Alicia Custodia García Yuen

marie Estelle Melaine pamen

RODRIGO SÁEZ ESCOBAR

Master Mind user´s manual

Content

[1) Installation 2](#_Toc137320758)

[1.1 VSCode and Project 2](#_Toc137320759)

[1.2 Extensions 2](#_Toc137320760)

[1.3 Build 2](#_Toc137320761)

[1.4 Run 3](#_Toc137320762)

[2) Play 3](#_Toc137320763)

[2.1 Master Mind basics 3](#_Toc137320765)

[2.2 Game Controls 4](#_Toc137320766)

[3) Code 5](#_Toc137320767)

[3.1 Code diagram 5](#_Toc137320769)

[3.2 Code function explanation 6](#_Toc137320770)

[4) Participation 11](#_Toc137320774)

[5) Bibliography 11](#_Toc137320775)

# Installation

The first step to play this Mastermind game is to have a Z80 compiler.

## VSCode and Project

First, we need VSCODE to run the code, where you can check the last version here <https://code.visualstudio.com/download>

Obviously, the project itself which is public in GitHub here <https://github.com/saezro/Mastermind-Z80>

## Extensions

To be able to compile and run the code is necessary to have these two extensions.

Interfaz de usuario gráfica, Texto, Sitio web

Descripción generada automáticamente

## Build

The folder of the project has already been built but to save any change follow these steps.

To build the project first we have to the add the folder to VSCode like this:

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Then select the project folder called “Mastermind” that you downloaded and that will open the project and should look like this:

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

To build it we must open the archive “.asm” and select “Terminal => Run Build Task”

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

## Run

To run the game after building it select “Run => Start Debugging”

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

# Play



## Master Mind basics

The game Mastermind consists in someone creating a code of 4 colors and other one trying to solve it.

The solver tries a combination of colors and the coder gives hints about it, red(if a color is in the right place), white(if a color is right but not in his place) and nothing(if nothing is right.)

For example the code is (**RED RED BLUE YELLOW**) and the solver says (**BLUE BLUE BLACK WHITE**) so the hits are (**WHITE WHITE**) because only the blue is right but not in the right place and theres 2 of them.

Interfaz de usuario gráfica

Descripción generada automáticamente con confianza bajaThis is and example of the game we have done.

In the first column the only color that is right is **BLACK** so the hint is just **WHITE**.

In the second column **BLACK** is in the right place and **RED** is not so the hint is **RED** **WHITE**.

In the last column all the colors are right in the place they should so the hints are **RED RED RED RED** that means that the game ends whith a win.

## Game Controls

Interfaz de usuario gráfica

Descripción generada automáticamente con confianza baja

(Example of a win)

To play this game there’s only 3 main keys to control it “**Z**” to change to the next color, “**X**” to change to the color before and “**C**” to accept the color.

The list of colors is (**BLACK RED PINK GREEN BLUE YELLOW WHITE**) starting in black.

When you finish the game winning or loosing you can press “**V**” to restart the game.

Interfaz de usuario gráfica

Descripción generada automáticamente con confianza media

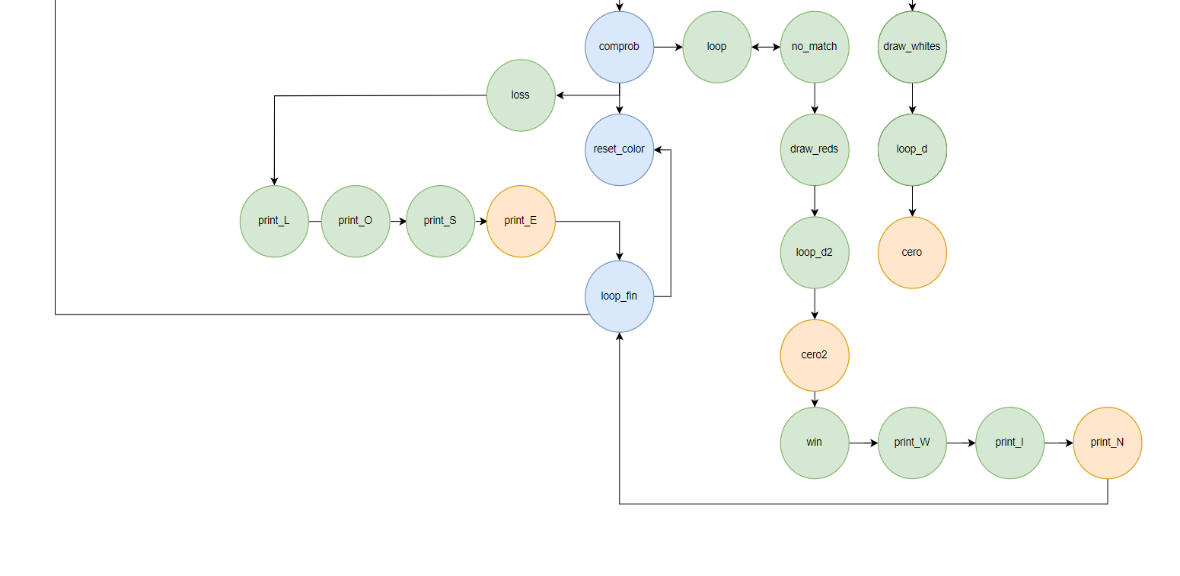
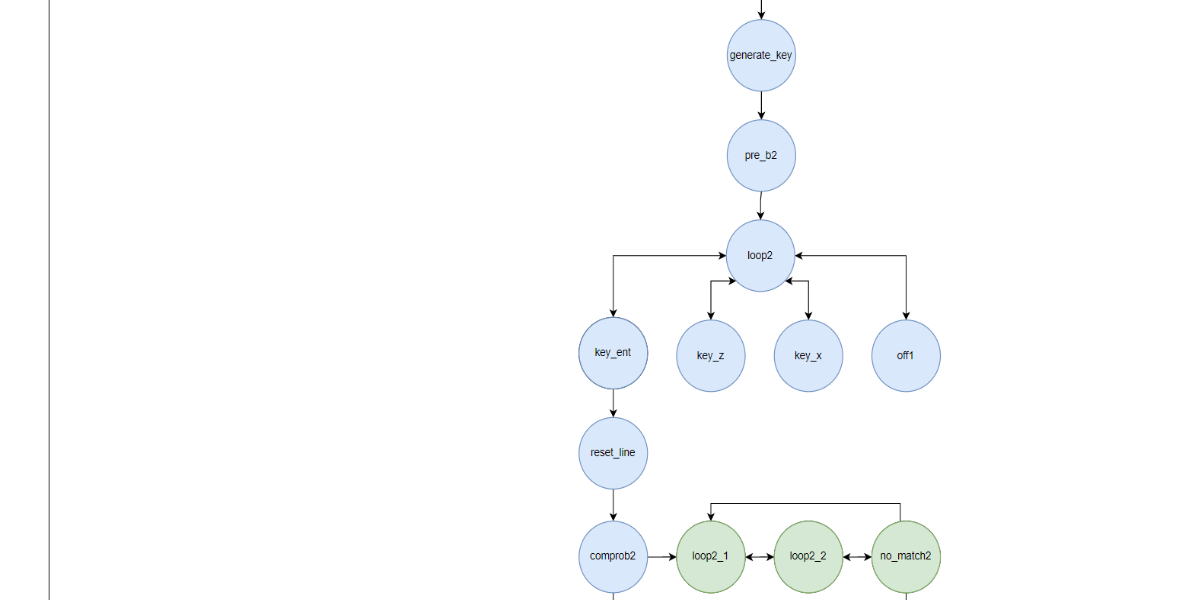
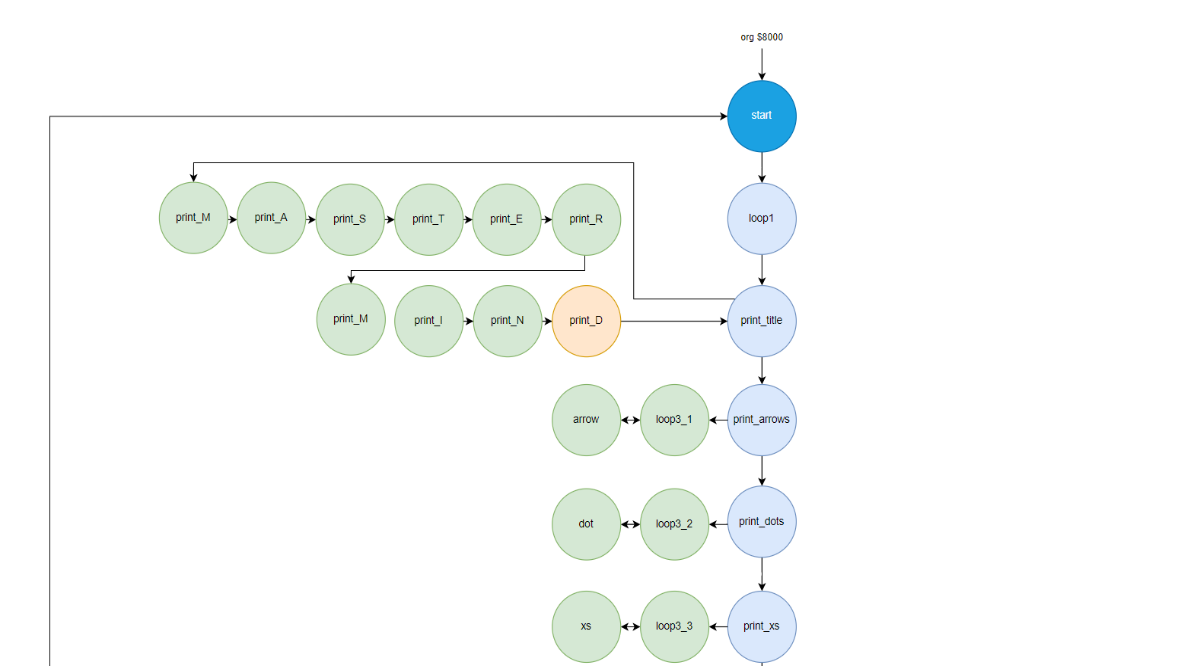
(Example of loss)

# Code

This part explains how the game works, each functionality and how the code flows from where to where.



## Code diagram



## Code function explanation

**Imagen que contiene reloj, computadora

Descripción generada automáticamenteTexto

Descripción generada automáticamente**

**org $8000:** is to start the game in that part of the memory.

**start:** initializes the registers to start to draw in the screen.

**loop1:** is the loop that prints all the screen with **BLUE**.

**PixelXYC:** changes the color of a pixel with the format in ”%” of the position X and Y chosen by C and B.

Texto

Descripción generada automáticamente

Diagrama

Descripción generada automáticamente**print\_title:** prints “MASTER MIND” in 11 squares where HL aims, calls letter by letter changing just the position between them.

Texto

Descripción generada automáticamente

**print\_M:** is the example of how it prints a letter with high resolution which prints line by line inside the square of a letter in binary. Each line in binary is printed and then moves down a line to pint the next one. This model is given by [J.M. Sanchez](#_Bibliography).

** 01100011 11 11**

**01110111 111 111**

**01111111     🡺 1111111 🡺**

**01101011 11 1 11**

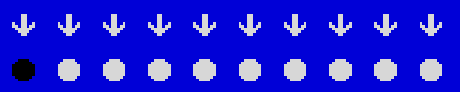
**01100011 11 11**

Texto

Descripción generada automáticamente**Diagrama

Descripción generada automáticamente**

**print\_arrows**, **print\_dots** and **print\_xs** are the 3 types of lines in the gameboard (🡺),(**O**) and (**X**) that are the position you are playing, the colors and the hints of the attempt.



Texto

Descripción generada automáticamente con confianza baja**Texto

Descripción generada automáticamente con confianza mediaprint\_arrows:** prints the first line of the board that is the line of **arrows** that indicates in which column you are playing. Works by printing arrow by arrow 10 times, each arrow in each **loop3\_1**, changing just the position with HL.

**arrow:** is the high-resolution arrow in binary like the letters, prints the arrow where HL is aiming.

**print\_dots:** prints the line of **dots** and works like **print\_arrows** just changing **arrows** to **dots**.

Texto

Descripción generada automáticamente**print\_xs**: prints the line of **Xs** and works like **print\_arrows** and **print\_dots** just changing **arrows** or **dots** to **Xs**.

Texto

Descripción generada automáticamente con confianza media

**Diagrama

Descripción generada automáticamentegenerate key:** this function is finally not done.

**pre\_b2:** this function just call key\_x to emulate pressing the key X to initialize the sequence of colors [like explained before](#_Game_Controls) the sequence is (**BLACK RED PINK GREEN BLUE YELLOW WHITE**) so the first color will be **BLACK.**

Texto

Descripción generada automáticamente**loop2:** is the principal loop in the code. First detects if nothing is pressed, if so, goes back, then detects which key is pressed is detects there´s one pressed.

**Texto

Descripción generada automáticamenteoff1:** changes d to 0 if there’s no key pressed.

**key\_x:** changes to the next color in the sequence of colors, if finds the value 1 means that color is already used, so goes to the next one to don’t repeat a color, if gets to the end of the sequence goes back to the beginning.

**key\_z:** works like **key\_x** but backwards, instead of adding 1 and get the next color, decrease 1 and takes the color before, if gets to the start of the sequence goes to the end.

Texto

Descripción generada automáticamente**key\_ent:** changes the value of the color in the sequence to 1 to choose the color, removes the BLINK property, and gives it to the next value, if it gets to the end of the input sequence goes to **reset\_line** to verify the attempt if not goes to the next dot, gives it the BLINK property and goes back to wait the input with **pre\_b2**.

Diagrama

Descripción generada automáticamente**Texto

Descripción generada automáticamentereset\_line:** resets when the attempt is done, first with **comprob2** that takes care of the **WHITE** hints and then **comprob** for the **RED** ones, after that removes the BLINK property from the last column, adds 1 to the try’s and if there’s 10 the game ends and goes to the **loss** screen, if don’t aims to the next column to prepare it to the next round giving the BLINK property to the arrow and the first dot, finally calls **reset\_color** to restart the colors of the sequence.

**Texto

Descripción generada automáticamente**

**comprob2:** compares the elements of key and input to know how many white **Xs** needs to be printed.

**loop2\_1:** loads in A the value of the **input** list, then IX aims to **key** list.

**loop2\_2:** this loop is the one that compares the 2 values, first loads the value of IX to C and compares A and C meaning that it’s comparing the values of key and input, at the end, depending on the result adds 1 to D or not, then goes to the next value of each list, at the end D will have the number of **WHITE** **Xs** and calls **draw\_writes** to print them.

Texto

Descripción generada automáticamente

**draw\_whites:** instead of adding **WHITE** **X**, this just remove the ones it doesn’t have to be in, first saves HL to don’t loose it, then aims to the last X on the column, if the number of **WHITE** **X** are 4 goes to the end because it doesn’t need to remove anything. Gives the FLASH property to the position its aiming and removes the **X**, then goes to the next one until is done.

Texto

Descripción generada automáticamente**Texto

Descripción generada automáticamentecomprob:** is like comprob2 but for the **RED** X, this time is easier because it has to compare with the same position in the lists. Compares the two values and if they are the same adds 1 to D and then the next position till it does it 4 times, when is done calls **draw\_reds** to print as many as it says D.

**draw\_reds:** prints the **X** to **RED**, first saves the value of HL to don’t loose it, then aims to the first **X** in the column, if there’s 4 **RED** X goes to the **win** phase directly, if not, paints as many **RED** X as D has (changes the color of the actual X and goes to the next one if must).

Diagrama

Descripción generada automáticamente

Texto

Descripción generada automáticamente

**Texto

Descripción generada automáticamentewin:** this function just shows the win screen, first aims to the top left corner to print there the message, then prints each letter with high resolution, and finaly changes it color to **RED** and calls **loop\_fin** where will wait for the restart.

**loss:** this function shows the loss screen like the win screen, aims to the top left corner, and prints each letter and change it the color to **RED** then and calls **loop\_fin** where will wait for the restart.

**Texto

Descripción generada automáticamente**Diagrama

Descripción generada automáticamente

**loop\_fin:** waits till the key V is pressed, then calls **reset\_color**, resets the try’s and goes back to **start** to play again.

This repeats till you close the emulation.



# Participation

**Alicia Custodia García Yuen:** 34%

Check of the trys and reset (**reset\_line**, **comprob**, **comprob2**…)

**Marie Estelle Melaine Pamen:** 34%

Flow of the game and changing of steps (**loop2**, **key\_ent**, **key\_x**, **key\_z** …).

**Rodrigo Sáez Escobar:** 32%

User´s manual, High Resolution functions(**print\_M**, **win**, **loss**, **PixelXYC**…)

# Bibliography

High resolution graphics model given by J.M. Sánchez