

# **INTRODUÇÃO**

**USANDO VS CODE, GITHUB E GITHUB  
COPILOT PARA PROGRAMAÇÃO EM C/C++**

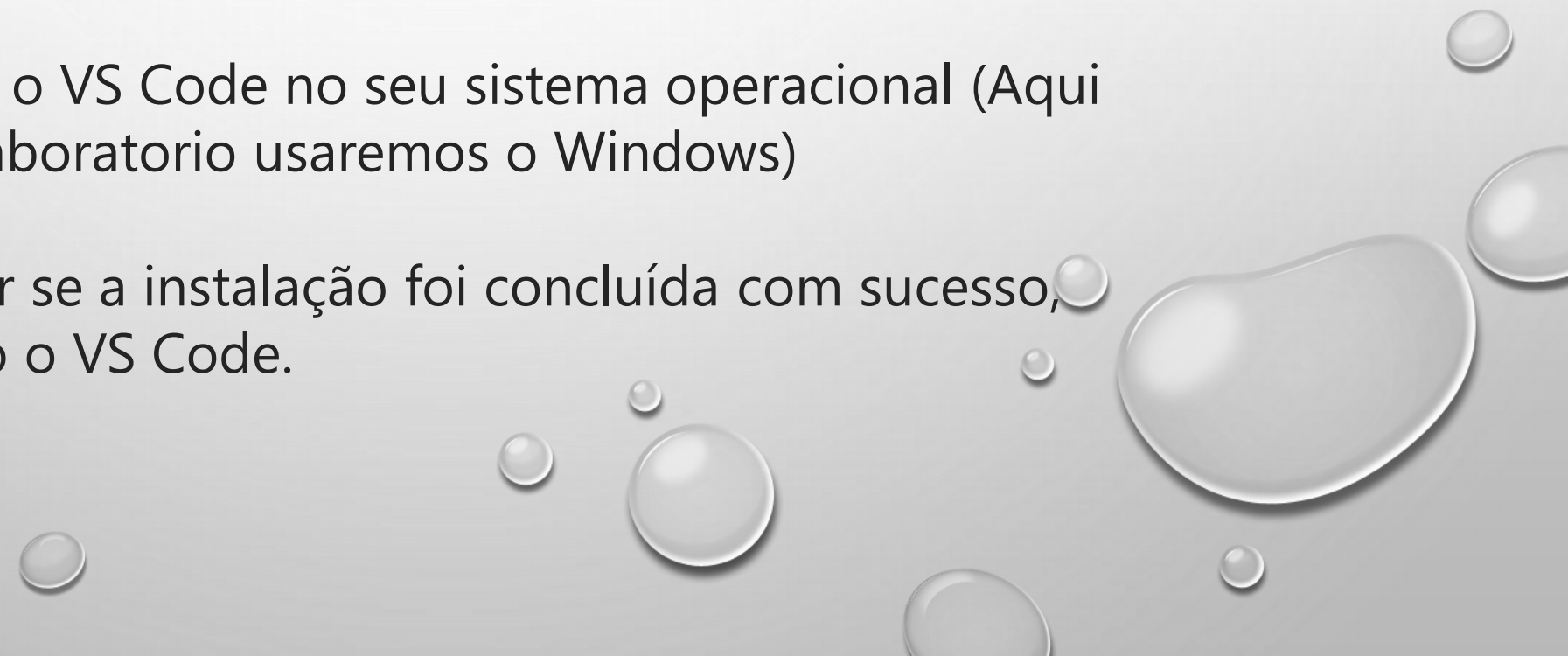
**OBJETIVO: MOSTRAR COMO CONFIGURAR  
E UTILIZAR O VS CODE, GITHUB E GITHUB  
COPILOT PARA FACILITAR O  
DESENVOLVIMENTO EM C/C++.**

**PROF. CLAUDIA J. BARENCO ABBAS**



# Instalando o Visual Studio Code

## Passos:

- Baixar o VS Code em <https://code.visualstudio.com>
  - Instalar o VS Code no seu sistema operacional (Aqui neste laboratório usaremos o Windows)
  - Verificar se a instalação foi concluída com sucesso, abrindo o VS Code.
- 

# Configurando o Ambiente para C/C++ no VS Code

## •Instalar Extensões:

- 1.Abra o VS Code e vá para a **Visualização de Extensões** (ícone de quadrado).
- 2.Procure e instale a extensão **C/C++** da Microsoft.
- 3.(Opcional) Instale a extensão **CMake Tools** se estiver usando CMake para compilar seus projetos.

## •Configuração do compilador:

- 1.Instale um compilador C/C++ (ex: **GCC** no Linux ou **MinGW** no Windows).
- 2.Adicione o caminho do compilador à variável de ambiente **PATH** (No Windows fica no Painel de Controle/Propriedades do Sistema)

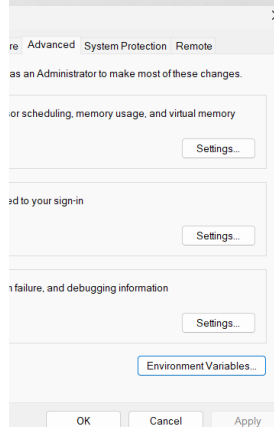
## •Configurar tarefas de build (Opcional):

No VS Code, crie um arquivo tasks.json para definir as tarefas de compilação.

Na aula usaremos o **terminal** para compilar os programas.

Exemplo: **g++ -Wall -std=c++11 helloworld.cpp -o helloworld**

**<O caminho para sua pasta onde esta o seu programa> ./helloworld**



# **CRIANDO UM REPOSITÓRIO NO GITHUB**

## **PASSOS:**

CRIE UMA CONTA NO [GITHUB](https://github.com) (CASO AINDA NÃO TENHA).

CLIQUE EM **NEW REPOSITORY**

DEFINA UM NOME, DESCRIÇÃO E A VISIBILIDADE DO REPOSITÓRIO.

SELECIONE **INITIALIZE THIS REPOSITORY WITH A README.**

CLIQUE EM **CREATE REPOSITORY.**

# Configurando o Git no VS Code

## Passos:

No VS Code, abra o terminal integrado.

Execute:

```
git config --global user.name "Seu Nome"
```

```
git config --global user.email "seu-email@dominio.com".
```

Faça o download do Git (<http://git-scm.com>) .

Atualize o PATH do seu Sistema operacional com o caminho do Programa Git (Como fez anteriormente)

Clone seu repositório GitHub com o comando:

```
git clone https://github.com/usuario/nome-repositorio.git
```

Faça suas edições no código e adicione, commit e envie as alterações usando o Git.

# PRINCIPAIS COMANDOS DO GIT

## Configuração Inicial

### •Configurar o nome do usuário:

git config --global user.name "Seu Nome"

### •Configurar o email do usuário:

git config --global user.email "seuemail@dominio.com"

## Iniciar um Repositório

### •Criar um novo repositório Git:git init

## Verificar o Status do Repositório

### •Verificar mudanças no repositório:git status

## Adicionar Arquivos para Commit

### •Adicionar arquivos específicos:

git add nome-do-arquivo

### •Adicionar todos os arquivos modificados:

git add .

## Realizar um Commit

### •Salvar mudanças no repositório local:git commit -m "Mensagem explicativa"

## Ver o Histórico de Commits

### •Exibir o histórico de commits:git log

## Atualiza todos os arquivos locais no Git

git push -u origin Local

**origin** – Nome da branch Remota (normalmente é origin)

**Local** – Nome da branch Local

# Git Commit

No Git, **commit** e **push** são dois comandos fundamentais, mas que têm funções diferentes no processo de versionamento de código.

## 1. git commit

**O que faz?** O comando git commit é usado para **salvar mudanças** no seu repositório local. Ele registra as alterações feitas no código no histórico de commits do seu repositório local.

•**Quando usar?** Sempre que você fizer alterações no seu projeto e quiser **salvar** essas mudanças de forma organizada e com um comentário explicativo.

Mas lembre-se, as alterações ficam apenas no seu repositório local, elas ainda não são enviadas para o repositório remoto (por exemplo, no GitHub ou GitLab).

### Exemplo de uso:

```
git commit -m "Adicionando nova funcionalidade X"
```

•**O que acontece?** O commit cria uma nova "versão" do código com base nas mudanças que você fez, e essas mudanças estão salvas no seu repositório local. Cada commit é associado a uma mensagem e a um identificador único (hash).

# GIT PUSH

- **O que faz?** O comando git push é usado para **enviar** os commits do seu repositório local para um repositório remoto. Isso compartilha as suas mudanças com outras pessoas ou com um servidor de backup como o GitHub.
- **Quando usar?** Depois de ter feito um commit e quiser **enviar** as suas mudanças para um repositório remoto (por exemplo, GitHub, GitLab, Bitbucket), para que outros colaboradores possam ver ou acessar as alterações feitas.
- **Exemplo de uso:**  
git push origin main
- **O que acontece?** O comando git push envia os commits realizados na sua branch local (por exemplo, main) para a branch remota correspondente.
- Isso faz com que outras pessoas possam ver as suas alterações ou que o código seja sincronizado com o repositório remoto.



# Fluxo de trabalho típico

1. **Faça alterações no código.**
2. **Use `git add`** para adicionar os arquivos modificados ao índice (staging area).
3. **Use `git commit`** para salvar essas alterações no repositório local.
4. **Use `git push`** para enviar seus commits para o repositório remote no GitHub.

# Usando GitHub Copilot

## Instalar GitHub Copilot:

- No VS Code, vá para a **Visualização de Extensões**.
- Pesquise por **GitHub Copilot** e instale a extensão.
- Após a instalação, faça login com sua conta do GitHub.

## Como usar:

- Comece a escrever um código e o GitHub Copilot sugerirá automaticamente complementos.
- Utilize as sugestões pressionando **Tab** para aceitar.
- Explore a funcionalidade de auto-completar e sugestões de código em tempo real.

# FLUXO DE TRABALHO: DESENVOLVENDO COM VS CODE, GITHUB E GITHUB COPILOT

- **CRIAÇÃO DE UM NOVO PROJETO:** CRIE UM REPOSITÓRIO NO GITHUB, CLONE NO VS CODE E INICIE O CÓDIGO.
- **ESCREVA O CÓDIGO EM C/C++** NO VS CODE COM A AJUDA DO GITHUB COPILOT.
- **CONTROLE DE VERSÃO:** UTILIZE O GIT NO VS CODE PARA VERSIONAR SEU CÓDIGO, FAZENDO COMMITS E PUSH PARA O GITHUB.
- **AUTO-COMPLETAR E SUGESTÕES:** USE O GITHUB COPILOT PARA ACELERAR O PROCESSO DE ESCRITA DO CÓDIGO.
- **COMPILAÇÃO E EXECUÇÃO:** COMPILE E EXECUTE O CÓDIGO DIRETAMENTE DO VS CODE USANDO O TERMINAL INTEGRADO.



# DICAS E BOAS PRÁTICAS

- **Uso de Branches:** Crie branches para trabalhar em diferentes funcionalidades ou matérias.
  - **Commits frequentes:** Faça commits pequenos e frequentes.
  - **Sincronização:** Sincronize seu repositório local com o GitHub para manter seu trabalho seguro.
- 