

FIAP GRADUAÇÃO

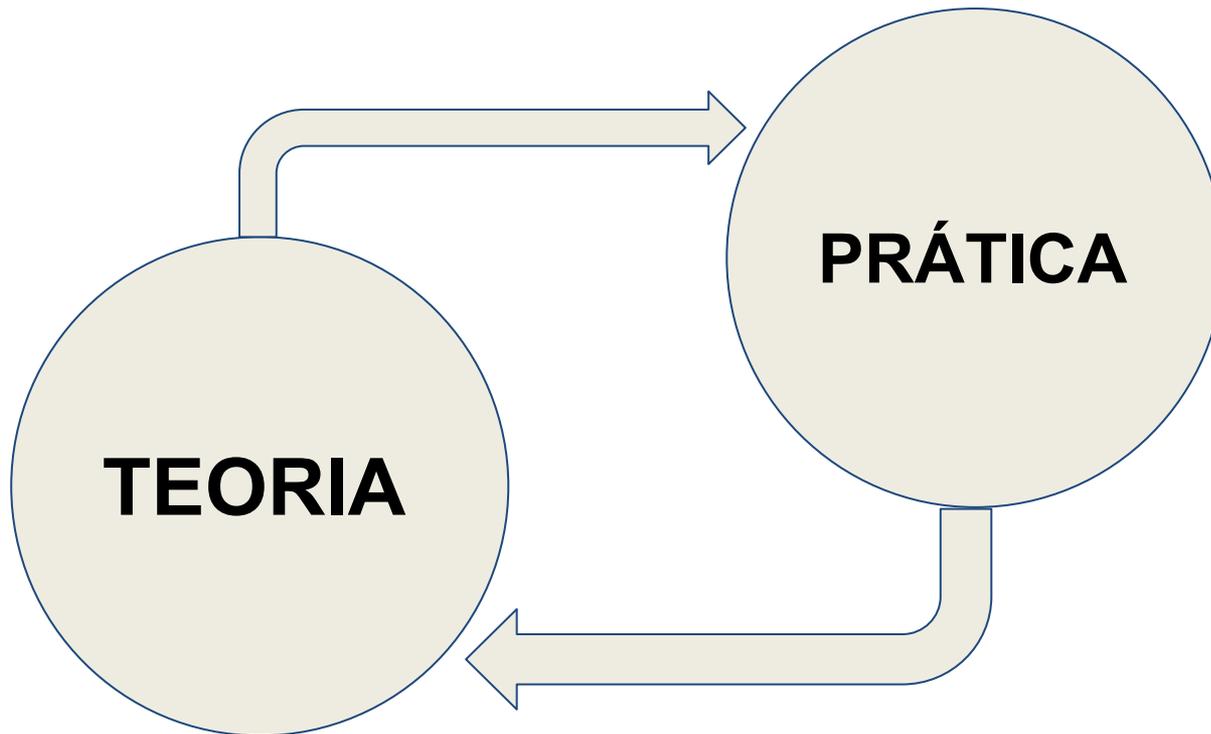
# TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

Disruptive Architectures: AI and IoT

**PROF. Arnaldo Viana**

## ■ O Que Esperar do Curso

- **Dinâmica das aulas:**
  - As aulas terão conteúdos teóricos e práticos.



## ■ Instalação da infra

- 1ª Parte: Instalação do Arduino IDE (win/linux).
  - <https://www.arduino.cc/en/software>

Interface de programação

- 2ª. Parte: Instalação SimuIDE(win/linux)
  - <https://www.simulide.com/p/downloads.html>
- 3ª. Parte: Instalação emulador de serial port
  - Com0Com (Win)
    - <https://sourceforge.net/projects/com0com/>
  - tty0tty (Linux)
    - <https://github.com/freemed/tty0tty>

Simulador quando não tenho um arduino

# Comunicação serial - Mandar informação

Mandar informação pela serial ja sabemos, apenas relembrando:

The image shows an IDE window with a code editor and a serial monitor. The code in the editor is as follows:

```

1 void setup() {
2 // inicializa a comunicação serial:
3 Serial.begin(9600);
4 }
5
6 void loop() {
7 // envia string e da quebra de linha
8   Serial.println("Ola mundo");
9   delay(500);
10 }
11

```

The serial monitor window shows the output of the program, which is "Ola mundo" repeated several times. A red circle highlights the text in the serial monitor, and a red callout box points to it with the text "Dados recebidos da serial".

Below the serial monitor is a diagram of an Arduino Nano-8 board with its pin headers labeled:

1 Tx	Vin
0 Rx	Gnd
Rst	Rst
Gnd	5V
2	A7
3 ~	A6
4	A5
5 ~	A4
6 ~	A3
7	A2
8	A1
9 ~	A0
10 ~	Arf
11 ~	3V3
12	13

At the bottom of the IDE window, a status bar shows the compilation and upload process:

```

SUCESSO!!! Compilação concluída
-----
Fazendo upload:
C:/Users/ArnaldoAVJ/AppData/Local/simulide/dimmer.hex

Firmware carregado em atmega328

```

# Comunicação serial - Receber informação

Recebendo informação pela serial:

backup.simu\*

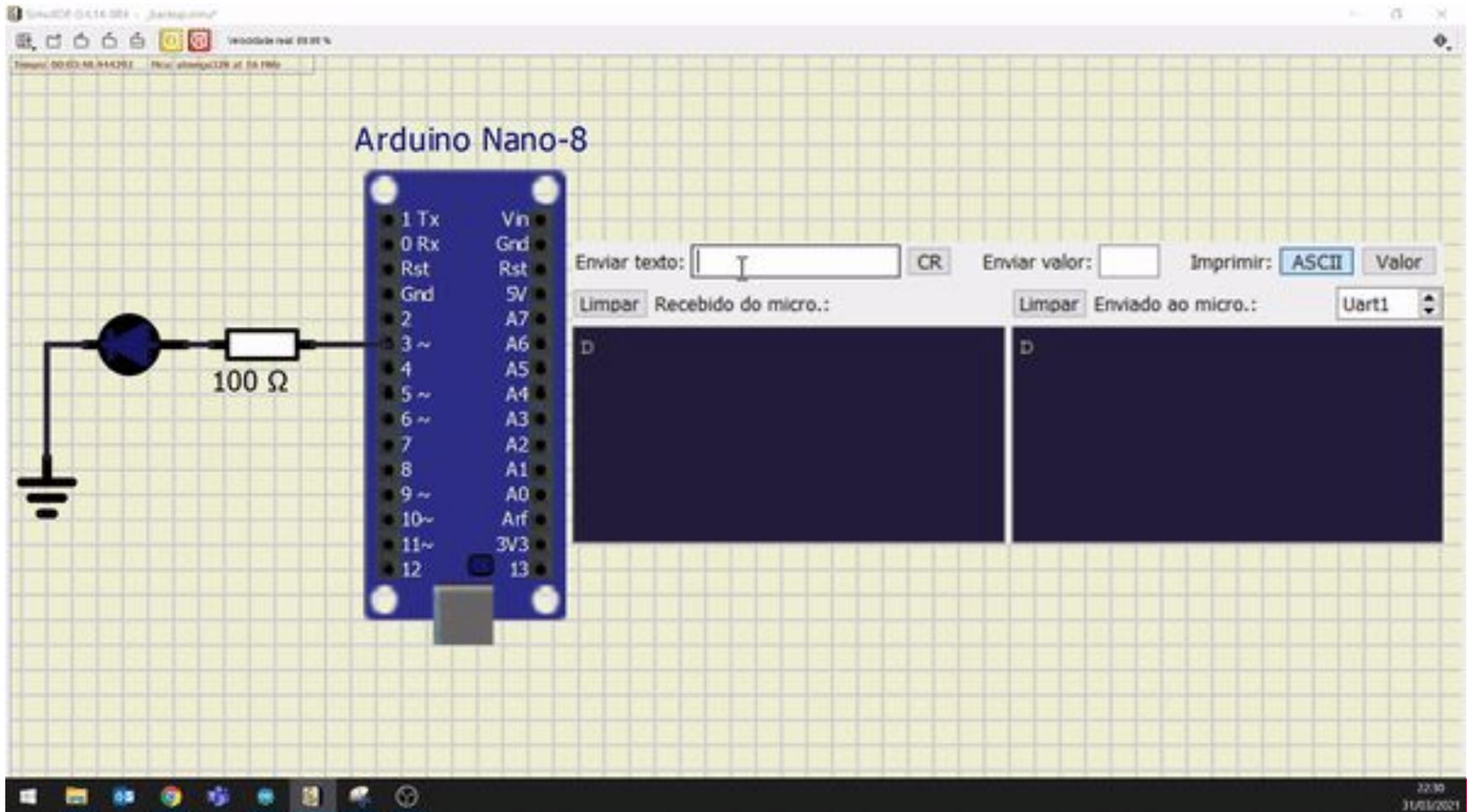
```

1 /*
2 exemplo1: loopback
3
4 recebe um caractere do monitor serial
5 e envia para o monitor serial
6 */
7
8 char dado;
9
10 void setup() {
11 // inicializa a comunicação serial:
12 Serial.begin(9600);
13 pinMode(3, OUTPUT);
14 }
15
16 void loop() {
17 // verifica se tem alguma coisa para receber
18 if(Serial.available()){
19 //salva o caractere em dado
20 dado = Serial.read();
21 //
22 Serial.print(dado);
23 }
24 }
25

```

# Acionamento via serial

A implementação é bem simples! Antes de ver o código, pense nas possibilidades....



# Acionamento via serial

```
const int ledPin = 11;           // nomeia o pin011 do arduino como ledPin

void setup() {
  pinMode(ledPin, OUTPUT);      // Pin011 como saída
  Serial.begin(9600);          // Inicializa a comunicação serial
}

void loop() {
  char dado;                    // Declara dado como variável local do tipo char
  if(Serial.available() > 0){   // Se tem alguma coisa no buffer da serial
    dado = Serial.read();       // Então salva em dado
    Serial.print(dado);         // Exibe o valor de dado
    if (dado == 'L'){           // Se dado igual a 'L'
      digitalWrite(ledPin, HIGH); // Então acenda o led
      delay(500);               // delay de 0,5 seg.
    }                            // end if dado == 'L'
    else if (dado == 'D'){      // Se dado igual a 'D'
      digitalWrite (ledPin, LOW); // Então apaga o led
      delay(500);               // delay de 0,5 seg.
    }                            // end if dado == 'D'
  }                              // end if Serial.available
}                                  // end loop
```

# Projeto 1

Vamos criar um jogo da memória :)

**INSTRUÇÕES:**  
DIGITE O NUMERO QUE CORRESPONDE A COR

**NUMERO - COR**

- 1 - LED VERMELHO
- 2 - LED VERDE
- 3 - LED AZUL

```

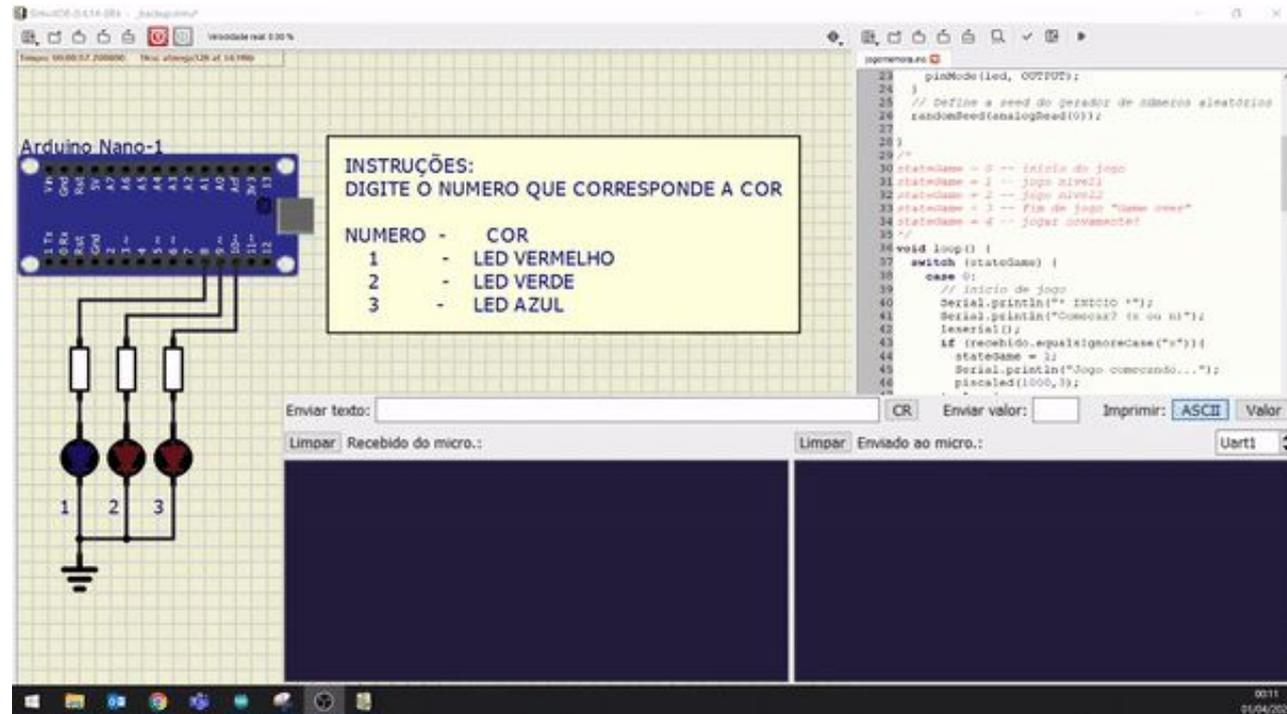
23 pinMode(led, OUTPUT);
24 }
25 // Define a seed do gerador de números aleatórios
26 randomSeed(analogRead(0));
27
28 }
29 //
30 stateGame = 0 -- início do jogo
31 stateGame = 1 -- jogo nível1
32 stateGame = 2 -- jogo nível2
33 stateGame = 3 -- fim de jogo "Game over"
34 stateGame = 4 -- jogar novamente!
35 //
36 void loop() {
37   switch (stateGame) {
38     case 0:
39       // início de jogo
40       Serial.println("** início **");
41       Serial.println("Comocar? (s ou n)");
42       Serial.println();
43       if (recebido.equalsIgnoreCase("s")){
44         stateGame = 1;
45         Serial.println("Jogo começando...");
46         piscasled(1000,3);
47       }
48     }
49   }
50 }

```

Enviar texto:  CR Enviar valor:  Imprimir: ASCII Valor

Limpar Recebido do micro.:  Limpar Enviado ao micro.:  Uart1

# Projeto 1



Note que o arduino do projeto 1 é um pouco diferente do modelo UNO que usamos em sala de aula, o nome dele é Arduino NANO, relaxa meu jovem pois trata-se basicamente de uma versão reduzida do Arduino UNO, o código é o igual.

# Desafios - Projeto1

Monte o circuito do Projeto1 na protoboard ou no simulador (simulide ou thinkercad):

1. Carregue o código fornecido de base e **JOGUE O JOGO**;
2. Abra o código fornecido e análise como ele foi elaborado:
  - Quais são as estruturas utilizadas? Quais eu ainda não conheço?
  - O jogo tem começo, meio e fim?
3. Altere o código base adicionando novas funcionalidades ao jogo.

Sugestões:

- Implemente o nível de dificuldade 3,4,5 e 6....;
- A dificuldade pode ser incrementada a cada rodada alterando os parâmetros da função geraSequencia(), identifique quais são esses parâmetros e altere no seu programa;
- Adicione mais LEDs (no lugar de 3 leds usar 6 leds) no circuito e implemente no código as alterações necessárias para funcionar;
- Altere a interface do jogo para tornar mais atrativo para o usuário;
- Proponha e implemente uma estrutura de programação diferente da utilizada e demonstre o funcionamento.

**Implemente 3 (ou mais) das sugestões acima, ou proponha e implemente uma nova funcionalidade ou melhoria no código.**