## **Roteiro Aula 05 – Introdução à Programação Orientada a Objetos (POO)**

### **Objetivos da aula:**

Apresentar alguns dos conceitos básicos de Programação Orientada a Objetos para estruturação de projetos e organização de componentes de código com exemplos em C++.

### **1. Abertura**

* Saudações
  + Meu nome é Pedro Martins, sou estudante do 3º ano de TADS na FT Unicamp e integrante da equipe de Training da Robocamp.
  + Nesta breve aula introdutória, vou apresentar alguns conceitos básicos de Programação Orientada a Objetos, utilizando exemplos práticos em C++.
* Meu objetivo é mostrar como os conceitos de classes e objetos podem ajudar a estruturar melhor os projetos, organizar o código em componentes e facilitar o reuso, a coesão entre partes do sistema e a divisão de responsabilidades. Sei que esses termos podem parecer abstratos agora, mas ao longo da aula eles vão fazer mais sentido dentro do contexto da robótica e da automação.
* Como o tempo é curto, vou oferecer a vocês — em linguagem figurada — apenas um “cubinho de gelo da ponta do iceberg” que é a POO. Não temos a pretensão de esgotar nenhum tópico aqui, mas fiquem tranquilos: estamos totalmente abertos para dúvidas e conversas mais aprofundadas depois da aula.

### 

### **2. Paradigmas de programação**

* Um paradigma de programação é basicamente um jeito de organizar e escrever código para resolver problemas com o computador.
* Hoje em dia existem 3 principais paradigmas de programação:
  + Estruturado
  + Funcional
  + Orientado a Objetos
* Recomendação:
  + Para os que tiverem curiosidade sobre o tema, o livro Arquitetura Limpa de Robert Martin, tem alguns capítulos no começo muito interessantes sobre o tema e comparativo entre eles, apesar de ser uma leitura não muito amigável para pessoas pouco familiarizadas com a área.
* Definição:
  + Mas, tentando definir um paradigma de uma forma só um pouco abstrata, podemos começar com a mentalidade de que:  
    *Cada estilo de programação nos força a pensar de um jeito. O paradigma Estruturado, nos fazia ordenar nosso código em blocos bem definidos de execuções e chamadas de funções coordenadas. Por outro lado, o paradigma orientado a objetos, nos força a pensar em termos de coisas (objetos) que sabem se comportar. Em vez de escrever tudo de forma solta, a gente agrupa comportamento com dados em estruturas chamadas classes.*
* Exemplo:
  + Vamos ficar no comparativo entre o paradigma estruturado, que é o que quase todo mundo que começa a programar tem um primeiro contato, e o Orientado a Objetos. Irei deixar o paradigma funcional de lado para essa aula.
  + Vejamos um pequeno trecho de código estruturado contendo uma representação de um carro (em C):

Esse primeiro exemplo é em C. Percebam que os dados do carro estão todos dentro de uma estrutura, mas o que ele **faz** — no caso, a função carro\_andarKm() — está separado. Ou seja, os **dados** e os **comportamentos** estão soltos, desconectados.

Agora vamos ver o mesmo exemplo em C++, usando orientação a objetos.

Aqui nós temos uma **classe Carro**, que junta os dados (marca, modelo, etc.) e os comportamentos (andarKm) no mesmo lugar. Essa é uma das ideias centrais da orientação a objetos: **encapsular** tudo que pertence a um objeto dentro dele.

Assim, o carro 'sabe' como andar, o que deixa o código mais coeso, mais fácil de manter e expandir.

### **3. POO - Classes e Objetos**

* Conceito de robótica e exemplos no cotidiano.
* Conceito de automação e comparação com robótica.
* Pergunta interativa: “Esse exemplo é robótica, automação ou ambos?” (resposta rápida com exemplos simples).

### **4. Microcontroladores e o Cérebro do Robô (10 min)**

* O que é um microcontrolador e onde encontramos no dia a dia.
* Diferença entre computador e microcontrolador.
* Breve analogia: “Microcontrolador = cérebro + sentidos (sensores) + músculos (atuadores)”.

### **5. Introdução ao Arduino (15 min)**

* O que é o Arduino: origem, propósito e por que é tão usado.
* Principais modelos (enfatizar o Arduino Uno).
* Funcionalidades básicas: pinos, USB, alimentação, microcontrolador.
* Mostrar o Arduino (real ou imagem projetada) e identificar componentes.

### **6. Aplicações com Arduino + Demonstração (15 min)**

* Exemplos de projetos reais com Arduino (rápido e visual):  
  + Piscar LED
  + Semáforo
  + Sensor de distância
  + Robô simples
* Demonstração ao vivo (se possível): programa blink.ino com LED piscando.
* Breve explicação sobre o código:  
  + setup()
  + loop()

### **7. Encerramento e Próximos Passos (5 min)**

* Recapitulação dos principais pontos.
* O que esperar da próxima aula: montar primeiro circuito (LED + resistor).
* Tarefa opcional: instalar Arduino IDE.
* Espaço para 1 ou 2 dúvidas rápidas.
* Compartilhar grupo de comunicação e materiais extras.

**Link do circuito no tinkercad :** [Tarefa07](https://www.tinkercad.com/things/j6Bhf2OHuqh-mighty-jaban-borwo?sharecode=mE2RoraCEE35foMk837jNLldHO6jinRHAMxSko6Ycp4)

**Aula Gravada :** [Aula07](https://drive.google.com/file/d/1xMSB4BAXOxklojcOGhFrlS1jyhZCF4k9/view?usp=sharing)