Chicago's 30

We use two sprites of the main characters of the game Chicago's 30, one is the enemy and the other the protagonist:

- **Enemy**: The enemy of Chicago's 30 appears as a special enemy starting at level 9.



- **Protagonist**: The protagonist appears as final boss in the last level, that is, in level 20.
  [INSERTAR CAPTURA AQUÍ TAMBIÉN]

4. Dedicated work hours

At the beginning, we dedicated 2 weeks to learn the basics of the language and the work environment, this took us 8 hours per week for each team member. After those two weeks, we had another 5 weeks in which we followed two parallel processes: architecture learning and videogame development.

These are the tasks used in the development of the game that are described in more detail:

1. In the first week we decided the mechanics of the main character. Also, we implemented the movement the teleportation, the creation of the obstacles of the maps and the collisions with them.

2. In the second and third week we implemented the mechanics of the characters, the shooting, crouching and running. Also the enemies, objects, the change of view of a same level and the change to the next level.

3. The last two were dedicated to the correction of errors, such as the erasure of the bullet when colliding with an enemy or double buffer, and the aesthetics of the game, which initially used tiles, but since we lacked memory we had to reject the idea and Use small sprites that were repeated over obstacles.