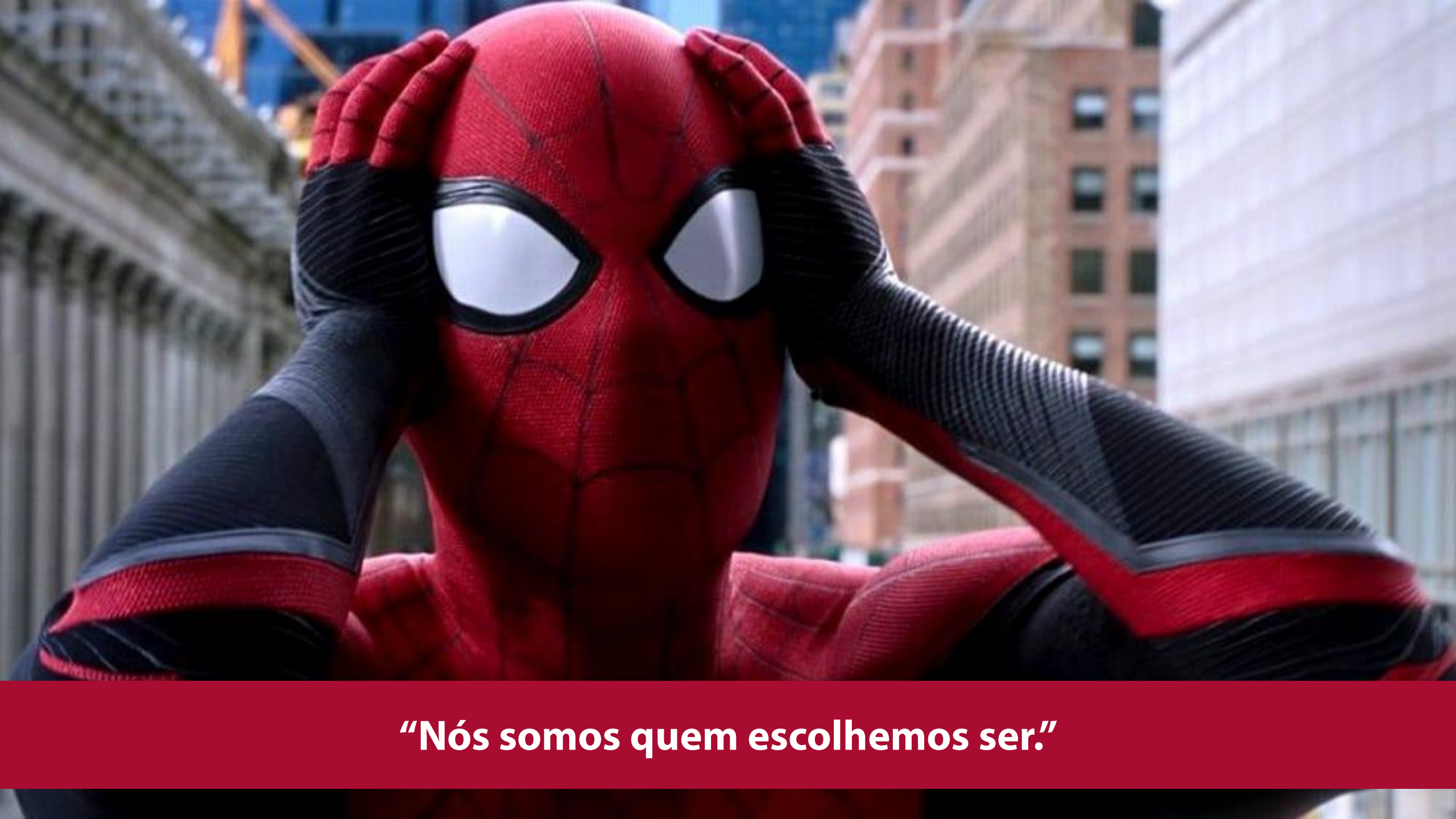
Programação Orientada a Objetos

Prof.ª Ma. Jessica Oliveira

Católica
Universidade Católica de Brasília



“Nós somos quem escolhemos ser.”

"Nenhuma quantia de dinheiro jamais comprou um segundo de tempo."

“Faça seus medos terem medo de você.”

“Sua vontade determina seus limites.”

Apresentação da Docente

beacons.ai/oijessicaoliveira

Católica
Universidade Católica de Brasília

Avaliação Diagnóstica

Católica
Universidade Católica de Brasília

Regras

- Quando um discente sinalizar que terminou de resolver o desafio, o tempo será pausado para que o mesmo apresente sua solução;
- Se o discente que for até o quadro apresentar não acertar a resposta ou não conseguir fazer a docente compreender a solução, os demais alunos continuam (com o tempo que houver restado) e quem terminar primeiro terá a chance de apresentar;
- Será recompensado o discente que terminar primeiro e conseguir acertar a resposta e explicar corretamente a solução.

Etapa 01

Nível fácil.

Católica
Universidade Católica de Brasília

Desafio 01 - Algoritmo Simples

- Escrever um algoritmo, em pseudocódigo, para calcular a média de três números.
- **Tempo:** até 3 minutos.

```
INICIO
    // Declaração de variáveis
    DECLARE num1, num2, num3, media COMO REAL

    // Entrada dos três números
    ESCREVA "Digite o primeiro número:"
    LEIA num1
    ESCREVA "Digite o segundo número:"
    LEIA num2
    ESCREVA "Digite o terceiro número:"
    LEIA num3

    // Cálculo da média
    media ← (num1 + num2 + num3) / 3

    // Exibição do resultado
    ESCREVA "A média dos números é: ", media
FIM
```

Desafio 02 - Teste Condicional

- Escrever um algoritmo, em pseudocódigo, que verifique se um número é positivo, negativo ou zero.
- **Tempo:** até 5 minutos.

```
INICIO
    // Declaração de variável
    DECLARE numero COMO INTEIRO

    // Entrada do número
    ESCREVA "Digite um número:"
    LEIA numero

    // Verificação do sinal
    SE numero > 0 ENTAO
        ESCREVA "O número é positivo."
    SENAO SE numero < 0 ENTAO
        ESCREVA "O número é negativo."
    SENAO
        ESCREVA "O número é zero."
    FIMSE
FIM
```

Etapa 02

Nível médio.

Católica
Universidade Católica de Brasília

Desafio 03 - Operações Matemáticas

- Escrever um programa em C que leia dois números inteiros e exiba a soma, subtração, multiplicação e divisão deles.
- **Tempo:** até 10 minutos.

```c
#include <stdio.h>

int main() {
    int num1, num2;

    // Entrada dos números
    printf("Digite o primeiro número inteiro: ");
    scanf("%d", &num1);
    printf("Digite o segundo número inteiro: ");
    scanf("%d", &num2);

    // Operações matemáticas
    printf("Soma: %d\n", num1 + num2);
    printf("Subtração: %d\n", num1 - num2);
    printf("Multiplicação: %d\n", num1 * num2);

 // Tratamento da divisão para evitar erro de divisão por zero
    if (num2 != 0) {
        printf("Divisão: %.2f\n", (float)num1 / num2);
    } else {
        printf("Divisão: Indefinida (divisão por zero não permitida).\n");
    }

    return 0;
}
```

Desafio 04 - Estruturas de Controle

- Escrever um programa em C que utilize um loop para calcular o fatorial de um número inteiro.
- **Tempo:** até 10 minutos.

```c
#include <stdio.h>

int main() {
    int num;
    long long fatorial = 1;

    // Entrada do número
    printf("Digite um número inteiro: ");
    scanf("%d", &num);

    // Validação para números negativos
    if (num < 0) {
        printf("Fatorial não definido para números negativos.\n");
    } else {
        for (int i = 1; i <= num; i++) {
            fatorial *= i;
        }
        printf("Fatorial de %d é %lld\n", num, fatorial);
    }

    return 0;
}
```

Etapa 03

Nível difícil.

Católica
Universidade Católica de Brasília

Desafio 05 - Programa de Números Primos

- Escrever um programa em C que exiba todos os números primos entre 1 e 1000.
- **Tempo:** até 15 minutos.

```c
#include <stdio.h>

int ehPrimo(int num) {
    if (num < 2) return 0;
    for (int i = 2; i * i <= num; i++) {
        if (num % i == 0) return 0;
    }
    return 1;
}

int main() {
    printf("Números primos entre 1 e 1000:\n");

    for (int i = 1; i <= 1000; i++) {
        if (ehPrimo(i)) {
            printf("%d ", i);
        }
    }

    printf("\n");
    return 0;
}
```

Etapa 04

Nível avançado.

Católica
Universidade Católica de Brasília

Desafio final: Conversão de Código

- Converter o código entregue de pseudocódigo para C e, depois, de C para Java.
- **Tempo:** até 20 minutos.

```
Início
    Escreva "Digite um número inteiro positivo:"
    Leia numero

    Se numero <= 1 então
        Escreva "Não é um número primo."
    Senão
        primo ← verdadeiro
        Para i de 2 até numero - 1 faça
            Se numero mod i = 0 então
                primo ← falso
                Pare
            FimSe
        FimPara

        Se primo = verdadeiro então
            Escreva "O número é primo."
        Senão
            Escreva "O número não é primo."
        FimSe
    FimSe
Fim
```

```c
#include <stdio.h>

int main() {
    int numero, i, primo = 1; // primo = 1 (verdadeiro), primo = 0 (falso)

    printf("Digite um número inteiro positivo: ");
    scanf("%d", &numero);

    if (numero <= 1) {
        printf("Não é um número primo.\n");
    } else {
        for (i = 2; i < numero; i++) {
            if (numero % i == 0) {
                primo = 0; // Número não é primo
                break;
            }
        }

        if (primo) {
            printf("O número é primo.\n");
        } else {
            printf("O número não é primo.\n");
        }
    }

    return 0;
}
```

```
import java.util.Scanner;

public class VerificaPrimo {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Digite um número inteiro
positivo: ");
        int numero = scanner.nextInt();
        scanner.close();

        if (numero <= 1) {
            System.out.println("Não é um número
primo.");
        } else {
            boolean primo = true;
            for (int i = 2; i < numero; i++) {
                if (numero % i == 0) {
                    primo = false;
                    break;
                }
            }
            if (primo) {
                System.out.println("O número é
primo.");
            } else {
                System.out.println("O número não
é primo.");
            }
        }
    }
}
```

Apresentação da Disciplina

Plano de ensino disponível no AVA.

Católica
Universidade Católica de Brasília

Avaliações

Composição das notas bimestrais.

Católica
Universidade Católica de Brasília

- Os dois bimestres seguirão a seguinte divisão de pontos:

  - Prova individual teórica: **1,5 ponto**;
  - Exercícios pré-aula e práticas individuais realizadas em sala: **1,0 ponto**;
  - Projetos, em equipes, compostos pelas entregas via AVA: **2,0 pontos**.

- Além disso, há o ponto do Programa Protagonismo Discente (PPD), que soma mais **1,0 ponto** na nota do semestre.

Metodologia de Ensino

Proposta para o semestre.

Católica
Universidade Católica de Brasília

Observações da Docente

Católica
Universidade Católica de Brasília

- A matemática é uma ciência EXATA! 6,99 É DIFERENTE DE 7,00! Assim, a nota a ser lançada será **aquela oriunda do resultados obtidos pelo discente** em suas atividades pontuadas em sala, ou fora desta, e na avaliação bimestral.
- Resultados maiores ou iguais a 6,50 serão avaliados individualmente com base nos seguintes critérios:

    - Interesse do aluno, mensurado pela quantidade de atividades executadas;
    - Conhecimento a ser revisto, reavaliando possíveis lacunas de notas obtidas em atividades em sala, ou fora desta, e;
    - Frequência.

- Resultados abaixo de 6,50 serão lançados *ipsis litteris*.
- Não existe "ajuda" (dar nota), isso é fora da ética de qualquer profissão!
- Entrega de trabalhos fora de prazo, só em casos especiais e acordados entre docente e aluno, além de serem justificados.
- No dia das avaliações:

    - Celular **DESLIGADO OU EM MODO SILENCIOSO**;
    - Saída da sala: **UM** discente por vez, **SEM** o celular;
    - Ao término da prova, favor não ficar no corredor.

Sugestões

- Não quer assistir a aula? Não atrapalhe quem quer aprender!
- Você paga pela sua graduação por que? Já se perguntou isso?
- Sair de casa todas as manhãs, de ônibus, van ou carro, se furtar da segurança e do aconchego do seu lar e da sua família para ficar no WhatsApp e/ou batendo papo em sala?
- Não culpe os outros pelo seu descaso! Lembre-se:

"O plantio é livre, mas a colheita é obrigatória!"

Aula 01 – 17/02/2025

Introdução à Programação Orientada a Objetos e Versionamento.

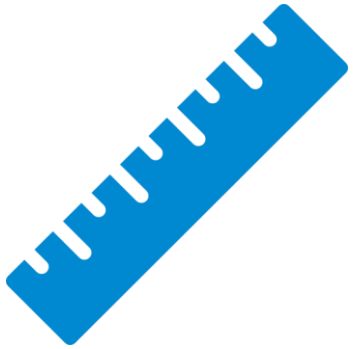
Universidade Católica de Brasília

O que é um algoritmo?

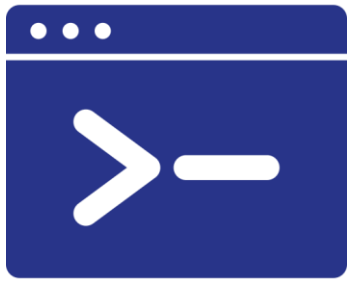
- Conjunto finito e ordenado de instruções que devem ser seguidas para resolver um problema ou realizar uma tarefa específica.
- Pode ser implementado em qualquer linguagem de programação ou descrito em pseudocódigos ou fluxogramas.

Características dos algoritmos

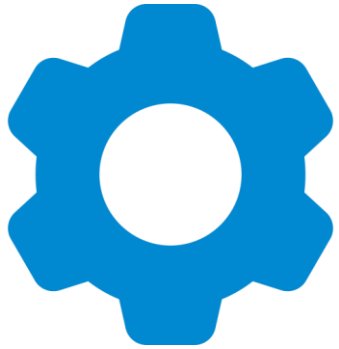
Finitude



Clareza e Precisão



Entrada e Saída



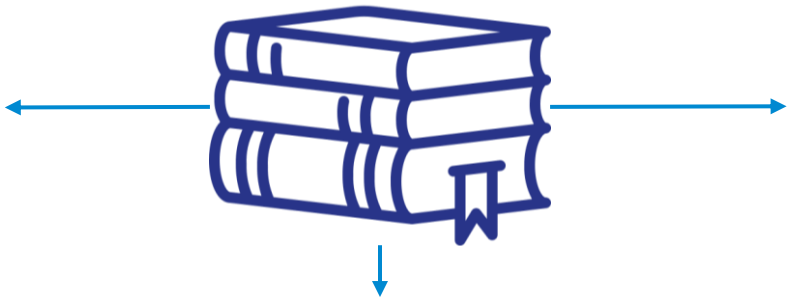
Efetividade

Programação Orientada a Objetos

- É um paradigma de programação que organiza o código em **objetos**, que possuem **atributos** e **métodos**.

Classe:
Livro.

Atributos:
Título;
Autor;
Número de páginas.



Métodos:
Abrir;
Fechar;
Ler.

Objeto:
Harry Potter.

Procedural vs. POO

Programação Procedural	Programação Orientada a Objetos
Código linear e sequencial	Código baseado em objetos
Funções separadas	Objetos com métodos e atributos
Difícil de manter	Fácil de manter e reutilizar

Por que o mercado prefere a POO?

- **Reutilização de código:** classes reutilizáveis.
- **Facilidade de manutenção:** código modular e mais fácil de atualizar.
- **Modularidade:** código mais organizado.
- **Modelagem do mundo real:** objetos representam elementos reais.

Resumindo...

- **Algoritmos** são sequências de passos para resolver problemas, enquanto a **programação orientada a objetos** organiza o código em classes e objetos, facilitando a reutilização, manutenção e representação do mundo real.

Microprojeto 01

🎯 **Objetivo:** compreender os paradigmas de programação e configurar o ambiente de desenvolvimento e versionamento.

Para realizar em sala...

- Divisão da turma em **quatro equipes**;
- Definição do **projeto a ser desenvolvido**;
- Definição do **nome do repositório no GitHub** (e do nome da equipe, se quiserem);
- Estabelecer um canal de comunicação eficiente para a equipe;
- Criar um quadro de tarefas (Trello, Notion...) para auxiliar na documentação e divisão de tarefas;
- Criação do **rascunho para o `README.md` inicial**, descrevendo brevemente: nome do projeto, o propósito do sistema e a equipe responsável (nome dos integrantes);

- Organização das funções na equipe: planejem como irão dividir as tarefas dentro da equipe e escolham **quem ficará responsável por cada papel**, lembrando que todos podem, e devem, contribuir com o código:

  - **Gerente do Projeto:** responsável por organizar tarefas e prazos da equipe, além de fazer as entregas no AVA pela equipe.
  - **Líder de Versionamento:** responsável por garantir que o Git/GitHub esteja sendo usado corretamente.
  - **Desenvolvedores:** programam as funcionalidades do sistema.
  - **Documentador:** atualiza o `README.md` e auxilia na escrita dos relatórios.
  - ***Tester*/Validador:** testa o código e verifica se as funcionalidades estão corretas.

Para o pós-aula...

- Criar e configurar o repositório no GitHub.
- Criar um arquivo **`README.md`** com informações básicas do projeto.
- Criar a estrutura inicial do projeto em Java no VS Code.
- Demonstrar o uso do Git para versionamento (commit inicial, configuração do **`.gitignore`**).
- Relatório técnico explicando a configuração do ambiente e a importância do versionamento.

Na próxima aula...

Aula 02 (24/02/2025) - Classes, Objetos, Atributos e Métodos.

Dúvidas?

jessica.oliveira@p.ucb.br