Aula 1: Introdução à Lógica de Programação com Python

Duração Total: 1h e 20min

Objetivo da Aula:

- ☐ **apresentação**
- ☐ **sobre as aulas e atividades**
- ☐ **dinâmica da aula síncrona - contextualização sobre o módulo atual e tira dúvidas e comentários**
- ☐ Introduzir os conceitos fundamentais da lógica de programação.
- ☐ Desenvolver a capacidade de resolver problemas simples usando Python.
- ☐ Ensinar os primeiros passos na criação de algoritmos e execução de programas.

Plano de Aula:

1. Boas-vindas e Introdução à Aula (5 min)

- **Objetivo:** Contextualizar os alunos sobre o que será visto e a importância de aprender lógica de programação.
- lógica é a base para ser um bom programador, pois a forma que se estrutura as soluções é o principal para ter uma boa implementação

- **Conteúdo:**
  - ☐ **Explicar o que são algoritmos:**
    Conjuntos de passos finitos e organizados que quando executados resolvem um determinado problema

    Um exemplo cotidiano seria a receita de um bolo, onde cada passo precisa ser seguido em ordem para que o bolo seja feito corretamente. Assim como em um algoritmo, você tem ingredientes (entrada), o processo de mistura e cozimento (processamento), e o bolo pronto (saída).

    ex:

    - Algoritmo para somar 2 números

- Algoritmo para converter km em metros
- Algoritmo para criar um tutorial de download e instalação

☐ Explicar o que é a lógica de programação.

A **lógica de programação** é o conjunto de raciocínios e regras que usamos para criar soluções em linguagem de computador. Quando falamos de lógica de programação, estamos nos referindo à maneira de pensar e estruturar o passo a passo para resolver um problema de forma clara e precisa, de modo que o computador consiga entender e executar.

Ela envolve alguns conceitos fundamentais:

- **Sequência**: As instruções são executadas uma após a outra, na ordem em que foram escritas.
- **Decisão (ou condição)**: O programa pode escolher diferentes caminhos, dependendo de uma condição. Por exemplo: "se o bolo estiver pronto, retire do forno, senão, continue assando."
- **Repetição**: Certos passos podem ser repetidos várias vezes, até que uma condição seja atendida. Por exemplo: "Continue misturando a massa até ficar homogênea."

☐ Introduzir a linguagem Python e seu uso como uma ferramenta prática para a lógica de programação.
☐ Definir os objetivos da aula: aprender a usar variáveis, operadores, estruturas de controle (condicionais e loops) e como desenvolver pequenos algoritmos.

2. Configuração do Ambiente (5 min)

- **Objetivo:** Certificar-se de que todos têm o ambiente de desenvolvimento Python pronto.
- **Passos:**

- Orientar os alunos a usarem o ambiente de desenvolvimento (IDLE, Jupyter Notebook ou qualquer IDE de preferência, como PyCharm ou Visual Studio Code).
- Certificar-se de que todos conseguem rodar o Python com um simples `print("Olá, Mundo!")`.

3. Introdução à Sintaxe Básica (10 min)

- **Objetivo:** Apresentar a sintaxe básica de Python e variáveis.
- **Conteúdo:**
  - Explicar o conceito de variáveis e tipos de dados (inteiro, float, string, booleano).
  - Mostrar como usar a função `print()` para exibir valores.

Exemplos práticos:
python

```
nome = "João"
idade = 20
altura = 1.75
print(nome, idade, altura)
```

4. Operadores Aritméticos e Expressões Simples (10 min)

- **Objetivo:** Ensinar o uso de operadores básicos (adição, subtração, multiplicação, divisão).
- **Conteúdo:**
  - Introduzir operadores: `+`, `-`, `*`, `/`, `//`, `%`, `**`.
  - Exercício prático:

Calcular a média de três notas:
python

```
nota1 = 7.5
nota2 = 8.0
nota3 = 6.5
media = (nota1 + nota2 + nota3) / 3
print("A média é:", media)
```

■

5. Estruturas Condicionais: `if`, `else` e `elif` (15 min)

- **Objetivo:** Ensinar a lógica condicional e como usá-la para tomada de decisões.
- **Conteúdo:**
    - Explicar como funcionam as estruturas condicionais em Python.

Exemplo prático:
python

```
idade = 18
if idade >= 18:
    print("Você é maior de idade.")
else:
    print("Você é menor de idade.")
```

    -
    - Exercício prático:

Fazer um programa que verifica se um número é par ou ímpar:
python

```
numero = int(input("Digite um número: "))
if numero % 2 == 0:
    print("O número é par.")
else:
    print("O número é ímpar.")
```

6. Laços de Repetição: `for` e `while` (15 min)

- **Objetivo:** Introduzir a repetição e loops.
- **Conteúdo:**
    - Explicar como usar o `for` e o `while` em Python.

Exemplo prático de loop `for`:
python

```
for i in range(1, 6):
    print("Número:", i)
```

    -

Exemplo prático de loop `while`:
python

```
contador = 0
while contador < 5:
    print("Contador:", contador)
    contador += 1
```

-
- Exercício prático:

Fazer um programa que exibe a tabuada de um número escolhido pelo usuário:
python

```
numero = int(input("Digite um número para ver a tabuada: "))
for i in range(1, 11):
    print(numero, "x", i, "=", numero * i)
```

-

7. Pequeno Projeto: Calculadora Simples (15 min)

- **Objetivo:** Aplicar os conceitos ensinados em um pequeno projeto integrado.
- **Conteúdo:**
  - Orientar os alunos a criarem uma calculadora simples que realiza operações básicas de soma, subtração, multiplicação e divisão.

Exemplo de como guiar os alunos:
python

```
print("Escolha a operação:")
print("1 - Soma")
print("2 - Subtração")
print("3 - Multiplicação")
print("4 - Divisão")

opcao = int(input("Digite sua opção (1/2/3/4): "))
num1 = float(input("Digite o primeiro número: "))
num2 = float(input("Digite o segundo número: "))

if opcao == 1:
    print("Resultado:", num1 + num2)
elif opcao == 2:
    print("Resultado:", num1 - num2)
elif opcao == 3:
```

```
    print("Resultado:", num1 * num2)
elif opcao == 4:
    if num2 != 0:
        print("Resultado:", num1 / num2)
    else:
        print("Erro: Divisão por zero!")
else:
    print("Opção inválida.")
```

        ○

8. Resolução de Dúvidas e Revisão (10 min)

- **Objetivo:** Sanar dúvidas dos alunos e reforçar conceitos importantes.
- **Conteúdo:**
    - Revisar os principais tópicos vistos (variáveis, operadores, condicionais, loops).
    - Responder às dúvidas individuais ou coletivas dos alunos.

9. Desafio Final e Encerramento (10 min)

- **Objetivo:** Propor um pequeno desafio para que os alunos testem o conhecimento adquirido.

Desafio: Pedir para que os alunos criem um programa que adivinhe um número entre 1 e 10, com tentativas limitadas. Exemplo:
python

```
import random
numero_secreto = random.randint(1, 10)
tentativas = 3

while tentativas > 0:
    chute = int(input("Adivinhe o número (1-10): "))
    if chute == numero_secreto:
        print("Parabéns! Você acertou.")
        break
    else:
        tentativas -= 1
        print("Errado. Tentativas restantes:", tentativas)

if tentativas == 0:
    print("Você perdeu! O número era:", numero_secreto)
```

-

- **Encerramento:** Agradecer a presença e participação dos alunos e reforçar a importância da prática constante.

Materiais Necessários:

- Computadores com Python instalado.
- IDE ou ambiente para execução de código Python (IDLE, Jupyter Notebook, VSCode, etc.).
- Projetor para demonstrar os códigos e a execução dos exercícios.

Dicas Extras para Tornar a Aula Didática:

- Incentive os alunos a testarem suas próprias modificações nos exemplos.
- Promova a interação entre os alunos para que resolvam os problemas juntos.
- Esteja sempre disponível para tirar dúvidas individualmente, mantendo o ritmo da aula.
- Ofereça exemplos simples e claros, sem complicar os primeiros conceitos.

Conclusão:

Essa estrutura oferece uma abordagem prática, passo a passo, introduzindo os alunos à lógica de programação com Python de forma acessível e interativa. Com exercícios práticos e um pequeno projeto integrador, os alunos terão a chance de aplicar o que aprenderam, consolidando os conceitos fundamentais da programação.

```
`# Função para preparar os ingredientes
def preparar_ingredientes():
    print("1. Separe os ingredientes:")
    print(" - 3 xícaras de farinha de trigo")
    print(" - 2 xícaras de açúcar")
    print(" - 4 ovos")
    print(" - 1 xícara de leite")
    print(" - 1/2 xícara de óleo")
    print(" - 1 colher de sopa de fermento em pó")
    print(" - 1 colher de chá de essência de baunilha\n")

# Função para misturar os ingredientes secos
def misturar_ingredientes_secos():
```

```
    print("2. Misture os ingredientes secos:")
    print(" - Farinha, açúcar e fermento\n")

# Função para adicionar os ingredientes líquidos
def adicionar_ingredientes_liquidos():
    print("3. Adicione os ingredientes líquidos:")
    print(" - Ovos, leite, óleo e essência de baunilha")
    print(" - Mexa até a mistura ficar homogênea\n")

# Função para colocar a massa no forno
def assar_bolo():
    print("4. Pré-aqueça o forno a 180°C")
    print("5. Despeje a massa em uma forma untada e enfarinhada")
    print("6. Asse por 40 minutos\n")

# Função para finalizar a receita
def finalizar():
    print("7. Retire o bolo do forno e deixe esfriar")
    print("8. Aproveite seu bolo delicioso!")

# Função principal que executa todas as etapas da receita
def fazer_bolo():
    preparar_ingredientes()
    misturar_ingredientes_secos()
    adicionar_ingredientes_liquidos()
    assar_bolo()
    finalizar()

# Executa o algoritmo da receita de bolo
fazer_bolo()

`
```

```
# Função para calcular a área de um triângulo usando base e altura
def calcular_area_com_altura(base, altura):
    # Fórmula da área do triângulo
    area = (base * altura) / 2
    return area

# Função principal que executa o algoritmo
def main():
    print("Cálculo da área de um triângulo\n")

    # Solicita a base e a altura do triângulo ao usuário
```

```
    base = float(input("Digite o valor da base do triângulo: "))
    altura = float(input("Digite o valor da altura do triângulo: "))

    # Calcula a área
    area = calcular_area_com_altura(base, altura)

    # Exibe o resultado
    print(f"\nA área do triângulo é: {area:.2f}\n")

    # Desenha o triângulo
    desenhar_triangulo()

# Executa o programa
main()
```