Informações

Cabeçalho

- 18/04/2019
- Brasília, DF, Brasil
- Software Básico
- Turma: A - 01/2019
- Professor: Marcelo Ladeira

Autores

- **Bruno Sanguinetti R. Barros** | *18/0046063* | GitHub
- **Gabriel Vasconcelos** | *0000000000* | GitHub
- **Leonardo de Almeida** | *0000000000* | NULL
- **Lucas Mafra** | *12/0126443* | GitHub
- **Wladimir Gramacho** | *0000000000* | GitHub

Getting Started

Essas instruções farão com que você tenha uma cópia deste projeto em sua máquina local para fins de desenvolvimento e teste.

Pré-requisitos

Para roda os devidos teste nesse software você precisa ter instalado em sua máquina o googletest, uma biblioteca de testes unitários para a linguagem de programação C/C++.

Por favor leia TUTORIAL GTEST INSTALL para mais detalhes sobre como instalar o gtest

Por favor leia GTEST INFO para mais detalhes sobre o gtest

Tenha certeza que o gtest esteja incluido durante os testes.

```
@include "gtest/gtest"
```

-DDMALLOC -DDMALLOC_FUNC_CHECK

Compilando e Executando

Como compilar `gcc -o main main.c reader.c printer.c -Wall -std=c99`:

```
~dir/sb-jvm-1-2019 user$ gcc -o  -DDMALLOC main main.c reader.c printer.c
-Wall -std=c99
```

Como executar `./main`:

```
~dir/sb-jvm-1-2019 user$ ./main
```

Entre no diretório `../` e execute o seguinte comando para compilar e executar:

```
~dir/sb-jvm-1-2019 user$ make run
```

Para apenas compilar

```
~dir/sb-jvm-1-2019 user$ make
```

Testes

Todos os testes foram feitos com comparação binaria, não fatal (não interrompe o teste após detecção do erro), fornecida pelo googletest como pode ser observado no arquivo de testes teste_file

Fatal assertion	Nonfatal assertion	Verifies
`ASSERT_EQ(val1, val2);`	`EXPECT_EQ(val1, val2);`	`val1 == val2`

Os testes foram realizados em etapas previamente determidadas:

- teste de conversão de caractere - função converte()
- teste de conversão de strings - função avalia()
- teste de unidade na avaliação/conversao romano-arabico
- teste de dezena na avaliação/conversao romano-arabico
- teste de centena na avaliação/conversao romano-arabico
- teste de milhar na avaliação/conversao romano-arabico
- teste de avaliação/conversao geral romano-arabico
- teste de tratamento de erros na avaliação-conversao romano-arabico

O que se esperava do teste em todas as estapas era uma saída de valor correto em algarismos arabicos dada determinada entrada em algarismo romanos. Em caso de entradas inválidas a função deveria retornar -1.

Testes de Coding Style

Durante todo o projeto, após cada aprovaçao nos teste, foram executados os comandos

```
cppcheck --enable=warning <file>
cpplint <file>
```

E em seguida feitas as devidas alterações para enquadrar o código nos padrões delimitados se nescessário

Built With

- Google Test - Test framework
- CPPLint - Coding Style Guide
- CPPCheck - Static Code Analysis Tool
- VSCode - Code Editor

License

This project is licensed under the MIT License - see the LICENSE file for details