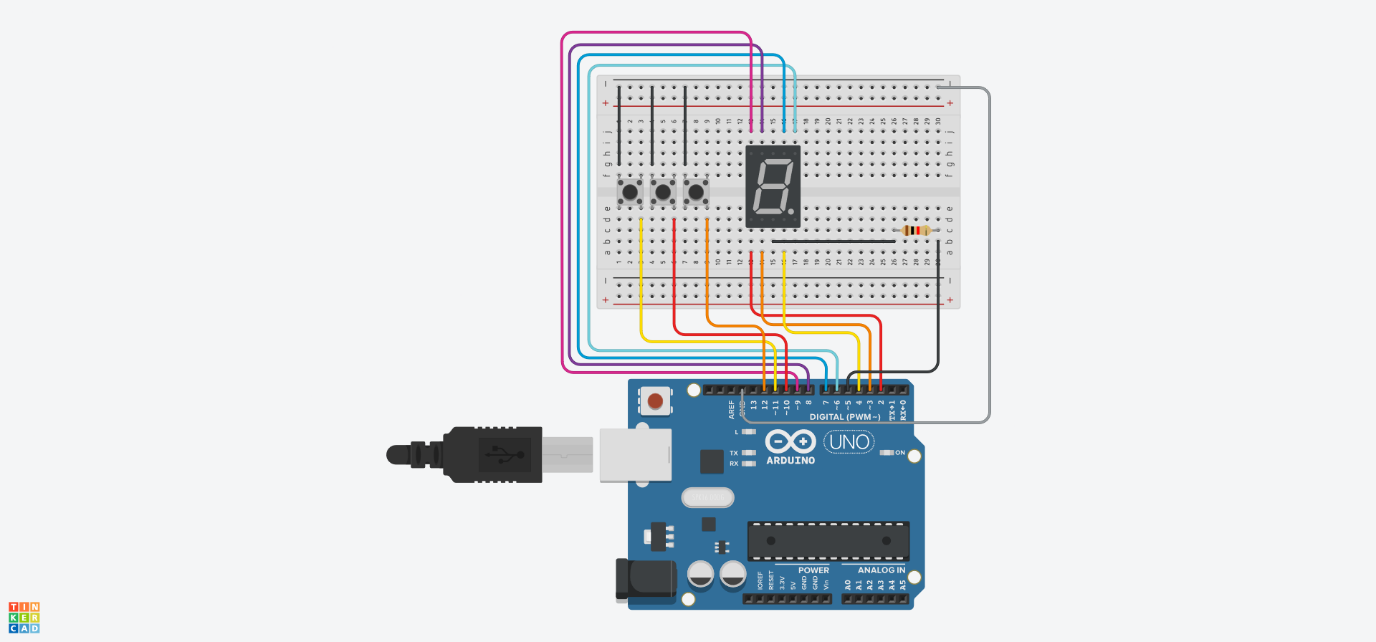
**Schematic diagram of the assembled circuit:**



**Program code in C++ using the Arduino IDE:**

// constants

const int pinCount **=** 7**;**

const int numberCount **=** 10**;**

const int maxNumber **=** 9**;**

const int minNumber **=** 0**;**

const int maxBrightness **=** 255**;**

const int minBrightness **=** 0**;**

const int brightnessChangeStep **=** 50**;**

// pins

// A B C D E F G

const int pins**[**pinCount**]** **=** **{**7**,** 6**,** 4**,** 3**,** 2**,** 8**,** 9**};**

const int increaseButtonPin **=** 10**;**

const int decreaseButtonPin **=** 11**;**

const int brightnessButtonPin **=** 12**;**

const int brightnessPin **=** 5**;**

// 7-segment display patterns for numbers 0-9

const int numbers**[**numberCount**][**pinCount**]** **=** **{**

// A B C D E F G

**{**0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 1**},** // 0

**{**1**,** 0**,** 0**,** 1**,** 1**,** 1**,** 1**},** // 1

**{**0**,** 0**,** 1**,** 0**,** 0**,** 1**,** 0**},** // 2

**{**0**,** 0**,** 0**,** 0**,** 1**,** 1**,** 0**},** // 3

**{**1**,** 0**,** 0**,** 1**,** 1**,** 0**,** 0**},** // 4

**{**0**,** 1**,** 0**,** 0**,** 1**,** 0**,** 0**},** // 5

**{**0**,** 1**,** 0**,** 0**,** 0**,** 0**,** 0**},** // 6

**{**0**,** 0**,** 0**,** 1**,** 1**,** 1**,** 1**},** // 7

**{**0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**},** // 8

**{**0**,** 0**,** 0**,** 0**,** 1**,** 0**,** 0**}** // 9

**};**

// variables

int currentNumber **=** 0**;**

int currentBrightness **=** maxBrightness**;**

bool isButtonsPressed **=** **false;**

bool isBrightnessIncreasing **=** **true;**

void setup**()** **{**

**for** **(**int i **=** 0**;** i **<** pinCount**;** i**++)** **{**

pinMode**(**pins**[**i**],** OUTPUT**);**

**}**

pinMode**(**increaseButtonPin**,** INPUT\_PULLUP**);**

pinMode**(**decreaseButtonPin**,** INPUT\_PULLUP**);**

pinMode**(**brightnessButtonPin**,** INPUT\_PULLUP**);**

pinMode**(**brightnessPin**,** OUTPUT**);**

**}**

void loop**()** **{**

handleNumberChange**();**

handleBrightnessChange**();**

displayNumber**();**

delay**(**100**);**

**}**

void handleNumberChange**()** **{**

**if** **(**isButtonPressed**(**increaseButtonPin**)** **&&** **!**isButtonsPressed**)** **{**

currentNumber **=** constrain**(**currentNumber **+** 1**,** minNumber**,** maxNumber**);**

isButtonsPressed **=** **true;**

**}** **else** **if** **(**isButtonPressed**(**decreaseButtonPin**)** **&&** **!**isButtonsPressed**)** **{**

currentNumber **=** constrain**(**currentNumber **-** 1**,** minNumber**,** maxNumber**);**

isButtonsPressed **=** **true;**

**}** **else** **if** **(**areButtonsReleased**())** **{**

isButtonsPressed **=** **false;**

**}**

**}**

void handleBrightnessChange**()** **{**

**if** **(**isButtonPressed**(**brightnessButtonPin**)** **&&** **!**isButtonsPressed**)** **{**

currentBrightness **=** isBrightnessIncreasing **?** currentBrightness **+** brightnessChangeStep **:** currentBrightness **-** brightnessChangeStep**;**

currentBrightness **=** constrain**(**currentBrightness**,** minBrightness**,** maxBrightness**);**

**if** **(**currentBrightness **==** maxBrightness**)** isBrightnessIncreasing **=** **false;**

**else** **if** **(**currentBrightness **==** minBrightness**)** isBrightnessIncreasing **=** **true;**

isButtonsPressed **=** **true;**

**}**

analogWrite**(**brightnessPin**,** currentBrightness**);**

**}**

void displayNumber**()** **{**

**for** **(**int i **=** 0**;** i **<** pinCount**;** i**++)** **{**

digitalWrite**(**pins**[**i**],** numbers**[**currentNumber**][**i**]);**

**}**

**}**

bool isButtonPressed**(**int pin**)** **{**

**return** **!**digitalRead**(**pin**);**

**}**

bool areButtonsReleased**()** **{**

**return** digitalRead**(**increaseButtonPin**)** **&&** digitalRead**(**decreaseButtonPin**)** **&&** digitalRead**(**brightnessButtonPin**);**

**}**

**Description of the circuit operation, including a detailed explanation of each line of code and the behavior of the circuit during operation:**

This Arduino-controlled circuit drives a 7-segment display to show numbers (0–9) and adjusts its brightness via button presses. Three buttons are used: two for increasing and decreasing the displayed number and one for controlling brightness. The brightness of the 7-segment display is modulated using PWM (Pulse Width Modulation) on a separate pin.

***Detailed Explanation of Each Line of Code and Circuit Behavior:***

1. Global Constants and Variables:

const int pinCount **=** 7**;**

const int numberCount **=** 10**;**

const int maxNumber **=** 9**;**

const int minNumber **=** 0**;**

const int maxBrightness **=** 255**;**

const int minBrightness **=** 0**;**

const int brightnessChangeStep **=** 50**;**

* 1. ***pinCount and numberCount***: These define the number of segments in the display and the number of numbers to display (0–9).
  2. ***maxNumber, minNumber***: Bounds for the number displayed
  3. ***maxBrightness, minBrightness, and brightnessChangeStep***: Define the range and step size for adjusting brightness. Brightness values are between 0 (off) and 255 (full brightness).

const int pins**[**pinCount**]** **=** **{**7**,** 6**,** 4**,** 3**,** 2**,** 8**,** 9**};**

const int increaseButtonPin **=** 10**;**

const int decreaseButtonPin **=** 11**;**

const int brightnessButtonPin **=** 12**;**

const int brightnessPin **=** 5**;**

* 1. ***Button Pins***: Define the input pins for the three buttons.
  2. ***brightnessPin***: Controls the brightness of the 7-segment display via PWM.
  3. ***pins array***: Represents the Arduino pins connected to the 7-segment display segments (A, B, C, D, E, F, G).

const int numbers**[**numberCount**][**pinCount**]** **=** **{**

**{**0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 1**},** // 0

**{**1**,** 0**,** 0**,** 1**,** 1**,** 1**,** 1**},** // 1

**{**0**,** 0**,** 1**,** 0**,** 0**,** 1**,** 0**},** // 2

**{**0**,** 0**,** 0**,** 0**,** 1**,** 1**,** 0**},** // 3

**{**1**,** 0**,** 0**,** 1**,** 1**,** 0**,** 0**},** // 4

**{**0**,** 1**,** 0**,** 0**,** 1**,** 0**,** 0**},** // 5

**{**0**,** 1**,** 0**,** 0**,** 0**,** 0**,** 0**},** // 6

**{**0**,** 0**,** 0**,** 1**,** 1**,** 1**,** 1**},** // 7

**{**0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**},** // 8

**{**0**,** 0**,** 0**,** 0**,** 1**,** 0**,** 0**}** // 9

**};**

* 1. ***numbers array***: A 2D array where each row represents the binary pattern to light up a specific number (0–9) on the 7-segment display.

1. Variables for Operation:

int currentNumber **=** 0**;**

int currentBrightness **=** maxBrightness**;**

bool isButtonsPressed **=** **false;**

bool isBrightnessIncreasing **=** **true;**

* 1. ***currentNumber***: Stores the current number being displayed on the 7-segment display.
  2. ***currentBrightness***: Stores the current brightness level of the display.
  3. ***isButtonsPressed***: A flag to prevent multiple button press actions while the button is held down.
  4. ***isBrightnessIncreasing***: Toggles between increasing and decreasing brightness.

1. Setup Function:

void setup**()** **{**

**for** **(**int i **=** 0**;** i **<** pinCount**;** i**++)** **{**

pinMode**(**pins**[**i**],** OUTPUT**);**

**}**

pinMode**(**increaseButtonPin**,** INPUT\_PULLUP**);**

pinMode**(**decreaseButtonPin**,** INPUT\_PULLUP**);**

pinMode**(**brightnessButtonPin**,** INPUT\_PULLUP**);**

pinMode**(**brightnessPin**,** OUTPUT**);**

**}**

* 1. ***pinMode():*** Configures the pins connected to the 7-segment display as outputs, button pins as inputs with internal pull-up resistors and brightness pin as an output to control the brightness of the display via PWM.

4. Loop Function:

void loop**()** **{**

handleNumberChange**();**

handleBrightnessChange**();**

displayNumber**();**

delay**(**100**);**

**}**

* 1. ***handleNumberChange()***: Checks the button inputs for changing the displayed number.
  2. ***handleBrightnessChange():*** Adjusts the brightness based on the brightness button input.
  3. ***displayNumber()***: Displays the current number on the 7-segment display.
  4. ***delay(100)***: Adds a small delay to debounce the buttons and stabilize the circuit.

5. Handle Number Change:

void handleNumberChange**()** **{**

**if** **(**isButtonPressed**(**increaseButtonPin**)** **&&** **!**isButtonsPressed**)** **{**

currentNumber **=** constrain**(**currentNumber **+** 1**,** minNumber**,** maxNumber**);**

isButtonsPressed **=** **true;**

**}** **else** **if** **(**isButtonPressed**(**decreaseButtonPin**)** **&&** **!**isButtonsPressed**)** **{**

currentNumber **=** constrain**(**currentNumber **-** 1**,** minNumber**,** maxNumber**);**

isButtonsPressed **=** **true;**

**}** **else** **if** **(**areButtonsReleased**())** **{**

isButtonsPressed **=** **false;**

**}**

**}**

* 1. ***constrain()***: Ensures the current number stays within the defined limits (0–9).
  2. ***isButtonPressed()***: Helper function to check if a button is pressed (returns LOW).
  3. ***areButtonsReleased():*** Checks if all buttons are released (returns HIGH for all).

Logic: If the increase or decrease button is pressed, the current number is incremented or decremented, and a flag (isButtonsPressed) prevents repeated changes while the button is held.

6. Handle Brightness Change:

void handleBrightnessChange**()** **{**

**if** **(**isButtonPressed**(**brightnessButtonPin**)** **&&** **!**isButtonsPressed**)** **{**

currentBrightness **=** isBrightnessIncreasing **?** currentBrightness **+** brightnessChangeStep **:** currentBrightness **-** brightnessChangeStep**;**

currentBrightness **=** constrain**(**currentBrightness**,** minBrightness**,** maxBrightness**);**

**if** **(**currentBrightness **==** maxBrightness**)** isBrightnessIncreasing **=** **false;**

**else** **if** **(**currentBrightness **==** minBrightness**)** isBrightnessIncreasing **=** **true;**

isButtonsPressed **=** **true;**

**}**

analogWrite**(**brightnessPin**,** currentBrightness**);**

**}**

* 1. ***Brightness Control***: Adjusts brightness either up or down depending on the flag isBrightnessIncreasing.
  2. ***analogWrite():*** Writes the PWM signal to control the brightness on brightnessPin.
  3. ***Bounds Checking***: Ensures brightness stays within the minBrightness and maxBrightness range.
  4. ***Toggling***: Switches between increasing and decreasing brightness when limits are hit.

7. Display Number on 7-Segment Display:

void displayNumber**()** **{**

**for** **(**int i **=** 0**;** i **<** pinCount**;** i**++)** **{**

digitalWrite**(**pins**[**i**],** numbers**[**currentNumber**][**i**]);**

**}**

**}**

* 1. ***digitalWrite():*** Activates the appropriate segments on the 7-segment display to show the current number.

8. Button State Helper Functions:

bool isButtonPressed**(**int pin**)** **{**

**return** **!**digitalRead**(**pin**);**

**}**

bool areButtonsReleased**()** **{**

**return** digitalRead**(**increaseButtonPin**)** **&&** digitalRead**(**decreaseButtonPin**)** **&&** digitalRead**(**brightnessButtonPin**);**

**}**

* 1. ***isButtonPressed():*** Checks if a button is pressed (reads LOW).
  2. ***areButtonsReleased():*** Verifies if all buttons are released (reads HIGH).

Circuit Behavior:

1. **Displaying Numbers**:
   1. When the circuit is powered on, the 7-segment display shows the currentNumber (starting from 0).
   2. the increase or decrease button increments or decrements the displayed number, with limits at 0 and 9.
2. **Brightness Control**:
   1. A third button adjusts the brightness of the display. Each press toggles the brightness up or down by brightnessChangeStep.
   2. Once the brightness reaches its upper or lower limit (255 or 0), it changes direction (increase or decrease).
3. **Debouncing**:
   1. The system uses the isButtonsPressed flag and a small delay to prevent multiple actions while the button is held down.