

Atividade 02 - Jogo de Memória "Genius" Simplificado	Versão 2.3	TECPUC Cursos Técnicos Ensino Médio GRUPO MARISTA
Relatório de Atividade	Data: 15/09/2023	

1. Atividade 02 - Jogo de Memória "Genius" Simplificado

1.1 Informações Gerais

TURMA		GRUPO	
-------	--	-------	--

Aluno	#### NOME COMPLETO ####	Funções	Pontuação da Equipe (1 a 3)
Aluno 1	Larissa Adames Gonçalves	Programadora	3
Aluno 2	Yasmin Kronemberger Aguilera Oliveira	Montadora / Tkcd	3
Aluno 3	Diego Sitorski	Montador	2
Aluno 4	Guilherme Falcão Bahi	Pesquisador	1

Pontuação:

1 – Pouca ou nenhuma participação (Redução de 50% da nota na atividade)

2 – Participação mediana (Redução de 25% da nota na atividade)

3 – Participação Adequada (Sem redução nota)

FALTOU

1.2 Evolução da Atividade


Atividade	Nome reduzido do exercício	Concluído (%)	OBS (Opcional)
Exercício 1	Fazer o jogo Genius no Arduino	100%	

1.3 Links

Descrição	Link
Vídeo (Quando solicitado)	
Montagem Thinkercad	https://www.tinkercad.com/things/3Ybjrk2Z2hx-smooth-borwo/editel?sharecode=x_mkEb7CeRLgwrI0cHxG1eaRwyFpQWOQu

1.4 Dificuldades encontradas

Houve uma dificuldade com o led verde, ele ficava aceso ininterruptamente, e apenas desligava quando o botão era pressionado.

Atividade 02 - Jogo de Memória "Genius" Simplificado	Versão 2.3	 TECPUC Cursos Técnicos Ensino Médio GRUPO MARISTA
Relatório de Atividade	Data: 15/09/2023	

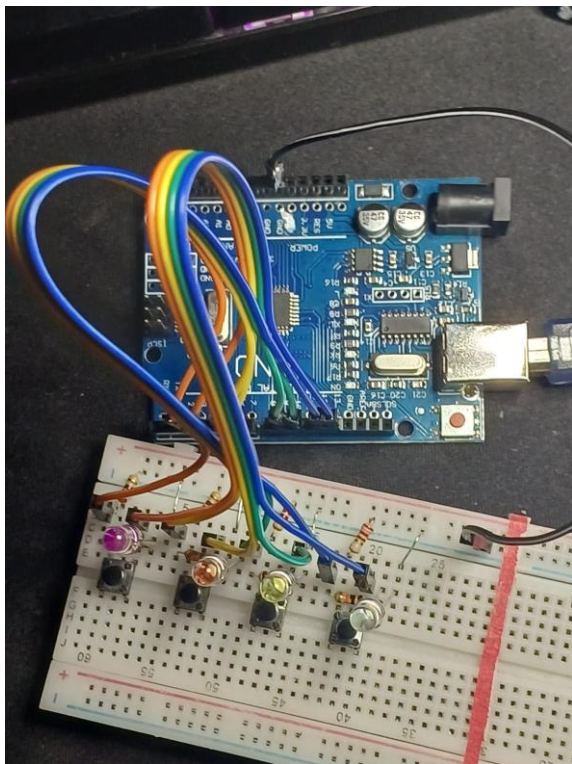
1.5 Lista de Componentes

Quant	Descrição
4	LEDS
1	Buzzer
Diversos	Jumpers
Diversos	Resistores de 10kΩ e 220Ω
1	Protoboard
1	Arduino UNO R3
1	Cabo USB type-A - type-B
4	Push Button

1.6 Resolução das Atividades

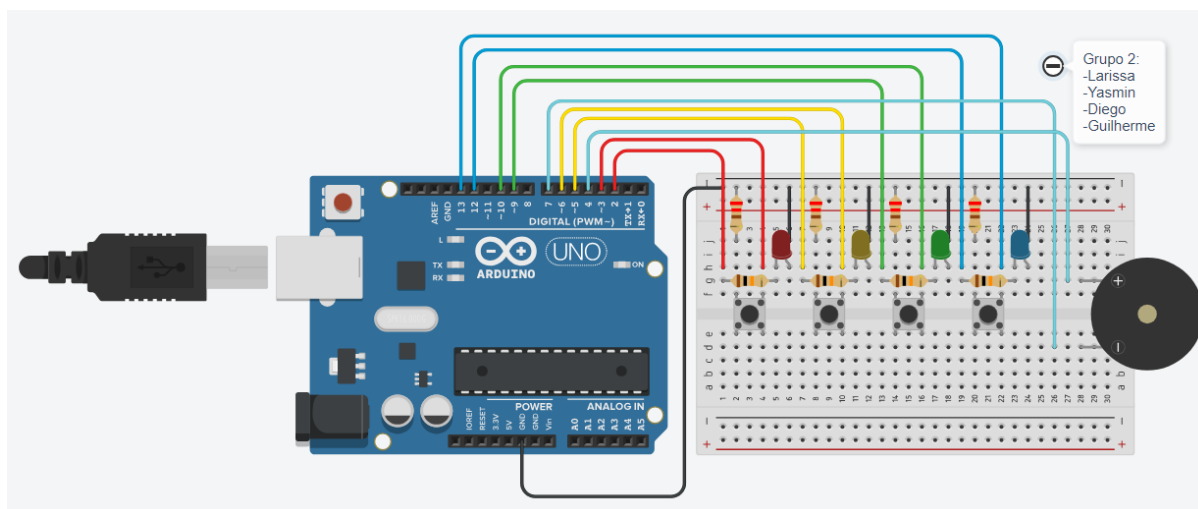
1.6.1 Atividade 1

- Fotos Montagem 1



- Esquema de Ligação 1 (Imagem do Projeto no Tinkercad)

PS: No Tinkercad não foi possível virar os leds, porém na montagem é o positivo que está no resistor, favor considerar que os leds estão invertidos no Tinkercad



Atividade 02 - Jogo de Memória "Genius" Simplificado	Versão 2.3	TECPUC Cursos Técnicos Ensino Médio GRUPO MARISTA
Relatório de Atividade	Data: 15/09/2023	

- Código Montagem 1

```
#define NOTE_B0 31
#define NOTE_C1 33
#define NOTE_CS1 35
#define NOTE_D1 37
#define NOTE_DS1 39
#define NOTE_E1 41
#define NOTE_F1 44
#define NOTE_FS1 46
#define NOTE_G1 49
#define NOTE_GS1 52
#define NOTE_A1 55
#define NOTE_AS1 58
#define NOTE_B1 62
#define NOTE_C2 65
#define NOTE_CS2 69
#define NOTE_D2 73
#define NOTE_DS2 78
#define NOTE_E2 82
#define NOTE_F2 87
#define NOTE_FS2 93
#define NOTE_G2 98
#define NOTE_GS2 104
#define NOTE_A2 110
#define NOTE_AS2 117
#define NOTE_B2 123
#define NOTE_C3 131
#define NOTE_CS3 139
#define NOTE_D3 147
#define NOTE_DS3 156
#define NOTE_E3 165
#define NOTE_F3 175
#define NOTE_FS3 185
#define NOTE_G3 196
#define NOTE_GS3 208
#define NOTE_A3 220
#define NOTE_AS3 233
#define NOTE_B3 247
```

Atividade 02 - Jogo de Memória "Genius" Simplificado	Versão 2.3	TECPUC Cursos Técnicos Ensino Médio GRUPO MARISTA
Relatório de Atividade	Data: 15/09/2023	

```
#define NOTE_C4 262
#define NOTE_CS4 277
#define NOTE_D4 294
#define NOTE_DS4 311
#define NOTE_E4 330
#define NOTE_F4 349
#define NOTE_FS4 370
#define NOTE_G4 392
#define NOTE_GS4 415
#define NOTE_A4 440
#define NOTE_AS4 466
#define NOTE_B4 494
#define NOTE_C5 523
#define NOTE_CS5 554
#define NOTE_D5 587
#define NOTE_DS5 622
#define NOTE_E5 659
#define NOTE_F5 698
#define NOTE_FS5 740
#define NOTE_G5 784
#define NOTE_GS5 831
#define NOTE_A5 880
#define NOTE_AS5 932
#define NOTE_B5 988
#define NOTE_C6 1047
#define NOTE_CS6 1109
#define NOTE_D6 1175
#define NOTE_DS6 1245
#define NOTE_E6 1319
#define NOTE_F6 1397
#define NOTE_FS6 1480
#define NOTE_G6 1568
#define NOTE_GS6 1661
#define NOTE_A6 1760
#define NOTE_AS6 1865
#define NOTE_B6 1976
#define NOTE_C7 2093
#define NOTE_CS7 2217
```

Atividade 02 - Jogo de Memória "Genius" Simplificado	Versão 2.3	TECPUC Cursos Técnicos Ensino Médio
Relatório de Atividade	Data: 15/09/2023	GRUPO MARISTA

```

#define NOTE_D7 2349
#define NOTE_DS7 2489
#define NOTE_E7 2637
#define NOTE_F7 2794
#define NOTE_FS7 2960
#define NOTE_G7 3136
#define NOTE_GS7 3322
#define NOTE_A7 3520
#define NOTE_AS7 3729
#define NOTE_B7 3951
#define NOTE_C8 4186
#define NOTE_CS8 4435
#define NOTE_D8 4699
#define NOTE_DS8 4978

#define CHOICE_OFF      0
#define CHOICE_NONE     0
#define CHOICE_RED      (1 << 0)
#define CHOICE_GREEN    (1 << 1)
#define CHOICE_BLUE     (1 << 2)
#define CHOICE_YELLOW   (1 << 3)

#define LED_RED         2
#define LED_GREEN       10
#define LED_BLUE        12
#define LED_YELLOW      5

#define BUTTON_RED      3
#define BUTTON_GREEN    9
#define BUTTON_BLUE     13
#define BUTTON_YELLOW   6

#define BUZZER1         4
#define BUZZER2         7

#define ROUNDS_TO_WIN   13
#define ENTRY_TIME_LIMIT 3000
#define MODE_MEMORY     0

```

```
#define MODE_BATTLE 1
#define MODE_BEEGEES 2
byte gameMode = MODE_MEMORY;
byte gameBoard[32];
byte gameRound = 0;
void setup()
{
    pinMode(BUTTON_RED, OUTPUT);
    pinMode(BUTTON_GREEN, OUTPUT);
    pinMode(BUTTON_BLUE, OUTPUT);
    pinMode(BUTTON_YELLOW, OUTPUT);
    pinMode(LED_RED, OUTPUT);
    pinMode(LED_GREEN, OUTPUT);
    pinMode(LED_BLUE, OUTPUT);
    pinMode(LED_YELLOW, OUTPUT);
    pinMode(BUZZER1, OUTPUT);
    pinMode(BUZZER2, OUTPUT);
    gameMode = MODE_MEMORY;
    if (checkButton() == CHOICE_YELLOW) play_beegees();
    if (checkButton() == CHOICE_GREEN)
    {
        gameMode = MODE_BATTLE;
        setLEDs(CHOICE_GREEN);
        toner(CHOICE_GREEN, 150);
        setLEDs(CHOICE_RED | CHOICE_BLUE | CHOICE_YELLOW);
        while (checkButton() != CHOICE_NONE) ;
    }
    play_winner(); // Após a conclusão da configuração, diga olá ao mundo
}
void loop()
{
    attractMode();
    setLEDs(CHOICE_RED | CHOICE_GREEN | CHOICE_BLUE | CHOICE_YELLOW); // Ativar
    todos os LEDs
    delay(1000);
    setLEDs(CHOICE_OFF); // Desligue os LEDs
    delay(250);
    if (gameMode == MODE_MEMORY)
    {
```

Atividade 02 - Jogo de Memória "Genius" Simplificado	Versão 2.3	TECPUC Cursos Técnicos Ensino Médio
Relatório de Atividade	Data: 15/09/2023	GRUPO MARISTA

```

        if (play_memory() == true)
            play_winner(); // GANHOU, toca som vitória
        else
            play_loser(); // Perdeu, toca som derrota
    }
    if (gameMode == MODE_BATTLE)
    {
        play_battle();
        play_loser();
    }
}

boolean play_memory(void)
{
    randomSeed(millis()); // Gerador aleatório
    gameRound = 0; // Redefinir o jogo para o começo
    while (gameRound < ROUNDS_TO_WIN)
    {
        add_to_moves(); // Adicione um botão aos movimentos atuais e reproduza-os
        playMoves(); // Jogue de volta o tabuleiro do jogo atual
        for (byte currentMove = 0 ; currentMove < gameRound ; currentMove++)
        {
            byte choice = wait_for_button(); // Veja o botão que o usuário
            pressiona
            if (choice == 0) return false; // Se a espera expirar, o jogador perde
            if (choice != gameBoard[currentMove]) return false;
        }
        delay(1000);
    }
    return true;
}

boolean play_battle(void)
{
    gameRound = 0;
    while (1)
    {
        byte newButton = wait_for_button();
        gameBoard[gameRound++] = newButton;
        for (byte currentMove = 0 ; currentMove < gameRound ; currentMove++)
        {

```



```
        byte choice = wait_for_button();
        if (choice == 0) return false;
        if (choice != gameBoard[currentMove]) return false;
    }
    delay(100);
}
return true;
}

void playMoves(void)
{
    for (byte currentMove = 0 ; currentMove < gameRound ; currentMove++)
    {
        toner(gameBoard[currentMove], 150);
        delay(150); // 150 funciona bem. 75 fica rápido.
    }
}

void add_to_moves(void)
{
    byte newButton = random(0, 4);
    if (newButton == 0) newButton = CHOICE_RED;
    else if (newButton == 1) newButton = CHOICE_GREEN;
    else if (newButton == 2) newButton = CHOICE_BLUE;
    else if (newButton == 3) newButton = CHOICE_YELLOW;
    gameBoard[gameRound++] = newButton;
}

void setLEDs(byte leds)
{
    if ((leds & CHOICE_RED) != 0)
        digitalWrite(LED_RED, HIGH);
    else
        digitalWrite(LED_RED, LOW);
    if ((leds & CHOICE_GREEN) != 0)
        digitalWrite(LED_GREEN, HIGH);
    else
        digitalWrite(LED_GREEN, LOW);
    if ((leds & CHOICE_BLUE) != 0)
```

```
    digitalWrite(LED_BLUE, HIGH);
else
    digitalWrite(LED_BLUE, LOW);
if ((leds & CHOICE_YELLOW) != 0)
    digitalWrite(LED_YELLOW, HIGH);
else
    digitalWrite(LED_YELLOW, LOW);
}

byte wait_for_button(void) {
    long startTime = millis();

    while ( (millis() - startTime) < ENTRY_TIME_LIMIT) // Faz um loop até que
    passe muito tempo
    {
        byte button = checkButton();
        if (button != CHOICE_NONE)
        {
            toner(button, 150);
            while (checkButton() != CHOICE_NONE) ;
            delay(10);
            return button;
        }
    }

    return CHOICE_NONE; // Se chegarmos aqui, expiramos!
}

byte checkButton(void)
{
    if (digitalRead(BUTTON_RED) == 0) return (CHOICE_RED);
    else if (digitalRead(BUTTON_GREEN) == 0) return (CHOICE_GREEN);
    else if (digitalRead(BUTTON_BLUE) == 0) return (CHOICE_BLUE);
    else if (digitalRead(BUTTON_YELLOW) == 0) return (CHOICE_YELLOW);
    return (CHOICE_NONE);
}

void toner(byte which, int buzz_length_ms)
{
    setLEDs(which);
    switch (which)
```

```
{
    case CHOICE_RED:
        buzz_sound(buzz_length_ms, 1136);
        break;
    case CHOICE_GREEN:
        buzz_sound(buzz_length_ms, 568);
        break;
    case CHOICE_BLUE:
        buzz_sound(buzz_length_ms, 851);
        break;
    case CHOICE_YELLOW:
        buzz_sound(buzz_length_ms, 638);
        break;
}

setLEDs(CHOICE_OFF); // Desligue todos os LEDs
}

void buzz_sound(int buzz_length_ms, int buzz_delay_us)
{
    long buzz_length_us = buzz_length_ms * (long)1000;
    while (buzz_length_us > (buzz_delay_us * 2))
    {
        buzz_length_us -= buzz_delay_us * 2;
        digitalWrite(BUZZER1, LOW);
        digitalWrite(BUZZER2, HIGH);
        delayMicroseconds(buzz_delay_us);
        digitalWrite(BUZZER1, HIGH);
        digitalWrite(BUZZER2, LOW);
        delayMicroseconds(buzz_delay_us);
    }
}

void play_winner(void)
{
    setLEDs(CHOICE_GREEN | CHOICE_BLUE);
    winner_sound();
    setLEDs(CHOICE_RED | CHOICE_YELLOW);
    winner_sound();
    setLEDs(CHOICE_GREEN | CHOICE_BLUE);
    winner_sound();
}
```

```
    setLEDs(CHOICE_RED | CHOICE_YELLOW);
    winner_sound();
}

void winner_sound(void)
{
    for (byte x = 250 ; x > 70 ; x--)
    {
        for (byte y = 0 ; y < 3 ; y++)
        {
            digitalWrite(BUZZER2, HIGH);
            digitalWrite(BUZZER1, LOW);
            delayMicroseconds(x);
            digitalWrite(BUZZER2, LOW);
            digitalWrite(BUZZER1, HIGH);
            delayMicroseconds(x);
        }
    }
}

void play_loser(void)
{
    setLEDs(CHOICE_RED | CHOICE_GREEN);
    buzz_sound(255, 1500);
    setLEDs(CHOICE_BLUE | CHOICE_YELLOW);
    buzz_sound(255, 1500);
    setLEDs(CHOICE_RED | CHOICE_GREEN);
    buzz_sound(255, 1500);
    setLEDs(CHOICE_BLUE | CHOICE_YELLOW);
    buzz_sound(255, 1500);
}


void attractMode(void)
{
    while (1)
    {
        setLEDs(CHOICE_RED);
        delay(100);
        if (checkButton() != CHOICE_NONE) return;
        setLEDs(CHOICE_BLUE);
        delay(100);
    }
}
```

```
    if (checkButton() != CHOICE_NONE) return;
    setLEDs(CHOICE_GREEN);
    delay(100);
    if (checkButton() != CHOICE_NONE) return;
    setLEDs(CHOICE_YELLOW);
    delay(100);
    if (checkButton() != CHOICE_NONE) return;
}
}

int melody[] = {
    NOTE_G4, NOTE_A4, 0, NOTE_C5, 0, 0, NOTE_G4, 0, 0, 0,
    NOTE_E4, 0, NOTE_D4, NOTE_E4, NOTE_G4, 0,
    NOTE_D4, NOTE_E4, 0, NOTE_G4, 0, 0,
    NOTE_D4, 0, NOTE_E4, 0, NOTE_G4, 0, NOTE_A4, 0, NOTE_C5, 0
};

int noteDuration = 115;
int LEDnumber = 0;
void play_beegees()
{
    setLEDs(CHOICE_YELLOW);
    toner(CHOICE_YELLOW, 150);
    setLEDs(CHOICE_RED | CHOICE_GREEN | CHOICE_BLUE);
    while (checkButton() != CHOICE_NONE) ;
    setLEDs(CHOICE_NONE);
    delay(1000);
    digitalWrite(BUZZER1, LOW);
    while (checkButton() == CHOICE_NONE) // Reproduzir música até você
    pressionar um botão
    {
        for (int thisNote = 0; thisNote < 32; thisNote++) {
            changeLED();
            tone(BUZZER2, melody[thisNote], noteDuration);
            int pauseBetweenNotes = noteDuration * 1.30;
            delay(pauseBetweenNotes);
            noTone(BUZZER2);
        }
    }
}

void changeLED(void)
```

Atividade 02 - Jogo de Memória "Genius" Simplificado	Versão 2.3	 TECPUC Cursos Técnicos Ensino Médio GRUPO MARISTA
Relatório de Atividade	Data: 15/09/2023	

```

{
    setLEDs(1 << LEDnumber); // Mude o LED
    LEDnumber++; // Ir para o próximo LED
    if (LEDnumber > 3) LEDnumber = 0; // Enrole o balcão, se necessário
}

```

Atividade 02 - Jogo de Memória "Genius" Simplificado	Versão 2.3	TECPUC Cursos Técnicos Ensino Médio
Relatório de Atividade	Data: 15/09/2023	GRUPO MARISTA

1.7 Referências

Arduino - Home. Disponível em: <<https://www.arduino.cc>>.

Como usar com Arduino - Chave Tátil / Push Button - BLOG MASTERWALKER SHOP. Disponível em: <<https://blogmasterwalkershop.com.br/arduino/arduino-utilizando-o-push-button>>.

Como fazer jumpers caseiros melhores do que os dos kits chineses. Disponível em: <<https://br-arduino.org/2015/04/como-fazer-jumpers-caseiros-melhores-do-que-os-dos-kits-chineses.html>>. Acesso em: 16 set. 2023.

ELETROGATE. Genius! - Jogo da Memória no Arduino. Disponível em: <<https://blog.eletrogate.com/genius-no-arduino-bora-jogar/>>. Acesso em: 16 set. 2023.

Jogo da Memória (Genius) - arduino jogo #03 - Squids Arduino. Disponível em: <<https://www.squids.com.br/arduino/index.php/projetos-arduino/jogos/252-jogo-da-memoria-genius-arduino-jogo-03>>. Acesso em: 16 set. 2023.

MAKERHERO, E. Aprenda a piscar um LED com Arduino. Disponível em: <<https://www.makerhero.com/blog/aprenda-a-piscar-um-led-com-arduino/>>.

Push Button Arduino: 3 Modos de ligar botões no Arduino. Disponível em: <<https://guiarobotica.com/push-button-arduino/>>.

Tinkercad Aula 12 - Servomotores. Disponível em: <<https://www.youtube.com/watch?v=R-lkljAtAME>>.

TINKERCAD. Tinkercad | From mind to design in minutes. Disponível em: <<https://www.tinkercad.com/dashboard>>.