- Prof. Renato Jardim Parducci

* Texto

  Descrição gerada automaticamenteLinha do tempo

  Descrição gerada automaticamente com confiança médiaGovernança e Melhores Práticas em SI
* ProjetoCalculadoraDigital
* Interface gráfica do usuário, Texto, Aplicativo, Email

  Descrição gerada automaticamenteCriando uma classe teste:

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

* Unit: teste unitário
* O teste não adiciona valor (deve ser inteligente e rápido), apenas verifica se o recurso foi implementado corretamente.
* JUNIT: automação por script de teste
* O teste é feito com base no requisito, em cima de uma especificação.

- Requisito (base dos testes)

* Texto

  Descrição gerada automaticamenteCriar uma classe Calculadora que faça as operações (métodos) de somar, subtrair, multiplicar, dividir, sendo que, todas elas farão operações sobre dois números inteiros recebidos por parâmetro e devolverão 1 número inteiro resultante. Na subtração, a segunda variável dos parâmetros, será subtraída da 1ª. Na divisão, a 1ª variável do parâmetro serão divididas pela 2ª. Considere que um usuário digitará na sequência o 1º e o 2º parâmetro usados (a digitação ocorrerá fora da classe Calculadora). Na soma e na multiplicação, a ordem dos fatores não é relevante.
* É preciso usar AssertEquals para a comparação e teste dos resultados

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

* Interface gráfica do usuário, Texto, Aplicativo

  Descrição gerada automaticamenteJUnit Test Suit: possibilita a junção dos testes criados para que eles sejam executados em conjunto.
* 26/02/22: até 23:00 entregar no portal do aluno o Checkpoint1
* GITHUB: flow, processo de fluxo de controle de versão

- Branch: versão de produção, que só será combinada ao final de seu desenvolvimento

Isso evita que eventuais bugs interfiram na versão atual

- Read.me -> hystory guarda a as versões anteriores

* Compare & pull request:

- Ao acabar de usar a branch que foi combinada, deletá-la, pois, ela parou de ter uso

* Interface gráfica do usuário, Texto, Aplicativo, Email

  Descrição gerada automaticamenteMUDAR a base para versão nova, NÃO atualizar a main

- Branches -> delete

* Ao acabar o projeto, atualizar a main e deletar a cópia de teste
* Sempre que for fazer um teste, criar uma cópia nova para teste (protegendo o main)
* Interface gráfica do usuário, Texto, Aplicativo, Email

  Descrição gerada automaticamenteGIT CMD: tem acesso ao fluxo e é possível utilizar a janela ao mesmo tempo da programação
* git config --global user.email [biamayat@gmail.com](mailto:biamayat@gmail.com)
* git clone <https://github.com/BiaMaya/3SIB-SistemaModelo-2022>

Login dentro do git e clonação do repositório

* dir: ver diretórios
* cd 3SIB-SistemaModelo-2022
* git branch
* git flow init
* git flow feature start TarefaModelagemAluno (criar branch para manutenção)
* usar git branch para saber onde você está
* git status: detecta alterações
* git diff README.md (mostra o que foi alterado no doc README.md)

As alterações não serão realmente feitas até o commit

* git add . (adiciona todos os modificados para dar commit)
* git add README.md (adiciona apenas README.md)
* git commit -m "inclusão de informações gerais" (dá commit e coloca uma mensagem)
* git checkout develop (vai para a branch develop)
* git checkout feature/TarefaModelagemAluno (volta para a outra branch)

O git controla a pasta do Windows

* git flow feature finish TarefaModelagemAluno (faz o pull request do GitHub)
* git push -u origin develop (manda atualizar o GitHub da internet)
* git flow release start SI-Versao01 (camada final de testes antes de commit na main)
* Tudo está em C:\Users\bilut\3SI-Reforco