* **Introdução**

O objetivo deste trabalho é implementar o jogo *Super Trunfo* em C++, com modo de jogo de player vs player, e 4 baralhos diferentes a serem escolhidos, por meio da criação de um TAD e da manipulação de vetores e stacks. Para a produção e compilação do programa foi usado Linux UbuntuLTS 18.04 e Linux Mint, C++ 11/14.

* **Implementação**

Classes:

class Cartas: Classe “mãe”

class Carro: Classe filha de Cartas

class Aviao: Classe filha de Cartas

class Dinossauro: Classe filha de Cartas

class Heroi: Classe filha de Cartas

Métodos principais:

void imprime\_carta override: método universal (moldado para cada classe/polimorfismo) que retorna os dados da carta.

void embaralhar\_cartas: metodo usado para embaralhar as stacks(pilhas) de cartas.

Void inicializar\_pilhas: usado para inicializar os deck de cartas desejado.

void jogar: que contém comandos para poder rodar o jogo a partir da chamada de um try/catch no main.

* **Testes**

Neste trabalho, apliquei testes unitários baseado no framework Doctest. ( <https://github.com/onqtam/doctest>).

Como nosso projeto é um jogo, a maioria das funções são para executar o jogo, logo a área de cobertura de possíveis testes foi reduzida devido a complexidade das funções que rodam o game.

* **Exceções**

Como nosso projeto é um jogo, tivemos limitados espaços para criar as exceções, visto que o código está todo escrito em forma de programação defensiva para todas as entradas do usuário, ou seja, se tratando de um jogo as exceções foram criadas de forma que os atributos digitados pelo usuário na hora da execução do jogo não assumisse valores inválidos, para isso usamos tratamentos de exceção com try e catch, e usamos também o switch para determinadas entradas dos usuários.

* **Anexos**

Listagem de arquivos:

* bin/main
* build/Aviao.o
* build/Carro.o
* build/Dinossauro.o
* build/Heroi.o
* include/Aviao.h
* include/Carro.h
* include/Cartas.h
* include/Dinossauro.h
* include/Heroi.h
* program/main.cpp
* program/tester.cpp
* src/Aviao.cpp
* src/Carro.cpp
* src/Cartas.cpp
* src/Dinossauro.cpp
* src/Heroi.cpp
* tests/test\_null.cpp
* third\_party/doctest.h
* Makefile
* README.md
* **Como compilar**

O programa é compilado a partir do Makefile com a seguinte sequência de comandos:

make clean

make

./bin/main

* **Considerações finais**

A produção do trabalho foi executada com sucesso, com percalços relacionados a alguns tópicos da disciplina, como programação por contrato, tratamento de exceções , manipulação de vetores, herança, polimorfismo entre outros. A vista disso, o trabalho foi proveitoso no que tange à prática de programação no contexto de PDS e aplicamos várias tecnicas que nos foi passada em sala de aula, no que se diz respeito tanto a programação quanto ao trabalho em equipe, como por exemplo programação em dupla revisão de codigo e etc.