# DOCUMENTACIÓN DE LA ELABORACIÓN DEL PROYECTO DE MATERIAL DIDÁCTICO DE ALGEBRA

## DOCUMENTATION OF THE PREPARATION OF THE ALGEBRA DIDACTIC MATERIAL PROJECT

Mauricio García Galindo

Monterrey, Nuevo León, México

Junio 9 de 2024

Forma de citar este artículo en APA:

García Galindo, M. (2024). *Documentación de la elaboración del proyecto de material didáctico de algebra* [Archivo PDF].

https://maurdsh.github.io/documentacion-algebra/Documentaci%C3%B3n%20final%20v.%201.%201.pdf

## Resumen

En el presente trabajo se documentó la creación de un material didáctico elaborado con componentes electrónicos y su aplicación en un salón de 5° grado de primaria en la escuela Anexa Pablo Cantú Villarreal. La investigación necesaria en el área de programación se fundamentó en un paradigma funcional, con un enfoque en “que” se está haciendo y no en “como” se está haciendo. Se concluyó que este tipo de material puede ser funcional en el proceso de aprendizaje significativo de los niños.

## Palabras clave:

Display de 7 segmentos: Es una forma de representar caracteres en equipos electrónicos. Este componente está integrado por siete segmentos que se pueden encender o apagar individualmente. Cada segmento tiene la forma de una pequeña línea; Código: Los códigos de programación son aquellos que estructuran el lenguaje de programación, que a su vez es el encargado de garantizar el correcto funcionamiento de las aplicaciones o programas que permiten una buena comunicación entre el usuario y la computadora; Arduino UNO: Es una placa de microcontrolador de código abierto basado en el microchip ATmega328P y desarrollado por Arduino.cc. La placa está equipada con conjuntos de pines de E/S digitales y analógicas que pueden conectarse a varias placas de expansión y otros circuitos; Diagrama: Un diagrama electrónico, también conocido como un esquema eléctrico o esquemático es una representación pictórica de un circuito eléctrico; Chips: Pequeña pieza de material semiconductor que contiene múltiples circuitos integrados con los que se realizan numerosas funciones en computadoras y dispositivos electrónicos; Led: Diodo semiconductor que emite luz cuando se le aplica tensión; Resistencia: Es la oposición al flujo de corriente en un circuito eléctrico; Voltaje: Indicador de la capacidad de mover la electricidad; Tarjetas electrónicas con microprocesadores de bajo consumo programables: Las placas de programación o de desarrollo son dispositivos que cuentan con un microcontrolador (microchip) programable que puede ejecutar diferentes instrucciones y, por lo tanto, son adecuadas para crear todo tipo de dispositivos 'inteligentes', incluidos robots.

## Abstract

In this work, the creation of a teaching material composed of electronic components and its application in a 5th grade primary school classroom at the Anexa Pablo Cantú Villarreal school was documented. The necessary research in the area of programming was based on a functional paradigm, with a focus on “what” is being done and not on “how” it is being done. It was concluded that this type of material can be functional in the significant learning process of children.

## Keywords:

7-segment display: It is a way of representing characters on electronic equipment. This component is made up of seven segments that can be turned on or off individually. Each segment is shaped like a small line; Code: Programming codes are those that structure the programming language, which in turn is responsible for guaranteeing the correct functioning of the applications or programs that allow good communication between the user and the computer; Arduino UNO: It is an open-

source microcontroller board based on the ATmega328P microchip and developed by Arduino.cc. The board is equipped with sets of digital and analog I/O pins that can connect to various expansion boards and other circuits; Diagram: An electronic diagram, also known as an electrical or schematic diagram, is a pictorial representation of an electrical circuit; Chips: Small piece of semiconductor material that contains multiple integrated circuits that perform numerous functions in computers and electronic devices; Led: Semiconductor diode that emits light when voltage is applied; Resistance: It is the opposition to the flow of current in an electrical circuit; Voltage: Indicator of the ability to move electricity; Electronic cards with programmable low-consumption microprocessors: Programming or development boards are devices that have a programmable microcontroller (microchip) that can execute different instructions and, therefore, are suitable for creating all types of 'smart' devices, including robots.

• Estudiante de Licenciatura en Educación Primaria. Escuela Normal Miguel F. Martínez Centenaria y Benemérita. G-mail: mauricio.garcia23@mfm.edu.mx

# CONTENIDO

# CONTENTS

# Introducción

> La educación, como proceso fundamental en la vida de los hombres, implica aprender y desaprender constantemente. Consta de una variedad de recursos y estrategias que facilitan y producen aprendizajes en el sujeto. Por ello, las instituciones educativas permiten el acceso a materiales didácticos para que los docentes[1] utilicen en el aula de clase, de tal forma que propicie una educación más dinámica y eficaz. De esta manera, la implementación de dichos materiales en los procesos escolares, conlleva una transmisión de conocimientos. A partir de esta dinámica se le autoriza al estudiante interactuar de manera más práctica y lúdica con los saberes requeridos en su formación. (Manrique Orozco y Gallego Henao, 2013, pp 102)

La dirección de este proyecto nace de la actividad de la elaboración de un material didáctico. En mi lluvia de ideas aterrice a realizar el material didáctico con componentes electrónicos para que el alumno pueda tener un interés similar al que pueden tener con algún celular, tablet, laptop, etc.

Con el área de oportunidad resultante de la aplicación del test de matemáticas en mi primera jornada de este semestre el tema al que enfocaría el material didáctico sería a la comparación de números, la cual más adelante se convirtió en comparación de cantidades de consumo agua (para poder relacionar el material con la planeación del proyecto de agua que se aplicara en la segunda jornada).

**Test de algebra**

https://docs.google.com/document/d/1naQPnuvXJ6U-_GhM-PXk-DZ7s8clClQhefD-DgslSaw/edit?usp=sharing

## Primer diseño

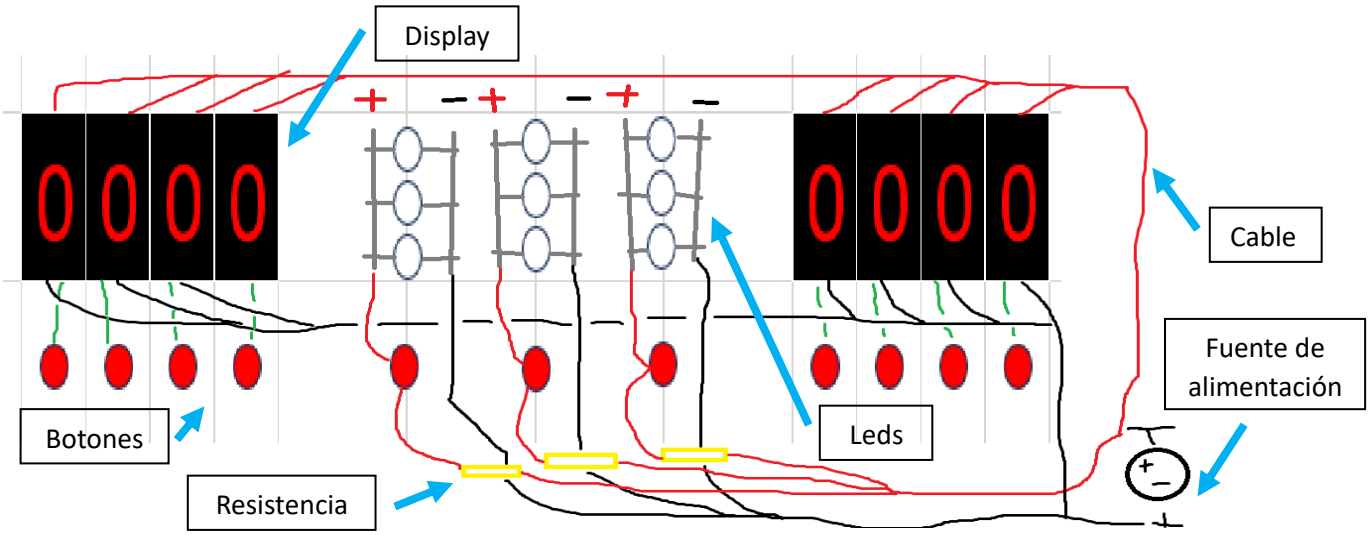


*Imagen 1. Diseño 1.*

Este fue el primer diseño, mi idea era utilizar unos componentes llamados **"display de 7 segmentos",** para con los botones poder mostrar una cantidad numérica en ambos lados para así con los 3 botones centrales encender los leds de en medio para marcar una igualdad, o los laterales señalando la cantidad mayor o menor dependiendo cual sea el ejercicio aplicado.

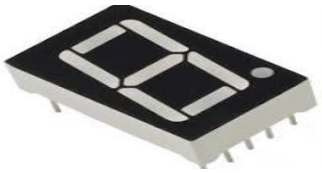

*Imagen 2. Display de 7.*

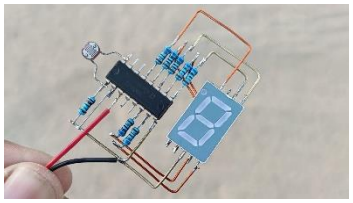

*Imagen 3. Circuito con display.*

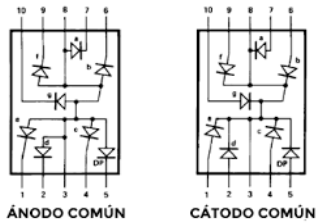


*Imagen 4. Diagrama del display.*

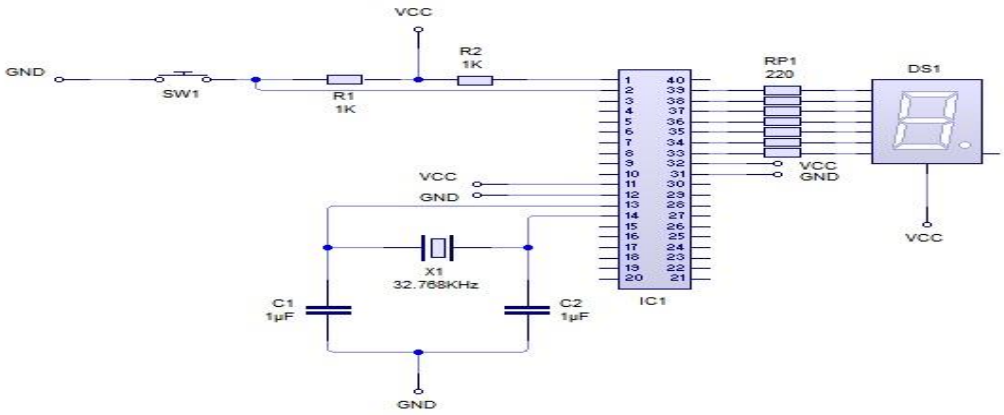


*Imagen 5. **Diagrama** de un solo contador (mi diseño ocupa mínimo 4 contadores y el diseño que realice lleva 8).*

El resto de componentes del diseño eran cables para la corriente positiva y negativa, una fuente de alimentación, la cual considere que fuera alguna batería recargable (o pilas AA, AAA, de 9v, etc.), también se utilizarían **leds**, **resistencias** y botones.

La **idea** que diseñé fue **descartada rápidamente**, ya que presentaba **muchos errores eléctricos** que no había considerado, como la incompatibilidad en los regresos de corriente eléctrica (cargas negativas) debido a que cada componente utilizaría una velocidad (voltaje) diferente la cual generaría un problema al querer regresar diferentes **voltajes** por un mismo medio, el no considerar el uso de **chips** para programar las pantallas, la gran cantidad de cables…., a parte elaborar una sola pieza contadora requeriría una cantidad exagerada de soldadura, ya que como en la imagen anterior el poder programar un solo display requiere muchas conexiones/soldaduras a un chip.

Si mi diseño contenía 8 displays y aunque lo redujera a un mínimo de 4 displays para poder comparar números de 2 dígitos requeriría realizar esa tarea de soldadura 4 veces más a parte la soldadura de los leds y los botones y la fuente de energía.

## Segundo diseño

Al concluir que mi mayor problemática en mi idea anterior era la soldadura de los chips con las pantallas, recurrí a investigar sobre **tarjetas electrónicas con microprocesadores de bajo consumo programables.**

**¿Por qué recurrir a placas con chips programables?** Porque en mi **primera idea** los **chips** venían **programados de fabrica** (más su complejidad a la hora de realizar las conexiones).

Después de una investigación entre la gran variedad de este tipo de tarjetas programables me decidí por utilizar **Arduino UNO.**


*Imagen 6. Arduino UNO.*

Esta tarjeta es programable desde cualquier computadora descargando el software **Arduino IDE,** el cual es el programa oficial para desarrollar código y cargarlo en una placa Arduino.

Decidí utilizar una pantalla táctil que emboné en los huecos de los pines de salida de la tarjeta, para entonces mi idea inicial adaptarla a un **código** con el cual por medio de la pantalla táctil pudiera darle vida a la comparación de cantidades de consumo de agua.

*Imagen 7. Arduino UNO con una pantalla táctil.*

## Código

### ChatGPT

Para poder elaborar el código recurrí a una larga conversación con ChatGPT, herramienta de la cual obtuve la estructura general del código que le daría vida al programa que ejecutaría el juego de la comparación de cantidades de agua.

```cpp
#include <MCUFRIEND_kbv.h>
#include <TouchScreen.h>
#include <Adafruit_GFX.h>
#include <Arduino.h>

// Calibration constants for the touchscreen
const int XP = 6, XM = A2, YP = A1, YM = 7; // Pins for touchscreen
const int TS_LEFT = 175, TS_RT = 903, TS_TOP = 938, TS_BOT = 185; // Calibration bour
TouchScreen ts = TouchScreen(XP, YP, XM, YM, 300); // Pressure threshold

// Touch pressure thresholds
const int MINPRESSURE = 200;
const int MAXPRESSURE = 1000;

// Screen dimensions
const int screenWidth = 240;
const int screenHeight = 320;

// Colors
const uint16_t BLACK = 0x0000;
const uint16th WHITE = 0xFFFF;
const uint16th RED = 0xF800;
const uint16th GREEN = 0x07E0;
const uint16th YELLOW = 0xFFE0;
// TFT display
MCUFRIEND_kbv tft;

// Game variables
int score = 0;
int level = 0;
int num1, num2; // Numbers to compare
int expectedBucket; // 0 = '+', 1 = '=', 2 = '-'

// Function to get touch input
TSPoint getTouch() {
    TSPoint p = ts.getPoint();
    pinMode(XM, OUTPUT); // Reset pin modes after reading
    pinMode(YP, OUTPUT);
    return p;
}

// Function to draw the game field
void drawGameField() {
    tft.fillScreen(WHITE); // Clear the screen
    tft.setTextSize(2);
    tft.setTextColor(BLACK);

    // Draw the numbers to compare and align to the left
    int16_t leftMargin = 10; // Left margin for text alignment
    tft.setCursor(leftMargin, 50); // Set cursor to the left
    tft.print("Compare: ");
    tft.print(num1);
```

Lo subrayado en color amarillo, son los errores de redacción que arrojo el código.

*Imagen 8. Primera parte del código de ChatGPT*

```
    tft.print(" ? ");
    tft.print(num2);

    // Draw the buckets at the bottom
    const int16_t bucketWidth = 60;
    const int16th bucketHeight = 30;
    const int16th bucketStartY = screenHeight - bucketHeight;

    // Calculate bucket spacing and positions
    int16th bucketGap = (screenWidth - (3 * bucketWidth)) / 4; // Gap between buckets
    tft.fillRect(bucketGap, bucketStartY, bucketWidth, bucketHeight, RED);
    tft.fillRect(2 * bucketGap + bucketWidth, bucketStartY, bucketWidth, bucketHeight
    tft.fillRect(3 * bucketGap + 2 * bucketWidth, bucketStartY, bucketWidth, bucketHe

    // Label the buckets
    tft.setCursor(bucketGap + 15, bucketStartY + 5);
    tft.print("+");
    tft.setCursor(2 * bucketGap + bucketWidth + 15, bucketStartY + 5);
    tft.print("=");
    tft.setCursor(3 * bucketGap + 2 * bucketWidth + 15, bucketStartY + 5);
    tft.print("-");
}

// Function to generate a new math problem
void generateProblem() {
    num1 = random(1, 100);
    num2 = random(1, 100);
    expectedBucket = random(0, 3); // 0 = '+', 1 = '=', 2 = '-'
}

void setup() {
    Serial.begin(9600); // Initialize serial communication
    tft.reset();
    int16th id = tft.readID();
    tft.begin(id); // Initialize the display
    randomSeed(analogRead(A0));

    generateProblem();
    drawGameField();
}

void loop() {
    TSPoint p = getTouch();

    if (p.z > MINPRESSURE && p.z < MAXPRESSURE) { // Valid touch pressure
        int16th x = map(p.x, TS_LEFT, TS_RT, 0, screenWidth);
        int16th y = map(p.y, TS_TOP, TS_BOT, screenHeight);

        if (y >= screenHeight - 40) { // Touch is in the bucket area
            const int16th bucketWidth = 60;
            int16th bucketGap = (screenWidth - (3 * bucketWidth)) / 4;

            if (x >= bucketGap and x < bucketGap + bucketWidth) { // "+" bucket
                if (expectedBucket == 0) {
```

*Imagen 9. Segunda parte del código de ChatGPT.*

```
                    score += 100;
                }
            } else if (x >= 2 * bucketGap + bucketWidth and x < 2 * bucketGap + 2 *
                if (expectedBucket == 1) {
                    score += 100;
                }
            } else if (x >= 3 * bucketGap + 2 * bucketWidth and x < 3 * bucketGap +
                if (expectedBucket == 2) {
                    score += 100;
                }
            }

            level++;

            if (level > 30) { // End of game after 30 levels
                tft.fillScreen(WHITE);
                tft.setCursor(60, 100);
                tft.setTextSize(3);
                tft.setTextColor(BLACK);
                tft.print("Game Over!");
                tft.setCursor(60, 150);
                tft.print("Final Score: ");
                tft.print(score);
                delay(5000);
                level = 0;
                score = 0;
                generateProblem();
                drawGameField();
            } else {
                generateProblem();
                drawGameField();
            }
        }
    }

    delay(100);
}
```

*Imagen 10. Tercera parte del código de ChatGPT.*

Al código anterior le corregí errores de redacción y lo termine de adecuar a la personalización que yo le tenía pensado al programa que se ejecutaría en la pantalla.

**Mi código**

```
#include <MCUFRIEND_kbv.h>
#include <TouchScreen.h>
#include <Adafruit_GFX.h>
#include <Arduino.h>
```

```
// Calibration constants for the touchscreen
const int XP = 6, XM = A2, YP = A1, YM = 7; // Pins for touchscreen
const int TS_LEFT = 175, TS_RT = 903, TS_TOP = 938, TS_BOT = 185; // Calibration boundaries
TouchScreen ts = TouchScreen(XP, YP, XM, YM, 300); // Pressure threshold

// Touch pressure thresholds
const int MINPRESSURE = 200;
const int MAXPRESSURE = 1000;

// Screen dimensions
const int screenWidth = 240;
const int screenHeight = 320;

// Colors
const uint16_t BLACK = 0x0000;
const uint16_t WHITE = 0xFFFF;
const uint16_t RED = 0xF800;
const uint16_t GREEN = 0x07E0;
const uint16_t YELLOW = 0xFFE0;
const uint16_t LIGHT_BLUE = 0x07FF;
const uint16_t BLUE = 0x001F;
const uint16_t BROWN = 0x964B00;
const uint16_t GRAY = 0x2d2b2b;

// TFT display
MCUFRIEND_kbv tft;

// Game variables
int score = 0;
int level = 0;
int num1, num2; // Numbers to compare
int expectedBucket; // 0 = '+', 1 = '=', 2 = '-'

// Function to get touch input
TSPoint getTouch() {
```

```
    TSPoint p = ts.getPoint();

   pinMode(XM, OUTPUT); // Reset pin modes after reading
    pinMode(YP, OUTPUT);
    return p;
}

// Function to draw the game field
void drawGameField() {
    tft.fillScreen(BLACK); // Clear the screen
    tft.setTextSize(2);
    tft.setTextColor(WHITE);

    // Draw the numbers to compare and align to the left
    int16_t leftMargin = 0; // Left margin for text alignment
    tft.setCursor(leftMargin, 50); // Set cursor to the left
    tft.print("Compara: ");
    tft.print(num1);
    tft.setTextColor(GREEN);
    tft.print("ml");
    tft.setTextColor(WHITE);
    tft.print(" ? ");
    tft.setTextColor(WHITE);
    tft.print(num2);
    tft.setTextColor(GREEN);
    tft.print("ml");
    tft.setTextColor(GRAY);
    tft.print("                    ####      #####");
    tft.setTextColor(GRAY);
    tft.print("   #");
    tft.setTextColor(LIGHT_BLUE);
    tft.print("####");
    tft.setTextColor(GRAY);
    tft.print("##    #");
    tft.setTextColor(LIGHT_BLUE);
    tft.print("####");
```

Lo subrayado en color amarillo, son las correcciones que hice al código y las nuevas líneas que agregué.

```
tft.setTextColor(GRAY);
tft.print("#  #");
tft.setTextColor(LIGHT_BLUE);
tft.print("#######");
tft.setTextColor(GRAY);
tft.print("##  #");
tft.setTextColor(LIGHT_BLUE);
tft.print("##");
tft.setTextColor(GRAY);
tft.print("#");
tft.setTextColor(LIGHT_BLUE);
tft.print("#");
tft.setTextColor(GRAY);
tft.print("# #");
tft.setTextColor(BROWN);
tft.print("#");
tft.setTextColor(LIGHT_BLUE);
tft.print("###");
tft.setTextColor(WHITE);
tft.print("#");
tft.setTextColor(GRAY);
tft.print("#");
tft.setTextColor(LIGHT_BLUE);
tft.print("##");
tft.setTextColor(GRAY);
tft.print("#");
tft.setTextColor(BROWN);
tft.print("#");
tft.setTextColor(GRAY);
tft.print("##");
tft.setTextColor(LIGHT_BLUE);
tft.print("##");
tft.setTextColor(GRAY);
tft.print("#");
tft.setTextColor(LIGHT_BLUE);
tft.print("#");
```

```
tft.setTextColor(GRAY);
tft.print("#");
tft.setTextColor(GRAY);
tft.print("  #");
tft.setTextColor(LIGHT_BLUE);
tft.print("####");
tft.setTextColor(GRAY);
tft.print("#");
tft.setTextColor(BROWN);
tft.print("#");
tft.setTextColor(LIGHT_BLUE);
tft.print("##");
tft.setTextColor(WHITE);
tft.print("#");
tft.setTextColor(BROWN);
tft.print("###");
tft.setTextColor(GRAY);
tft.print("#");
tft.setTextColor(LIGHT_BLUE);
tft.print("#");
tft.setTextColor(GRAY);
tft.print("##");
tft.setTextColor(GRAY);
tft.print("   ##");
tft.setTextColor(LIGHT_BLUE);
tft.print("######");
tft.setTextColor(GRAY);
tft.print("#");
tft.setTextColor(LIGHT_BLUE);
tft.print("#");
tft.setTextColor(WHITE);
tft.print("#");
tft.setTextColor(BROWN);
tft.print("###");
tft.setTextColor(GRAY);
tft.print("#");
```

```
tft.setTextColor(GRAY);
tft.print("     #");
tft.setTextColor(LIGHT_BLUE);
tft.print("#");
tft.setTextColor(GRAY);
tft.print("######");
tft.setTextColor(LIGHT_BLUE);
tft.print("##");
tft.setTextColor(GRAY);
tft.print("#");
tft.setTextColor(WHITE);
tft.print("#");
tft.setTextColor(BROWN);
tft.print("##");
tft.setTextColor(GRAY);
tft.print("#      ##");
tft.setTextColor(YELLOW);
tft.print("#####");
tft.setTextColor(GRAY);
tft.print("###");
tft.setTextColor(WHITE);
tft.print("###");
tft.setTextColor(GRAY);
tft.print("#       ###");
tft.setTextColor(YELLOW);
tft.print("#####");
tft.setTextColor(LIGHT_BLUE);
tft.print("#");
tft.setTextColor(GRAY);
tft.print("##          #");
tft.setTextColor(LIGHT_BLUE);
tft.print("#");
tft.setTextColor(GRAY);
tft.print("####");
tft.setTextColor(LIGHT_BLUE);
tft.print("###");
```

```
    tft.setTextColor(GRAY);
    tft.print("#          ##   ###");

    // Draw the buckets at the bottom
    const int16_t bucketWidth = 60;
    const int16_t bucketHeight = 30;
    const int16_t bucketStartY = screenHeight - bucketHeight;

    // Calculate bucket spacing and positions
    int16_t bucketGap = (screenWidth - (3 * bucketWidth)) / 4; // Gap between buckets
    tft.fillRect(bucketGap, bucketStartY, bucketWidth, bucketHeight, RED);
    tft.fillRect(2 * bucketGap + bucketWidth, bucketStartY, bucketWidth, bucketHeight,
GREEN);
    tft.fillRect(3 * bucketGap + 2 * bucketWidth, bucketStartY, bucketWidth, bucketHeight,
YELLOW);

    // Label the buckets
    tft.setTextColor(BLACK);
    tft.setCursor(bucketGap + 25, bucketStartY + 9);
    tft.print("+");
    tft.setCursor(2 * bucketGap + bucketWidth + 25, bucketStartY + 9);
    tft.print("=");
    tft.setCursor(3 * bucketGap + 2 * bucketWidth + 25, bucketStartY + 9);
    tft.print("-");
}

// Function to generate a new math problem
void generateProblem() {
    num1 = random(1, 100);
    num2 = random(1, 100);

    expectedBucket = random(0, 3); // 0 = '+', 1 = '=', 2 = '-'
}

void setup() {
    Serial.begin(9600); // Initialize serial communication
```

```
    tft.reset();
    int16_t id = tft.readID();
    tft.begin(id); // Initialize the display
    randomSeed(analogRead(A0));

    generateProblem();
    drawGameField();
}

void loop() {
    TSPoint p = getTouch();

    if (p.z > MINPRESSURE && p.z < MAXPRESSURE) { // Valid touch pressure
        int16_t x = map(p.x, TS_LEFT, TS_RT, 0, screenWidth);
        int16_t y = map(p.y, TS_TOP, TS_BOT, 0, screenHeight);

        if (y >= screenHeight - 40) { // Touch is in the bucket area
            const int16_t bucketWidth = 60;
            int16_t bucketGap = (screenWidth - (3 * bucketWidth)) / 4;

            if (x >= bucketGap and x < bucketGap + bucketWidth) { // "+" bucket
                if (expectedBucket == 0) {
                    score += 100;
                }
            } else if (x >= 2 * bucketGap + bucketWidth and x < 2 * bucketGap + 2 *
bucketWidth) { // "=" bucket
                if (expectedBucket == 1) {
                    score += 100;
                }
            } else if (x >= 3 * bucketGap + 2 * bucketWidth and x < 3 * bucketGap + 3 *
bucketWidth) { // "-" bucket
                if (expectedBucket == 2) {
                    score += 100;
                }
            }
```

```
level++;

if (level > 50) { // End of game after 30 levels
  tft.fillScreen(BLACK);
  tft.setCursor(0, 50);
  tft.setTextSize(3);
  tft.setTextColor(WHITE);
  tft.print("Fin del juego");
  tft.setCursor(0, 100);
  tft.print("Puntaje Final:   ");
  tft.print(score);
  tft.setTextSize(2);
  tft.setTextColor(GRAY);
  tft.print("                                                ########          #");
  tft.setTextColor(RED);
  tft.print("########");
  tft.setTextColor(GRAY);
  tft.print("#         #");
  tft.setTextColor(RED);
  tft.print("###");
  tft.setTextColor(GRAY);
  tft.print("####");
  tft.setTextColor(RED);
  tft.print("###");
  tft.setTextColor(GRAY);
  tft.print("#        #####");
  tft.setTextColor(WHITE);
  tft.print("##");
  tft.setTextColor(GRAY);
  tft.print("#####        #");
  tft.setTextColor(WHITE);
  tft.print("###");
  tft.setTextColor(GRAY);
  tft.print("####");
  tft.setTextColor(WHITE);
  tft.print("###");
```

```
            tft.setTextColor(GRAY);
            tft.print("#        #");
            tft.setTextColor(WHITE);
            tft.print("########");
            tft.setTextColor(GRAY);
            tft.print("#        ########");

            delay(5000);
            level = 0;
            score = 0;

        } else {
            generateProblem();
            drawGameField();
        }
    }
}

  delay(100);
}
```

El código final funcionó muy bien, eran 30 problemas por juego y al final daba un puntaje en base a los aciertos. El programa consistía en escoger por medio del táctil de la pantalla si el primero de los números era mayor, igual o menor al segundo (incluí al lado de los números "ml" para que fuera una comparación de cantidades de agua). En medio de la pantalla había un dibujo de un Pokémon tipo agua, ya que la intención era relacionar el material didáctico con Pokémon, debido a que le gusta mucho a la mayoría de los niños del salón.


*Imagen 12. Material funcional con diseño squirtle.*


*Imagen 13. Material funcional con diseño squirtle.*

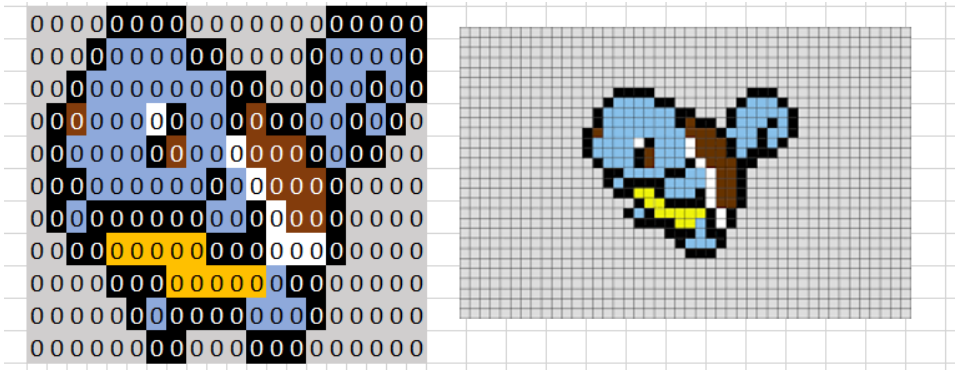

*Imagen 14. Diseño de squirtle para su programación.*


*Imagen 15. Diseño vaporeon.*

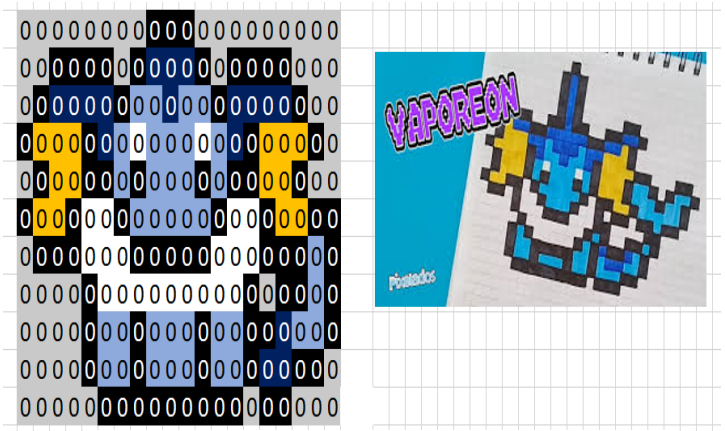


*Imagen 16. Diseño de vaporeon para su programación.*

## Excel de los diseños

https://docs.google.com/spreadsheets/d/12CINRODYC7dUEvrZhjZMYAmf4_wLC6FCM804K8y5bMw/edit?usp=sharing

## Soldadura

Realicé la soldadura para el desarrollo de 2 dispositivos, la intención de la aplicación era dividir el salón en 2 equipos para que en una ronda se fueran pasando el dispositivo mientras cada quien resolvía una interrogante y el equipo que al final obtuviera más puntos ganaría un sticker de una estrella (estrategia que mi maestra de jornada suele utilizar).

La soldadura fue para darle autonomía a cada dispositivo, se le agregó una batería recargable y un modulo de carga para que no tuviera que depender de un enchufe externo para funcionar como se hacía anteriormente.

Elaboré el siguiente diagrama para realizar la soldadura de la batería y otros componentes.

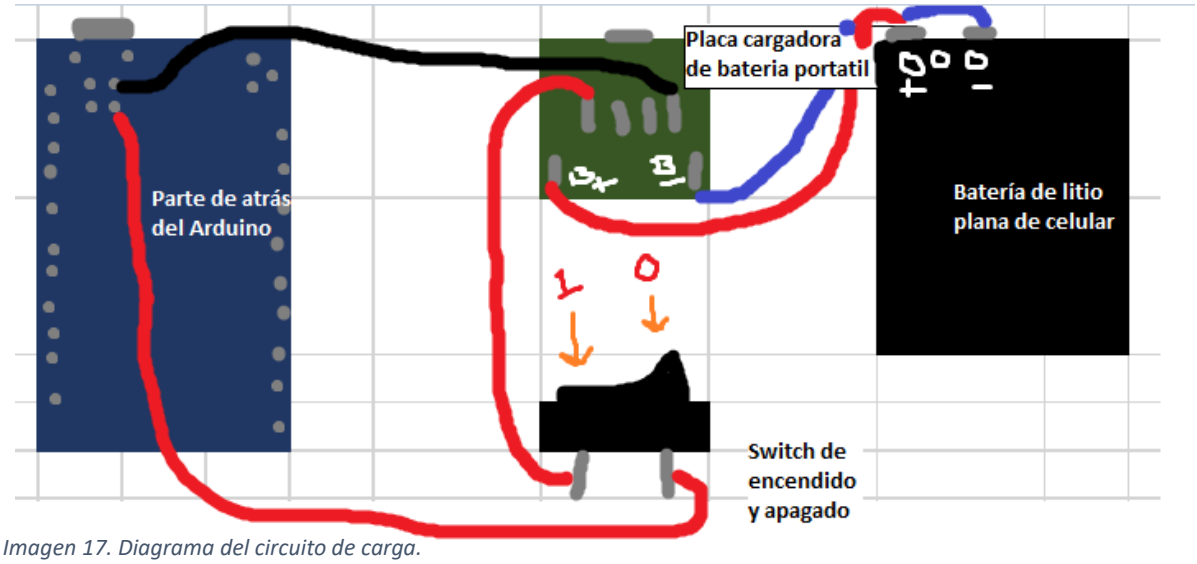


*Imagen 17. Diagrama del circuito de carga.*

**Vídeo de elaboración:**

https://youtu.be/7I0DL7sW58w

## Resultados:

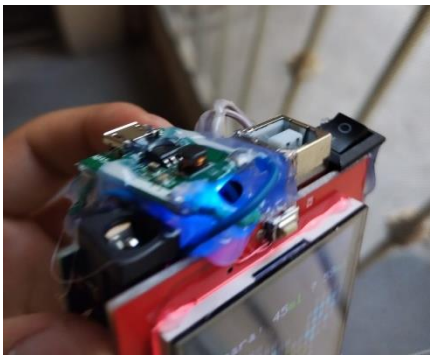

*Imagen 18. Circuito de pila recargable.*

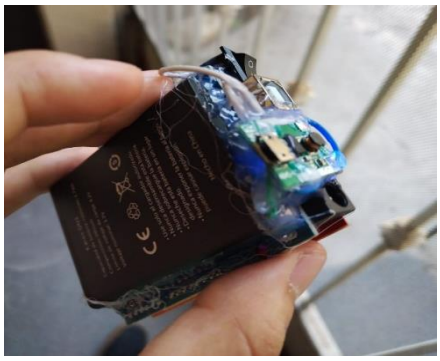

*Imagen 19. Circuito de pila recargable.*

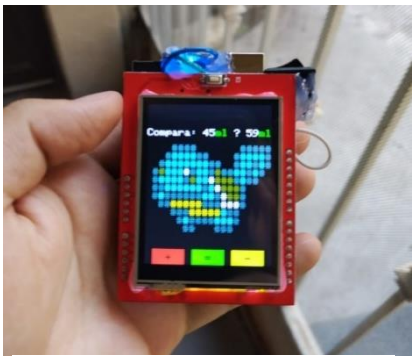


*Imagen 20. Material funcional con diseño squirtle.*

## Carcasa

La carcasa decidí elaborarla con fomi moldeable, ya que este secaría al cabo de un rato y ya sobre este material pintaría cuidadosamente algún diseño inspirado en Pokémon. Como el fomi es un material aislante supuse que el fomi moldeable también.

Los dispositivos… … … dejaron de funcionar después de realizarles la carcasa. Pienso que el fomi de algún modo estropeo algunos componentes al entrar en contacto, ya que antes de forrar los dispositivos con el fomi ambos funcionaban perfectamente, retenían la carga en sus baterías y prendían al encendido del switch.

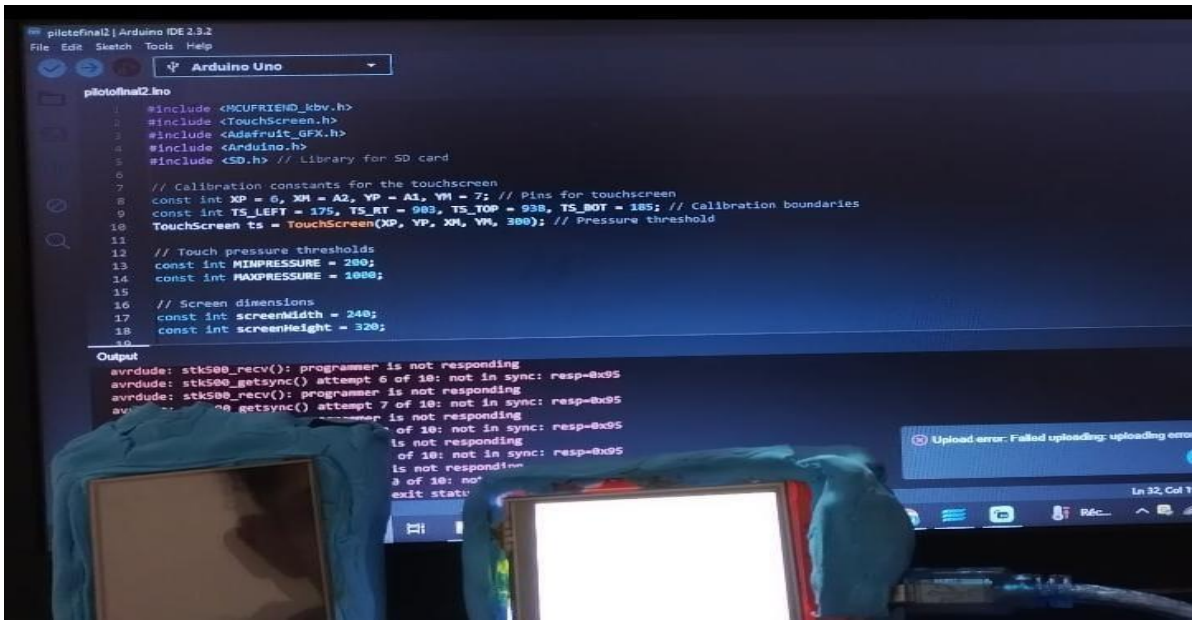


*Imagen 21. Efecto Fomi.*
*Un dispositivo incapaz de prender por batería y otro encendido por cable. Ambos incapaces de ejecutar el código previamente desarrollado y probado. De fondo un error al tratar de volver a cargar el código a las placas Arduino.*

El fomi no causo ningún corto circuito del que me pudiera dar cuenta ya que de ser ese el caso ninguna de las 2 pantallas seguiría encendiendo, pero ambas por cable prendían. No entiendo de que manera afecto el fomi a algunos de los componentes que simplemente ya no se pudo reprogramar ninguna de las placas y por ende ya no se dio imagen. En un momento una pantalla si logro ejecutar el código por un breve tiempo (luego se volvió a desprogramar) pero el táctil ya no le funciono.

## Restauración

Se quitó todo el fomi moldeable y se limpio intensivamente cada pieza con alcohol isopropílico el cual es un alcohol de limpieza especial para componentes electrónicos.

Se probo directo a la luz por USB y funcionó correctamente otra vez. Se restauraron los circuitos de soldadura de las pilas y se remplazaron los módulos de carga y pilas que se estropearon.

Se volvió a realizar una carcasa en este caso de silicón y se pintó con pintura inflable.

**Vídeo de restauración**

https://youtu.be/-8umJ1611AQ

**Dispositivo**


*Imagen 22. Dispositivo final 1.*


*Imagen 23. Dispositivo final 2.*

# Resultados de la aplicación

**Funcionamiento del material didáctico**

https://youtu.be/gXmp7RtyUfE

**Aplicación**

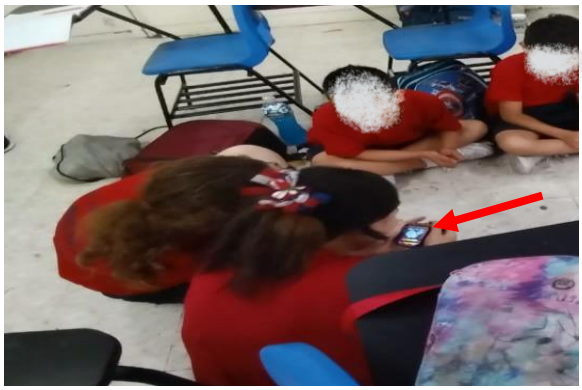

*Imagen 24. Aplicación del material*

*Imagen 25. Aplicación del material*

A los niños se les hizo muy novedoso el material, realicé 2 rondas, la primera para que se relacionaran con la dinámica de la actividad y con el funcionamiento de la pantalla la cual funciona a presión por ser de cristal líquido y no al mínimo contacto como la de un celular.

Durante esa ronda de prueba les fui explicando la finalidad del material, las instrucciones y el modo de juego. Ya la segunda ronda fue en modo de carrera para ver cual equipo terminaba primero y con el mayor puntaje. Los puntajes obtenidos en esta ronda fueron muy prometedores, un equipo obtuvo 1,200 y el otro 2,100 (el máximo era 3000), los resultados fueron positivos para haber sido la primera aplicación de este tipo de material didáctico con los niños.

En conclusión, la actividad les sirvió como práctica de habilidad mental por la cuestión de la carrera y esto de la mano de la comparación de números, si les gusto el material, pero fuera de este juego no se pudo retomar.

El material si es reutilizable, solo hace falta desarrollar el software o los códigos para poder seguirle sacando el máximo provecho en la misma u otras áreas como español, geografía, ciencias, exámenes, diagnósticos, etc.

## Referencias

1. colaboradores de Wikipedia. (2024, 4 junio). Arduino uno. *Wikipedia, la Enciclopedia Libre*. https://es.wikipedia.org/wiki/Arduino_Uno

2. colaboradores de Wikipedia. (2024a, mayo 13). Diagrama electrónico. *Wikipedia, la Enciclopedia Libre*. https://es.wikipedia.org/wiki/Diagrama_electr%C3%B3nico

3. Fluke. (s. f.). ¿Qué es la resistencia? *Fluke*. https://www.fluke.com/es-sv/informacion/blog/electrica/que-es-la-resistencia

4. Manrique Orozco, A. M., & Gallego Henao, A. M. (2013). El material didáctico para la construcción de aprendizajes significativos. *Revista Colombiana de Ciencias Sociales, 4*(1), 101-108.

5. OpenAI. (s. f.). ChatGPT. https://openai.com/chatgpt

6. Real Academia Española. (2014). *Diccionario de la lengua española*. Madrid, España.

7. Reyes, I. C. (2024, 21 marzo). 5 tipos de paradigmas de programación | CognosOnline México. *CognosOnline*. https://cognosonline.com/mx/blog-mx/que-son-paradigmas-de-programacion/#:~:text=En%20programaci%C3%B3n%2C%20se%20conocen%20como,llegar%20a%20los%20resultados%20esperados.