FATEC

Desenvolvimento de Software Multiplataforma

2º SEMESTRE 2024

IAL-011 - Internet das Coisas e Aplicações

Prof. Me. Eng. Santana

C++ e Ambiente de Desenvolvimento

Fatec
Faculdade de Tecnologia

desenvolvimento
de software multiplataforma

C++ / Wiring

- A linguagem "Wiring" é uma linguagem de programação e uma estrutura de software utilizada no Arduino e em outras plataformas de prototipagem eletrônica baseadas no microcontrolador Atmel AVR. Ela foi desenvolvida por Hernando Barragán como parte de sua tese de mestrado na Interaction Design Institute Ivrea, na Itália, e serve como a base para a linguagem de programação utilizada no Arduino.

- A linguagem Wiring é baseada na linguagem de programação C/C++ e foi projetada para ser acessível e fácil de aprender, especialmente para iniciantes em eletrônica e programação. Ela simplifica a interação com os pinos de entrada e saída do microcontrolador, fornecendo uma série de funções e bibliotecas que permitem aos desenvolvedores escreverem código para controlar dispositivos eletrônicos e interagir com o ambiente físico de forma intuitiva.

Declaração de variaveis

int ledPin = 13; // Declara uma variável do tipo inteiro

float temperature = 24.5; // Declara uma variável do tipo float

char myChar = 'A'; // Declara uma variável do tipo char

const int ledpin=13;// constante que não poderá ser alterada

Vetores e Matrizes

- Vetores (Arrays Unidimensionais): Um vetor é uma coleção ordenada de elementos do mesmo tipo, acessados por um índice inteiro.

```
int vetor[5];
int vetor[5] = {10, 20, 30, 40, 50};
```

Matrizes (Arrays Multidimensionais):

Uma matriz é uma coleção multidimensional de element organizados por combinações de índices.

```
int matriz[3][3];
int matriz[3][3] = { {1, 2, 3}, {4, 5, 6}, {7, 8, 9} };
```

Declaração de variaveis

Tipo	Especificação
boolean	Dados do tipo booleano podem possuir apenas o valor Verdadeiro (TRUE) ou Falso (FALSE).
byte	Um dado do tipo byte armazena um número de 8 bits sem sinal que deve possuir um valor entre 0 e 255.
char	O tipo caractere utiliza 1 byte de memória e armazena o valor de um caractere. A representação simbólica do caractere deve ser escrita entre aspas simples (' ').
int	O tipo de dados inteiro é referente aos valores conjuntos dos numéricos inteiros naturais positivos e negativos, incluindo o zero e abrangendo a faixa de -32.768 a 32.767. Necessita de 2 bytes da memória para armazenamento.
float	Tipo de dado que representa o conjunto de números reais, positivos e negativos. Chamados de números de ponto flutuante, abrangem a faixa de 3,4028235E+38 a -3,4028235E+38. São necessários 4 bytes da memória para armazenar um valor desse tipo de dados.
String	Strings representam um conjunto ou cadeia de caracteres, como quando formamos uma palavra ou frase. Seu armazenamento é variável, dependendo da quantidade de caracteres que formam a cadeia. Um valor String deve ser delimitado por aspas duplas (" ").

Operadores Aritmeticos

Operadores Aritméticos	
Símbolo	Operação
+	Adição
-	Subtração
*	Multiplicação
/	Divisão
=	Atribuição
++	Incremento; dessa forma, a++ significa o mesmo que a = a + 1
--	Decremento; dessa forma, a-- representa o mesmo que a = a - 1
+=	Operação composta de adição; assim, a += b significa o mesmo que a = a + b
-=	Operação composta de subtração; assim, a -= b tem o mesmo significado que a = a - b
*=	Operação composta de multiplicação; assim, a *= b representa o mesmo que a = a * b
/=	Operação composta de divisão; assim, a /= b tem o mesmo efeito que a = a / b

Operadores Relacionais e Lógicos

Operadores Relacionais	
Símbolo	Operação
==	Igual
!=	Diferente
>	Maior
<	Menor
>=	Maior ou igual
<=	Menor ou igual

Operadores Lógicos	
Símbolo	Operação
&&	E (AND)
||	OU (OR)
!	NÃO (NOT)

Operadores Bit a Bit

Operadores Bit a Bit	
Símbolo	Operação
&	E (AND)
|	OU (OR)
~	NÃO (NOT)
^	OU Exclusivo (XOR)
<<	Deslocamento de bit à esquerda
>>	Deslocamento de bit à direita

```
int a = 0b0001; // 1 em binário
result = a << 2; // Resulta em 0b0100 (4 em decimal)
```

Estruturas de Controle

```
if (sensorValue > 500) {
    // Código a ser executado se a condição for verdadeira
    digitalWrite(ledPin, HIGH);
} else {
    // Código a ser executado se a condição for falsa
    digitalWrite(ledPin, LOW);
}
```

Estrutura de Seleção

```
switch (opcao) {
    case 1:
                //executar instruções
    break;
    case 2:
                //executar instruções
    break;
    default: //executar instruções caso não
                encontre um case com a opção
                        informada
    break;
}
```

Estruturas de Repetição

```
for (int i = 0; i < 10; i++) {
        // Executa este bloco 10 vezes
        digitalWrite(ledPin, HIGH);
        delay(100);
        digitalWrite(ledPin, LOW);
        delay(100);
}
```

```
int contador = 1;
while (contador <= 10) {
            Serial.println (contador);
            contador = contador + 1;
}
```

Funcoes

```
int sum(int a, int b) {
  return a + b;
}

void loop() {
  int result = sum(5, 10); // Chama a função sum
  // Use o resultado da função
}
```

Bibliotecas

```
#include <Wire.h> // Inclui a biblioteca Wire para comunicação I2C

void setup() {
  Wire.begin(); // Inicializa a biblioteca Wire
}

void loop() {
  Wire.beginTransmission(8); // Inicia a transmissão para o dispositivo com endereço 8
  Wire.write("Hello"); // Envia dados
  Wire.endTransmission(); // Finaliza a transmissão
  delay(1000);
}
```

Metodos/Funcoes Especificas Arduino

```
void setup() {
  // Inicializa o pino 13 como saída
  pinMode(13, OUTPUT);
}

void loop() {
  // Acende o LED conectado ao pino 13
  digitalWrite(13, HIGH);
  delay(1000); // Espera por um segundo
  // Apaga o LED conectado ao pino 13
  digitalWrite(13, LOW);
  delay(1000); // Espera por um segundo
}
```

Funcoes - Arduino

- **Entrada e Saída Digital**
- **Função pinMode()**

```
void setup() {
  pinMode(13, OUTPUT); // Configura o pino 13 como saída
  pinMode(7, INPUT); // Configura o pino 7 como entrada
}
```

Funcoes - Arduino

- **Função digitalWrite()**

```
digitalWrite(13, HIGH); // Define o pino 13 como HIGH (5V)
digitalWrite(13, LOW);  // Define o pino 13 como LOW (0V)
```

Funcoes - Arduino

- **Função digitalRead()**

```
int buttonState = digitalRead(7);
```

Ambientes de Desenvolvimento

- https://www.tinkercad.com/
- https://wokwi.com/
- Arduino IDE
- VSCODE + Arduino Plugin
- VSCODE + WOKWI Plugin

Lab

- Criar uma conta no Tinkercad
- Criar um Circuito Novo
- Adicionar um Arduino Uno R3
- Simular

Lab

```
//
void setup()
{
 Serial.begin(9600);
}

void loop()
{
 Serial.println("Ola Mundo!");

}
```