IMD0030

Linguagem de Programação I

Projeto 3

Loja Pet Fera

Discente: David Cardoso

Matrícula: 2016097264

Dezembro/2016, Natal/RN

1. Introdução

1.1. O propósito deste documento é relatar sobre a **Implementação e Customização** do **Programa Pet Fera**, que tem como principal funcionalidade o gerenciamento de cadastros de animais e funcionários da loja.

2. Detalhes de Implementação

2.1. O programa foi implementado de forma modular e é composto por arquivos fonte organizados da seguinte maneira:

2.1.1. ***Source*** (src): **main.cpp, auxiliar.cpp** e os arquivos relacionados à implementação **(.cpp)** dos métodos das **classes de pessoas, animais e suas respectivas classes derivadas**;

2.1.2. ***Include***:

2.1.2.1. Arquivo **auxiliar.h** que contém funções auxiliares do programa principal;

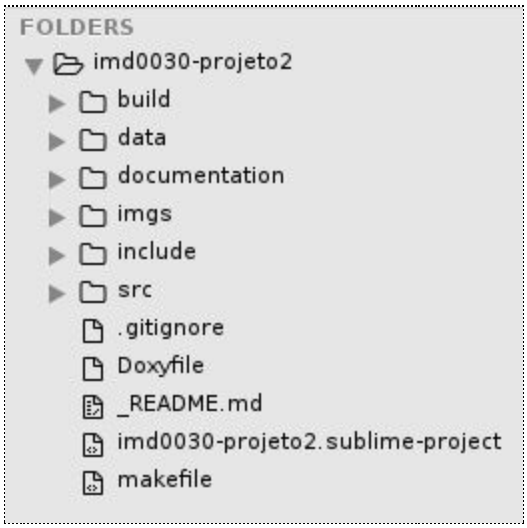
2.1.2.2. Arquivos de cabeçalho **(.h)** relacionados à declaração dos atributos e métodos das **classes de pessoas, animais e suas respectivas classes derivadas**;

2.1.3. ***Build***: arquivos objeto **(.o)** e executáveis **(.exe);**

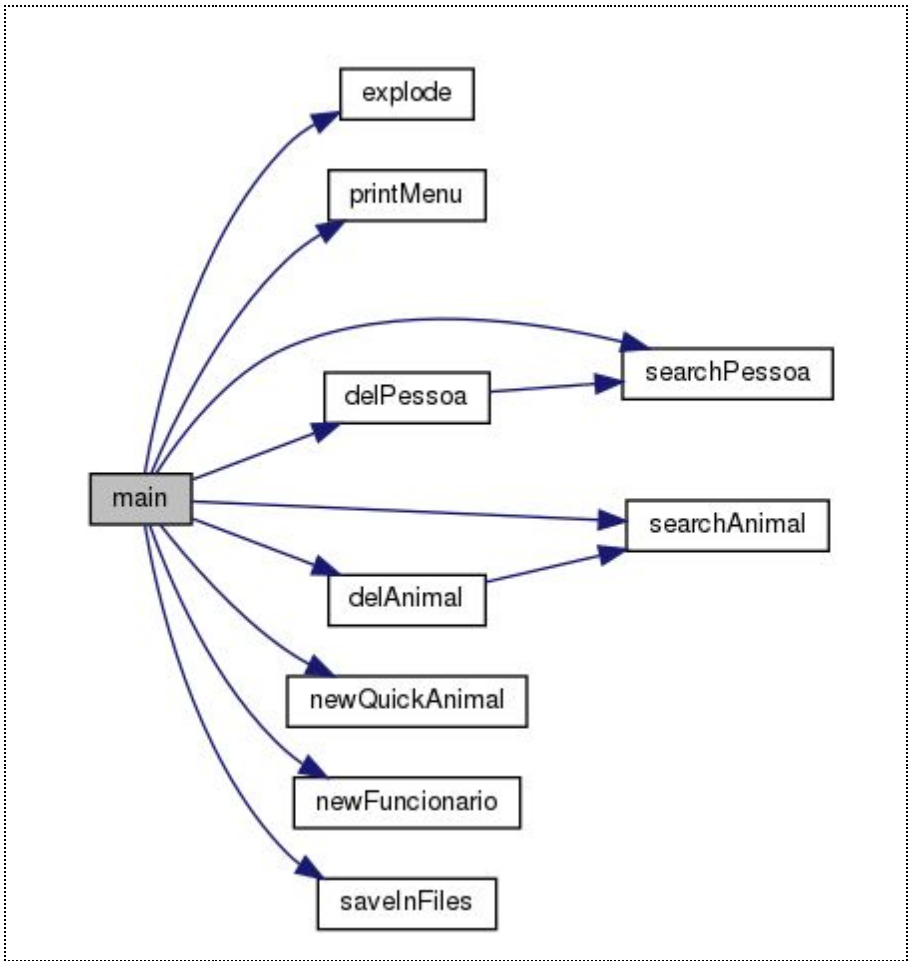
2.1.4. ***Library*** (lib): arquivos binários de bibliotecas. Neste projeto está sendo utilizado biblioteca dinâmica;

2.1.5. ***Data***: arquivos de entrada e saída **(padrão CSV)** utilizados para ler e armazenar os dados dos objetos Pessoa e Animal;

2.1.6. **Arquivos secundários** foram criados para uma melhor organização e execução do projeto, tais como: **README.md**, **makefile**, **Doxyfile** e .**gitignore**;



2.2. No arquivo principal, **main.cpp**, há o código central do programa, que é responsável pelo recebimento dos argumentos via linha de comando, inicialização de variáveis e **preenchimento dos dicionários (map) de Pessoas e Animais** a partir dos arquivos de entrada. Além disso, o **main.cpp** gerencia a interface principal de interação com o usuário, permitindo ações como busca e cadastro de animais e funcionários; O arquivo **auxiliar.h** dá suporte à essa interação através de diversas funções;



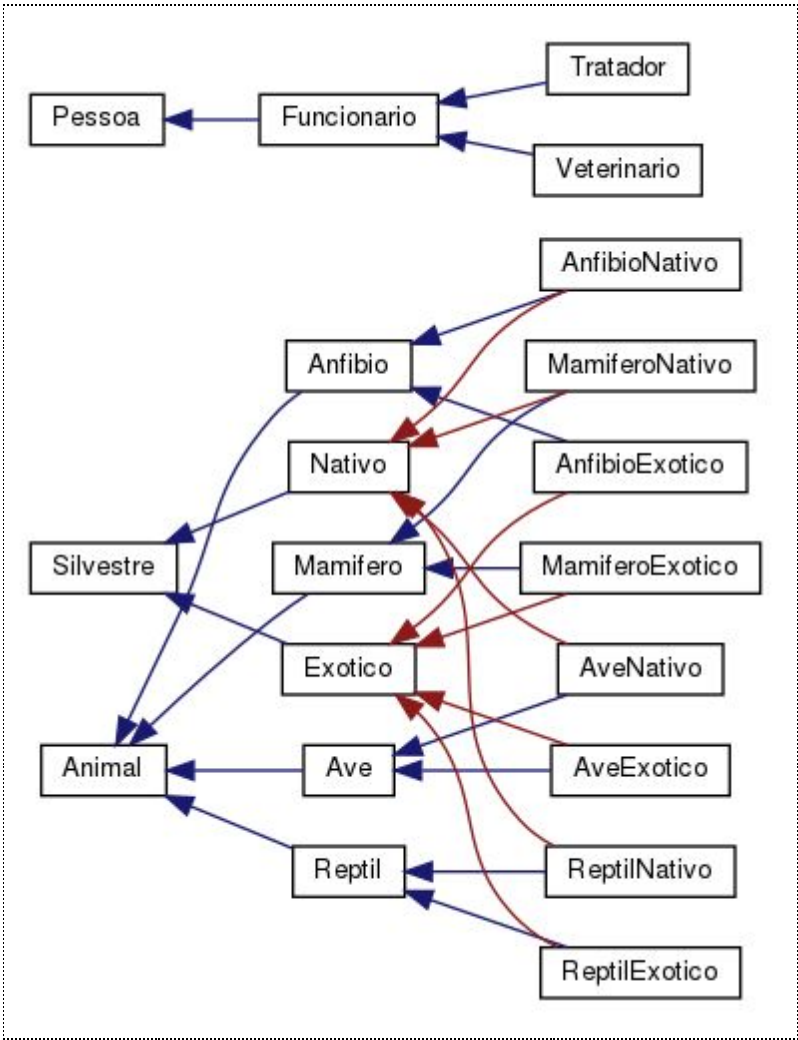
2.3. Pensando em deixar o programa mais genérico, optou-se por criar a **classe Pessoa** para ser base das classes derivadas **Funcionário**, **Veterinário** e **Tratador**; Pois, dessa forma, é mais fácil evoluir o programa para tratar outros tipos de pessoas como, por exemplo, Cliente;

2.4. As classes relacionadas à animais foram divididas da seguinte forma:

2.4.1. **Animal Silvestre** como base para **Animal Exótico** e **Animal Nativo**;

2.4.2. **Animal** como base para as classes de animais, atualmente suportadas, **Anfíbio**, **Ave**, **Mamífero** e **Réptil**;

2.4.3. O dicionário (map) de animal é preenchido com objetos com **herança múltipla** que combina um tipo de classe de animal com um tipo de animal silvestre como, por exemplo, **Anfíbio Exótico** e **Anfíbio Nativo, Mamífero Exótico e Mamífero Nativo, etc.**;



2.5. Além dos arquivos fonte, é importante citar a finalidade de cada arquivo secundário do projeto:

2.5.1. **README.md** contém informações gerais sobre o projeto;

2.5.2. **makefile** contém instruções para automatizar a compilação dos arquivos objeto **(.o)** e do executável **petfera.exe**, assim como, comandos extras para teste com valgrind e outras ações auxiliares;

2.5.3. **Doxyfile** contém as tags de configuração do programa Doxygen - responsável pela geração da documentação do programa;

2.5.4. **.gitignore** contém os arquivos e as extensões que devem ser ignorados pelo comando “git add” e, assim, não são adicionados ao controle de versão do código fonte do programa.

3. Informações adicionais

3.1. Link deste relatório: https://goo.gl/7DPME2

3.1.1. Observação: este documento está liberado para comentários online.

3.2. Documentação Doxygen:

3.2.1. file:///dir/p/projeto/imd0030-projeto3/documentation/doc/index.html