Modulo-0

Decomposição **>** Reconhecimento de padrões **>** Abstração **>** Design de algoritmos

Raciocínio lógico

Indução: Leis, Teorias – Fenômeno observado.

Dedução: Leis, Teorias – Previsões e explicações.

Abdução: Diagnósticos, Processo investigativo – Premissa. Ex: “ *A grama está molhada, logo deve ter chovido.*” (talvez sejá).

Inferência:

Sintética: Conclusão experimentação

Abdução e Indução.

Analítica: Dedução e exatas.

Dedução.

Aperfeiçoamento:

**Ato de aperfeiçoar**

* Encontrar solução eficiente;
* Otimizar processos;
* Simplificar linhas de códigos;
* Funções bem definidas.

Decomposição:

* Identificar
* Determinar as etapas
* Executar cada etapa
* Agregar tudo ao finalizar

Ex: Aplicativo

* Finalidade
* Interface
* Funcionalidades **Definição de componentes**
* Pré-requisitos

Padrões:

* Modelo
* Estrutura invariante **Similaridades e Diferenças**
* Repetição

Abstração:

Generalização

Algoritmos:

* Receber
* Manipular
* Armazena

Como construir um algoritmo?

1. Compreensão do problema
2. Definição dados de entrada
3. Definir processamento
4. Definir dados de saída
5. Utilizar um método de construção
6. Teste e diagnostico.

Construção:

**Narrativa:** Utilização da linguagem natural, diversas interpretações.

**Fluxograma**: Utilização de símbolos pré-definidos, conhecimento prévio da estrutura e dos símbolos.

**Pseudocódigo**: Ex: Portugol

Modulo-1

GIT x GITHub

Git: sem interface, apenas CLI

Comandos básicos:

Windows:

* cd – Entrar / sair
* dir – listar diretórios
* mkdir – criar pastas
* del / rmdir – deletar arquivos / deletar diretórios (/s /q)

Linux:

* cd – Entrar / sair
* ls – listar diretórios
* mkdir – criar pastas
* rm / rf – deletar arquivos / deletar diretórios (rm -rf )

INSTALAÇÃO PADRÃO DO GIT 2.33 +

Como o Git funciona por baixo dos panos:

**SHA1:**

hash, algoritmo de criptografia, gera um conjunto de 40 arquivos.

**Objetos fundamentais:**

**Blobs:** Estrutura básica do Tipo: Blob | tamanho: 9 | \0 | conteudo

São iguais quando ex:

*echo ‘conteudo’ | git hash-object –stdin*

*echo -e ‘blob 9\0conteudo’ | openssl sha1*

*Git armazena meta dados !*

**Trees:** Estrutura básica Tree, armazenam o blob.

Tipo: Tree | \0 | tamanho “?” | blob | sa4d8s | texto.txt <- conteúdo

Armazena meta dados também, ela possui \0, aponta para o blob que por sua vez possui um sha1 do arquivo, guarda o nome do arquivo também, e possuem o sha1 dos metadados.

**Commits:** Objeto que vai juntar tudo, vai dar sentido a tudo.

Ele contém a Tree, parente, autor, mensagem “first commit”, timestamp.

Significam uma alteração.

Possui um sha1 também de tudo.

**Sistema distribuído segurança:**

Pela forma como foi projetado, o tornou seguro.

Chave SSH e Token

Chave SSH:

A grosso modo, é uma forma de se realizar uma conexão segura entre duas maquina.

Uma chave publica e uma chave privada.

Colocar a nossa chave pública no github, para que o github já conheça a chave da nossa maquina.

Ir em Perfil > Settings > SSH Keys... > New SSH .

Agora vamos gerar uma chave SSH pelo Git BASH.

Clicar em Enter para concordar em salvar a chave no local padrão, definir uma senha, após isso, dar um cd na pasta onde a chave foi criada, e em seguidar dar um cat para visualizar a chave .pub, copiar ela e colar na tela do Github.

Escreva: ssh-keygen -t ed25519 -C guilherme\_fa@outlook.com

Após ter salvo no Git ativar o ssh-agent:

eval $(ssh-agent -s)

Passar para o ssh-agent a chave privada:

ssh-add id\_ed25519

Clonar um repositório com uma chave ssh configurada.

*git clone “link ssh”*

Token de acesso Pessoal

Congurar um token, no GitHub vá em :

Ir em Perfil > Settings > Developer settings.> Personal access tokens

Colocar Note (nome), uma data de expiração e marcar em **Repo (all)**

**Copiar o Token e guardar em algum lugar, pois não vai ser mais possível ver ele novamente.**

Usar o protocolo HTTPS para o git clone:

*git clone “link HTTPS”*

PRIMEIROS COMANDOS DO GIT

* **git init** – Iniciar o repositório do git.
* **git add** – Adicionar um arquivo para commitar.
* **git commit** – onde a magica acontece, criar um commit.

Criar repositório:

Clicar sobre a pasta com o direito com o GIT Bash, já vai abrir no diretório direto.

Criar a pasta, dar um git init para iniciar o repositório.

È preciso configurar o Git com os seguintes comandos:

*git config --global user.email guilherme\_fa@outlook.com*

depois

*git config --global user.name Gui\_Anacleto*

Vamos criar um arquivo Markdown -> README.md

E adicionar ele a fila do GIT:

*git add “name”*

Ou para pegar todos os arquivos do diretório:

*git add \**

Após isso basta dar o commit:

*git commit –m “commit inicial”*

Após dar o commit, percebera que será exibido o inicio do sha1 dele.

CICLO DE VIDA DO FILE GIT

**Git Init**

**Tracked:**

* **Unmodified – Não alterado**

Um arquivo no Git que foi modificado de **Unmodified** para **Modified**

Rodar o **Git add** vai mandar para **Staged**(aguardardando)

Quando removemos ele ele volta para **Untracked**

* **Modified - Alterado**

Aquivo modificado.

* **Staged - Aguardando**

Arquivo aguardando o comiit

**Untracked:** Não sabe da existência dele.

*git add* -> Manda o Untracked para Staged(aguardando)

**Commit:**

Todos os arquivos saem do **Staged** vão para para **Unmodified**, para que sejam commitados.

**Servidor:** Remote repositor

**Ambiente de desenvolvimento:** Working directory(repo. trabalho), Staging área(área de espera \**GIT ADD*\* ), Local Repository(*commit* -> passa a integrar seu repositório local para poder ser empurrado para servidor.).

*git status* – para verificar os status dos arquivos.

*mkdir receitas*

*mv name\_fiel ./receitas* – mover para pasta receitas

*git add . – Adicionar todas as alterações para o Staged área.*

TRABALHANDO COM GIT

Para trazer todas as configurações do seu git digite:

*git config --list*

Para limpar as configurações:

Limpa email: git config --global --unset user.email

Limpa name: git config --global --unset name.email

Para reconfigurar

git config --global user.name “name”

git config --global user.email “email”

Criar um repositório.

E copiar sua URL do HTTPS de caminho:

Ex:

https://github.com/Gui-Anacleto/Bootcamp-Santander-2022.git

Configurar o repositório remoto com o comando:

git remote add origin https://github.com/Gui-Anacleto/Bootcamp-Santander-2022.git

Para listar os repositórios cadastrados use:

git remote -v

Para terminar de empurrar para o repositório local basta:

git push origin master