Universidade Federal do Rio Grande do Norte

Instituto Metrópole Digital - IMD
Bacharelado em Tecnologia da Informação – BTI
09 de Dezembro de 2017

Relatório Resumo do PetFera

Componentes: Gleydvan Macedo e João Vítor Venceslau Coelho

Professor: Silvio Costa Sampaio

O que foi feito

Foram criadas as classes para representar um total de 12 possíveis tipos de animais, sendo que cada uma das Classes Anfíbio, Mamífero, Reptil e Aves, possuem duas subclasses que representam suas versões nativas e exóticas, e todas derivam da classe Animal.

Também foi criada a Classe Funcionário, que possui como subclasses Veterinário e Tratador. Além da Classe Pet_Fera_Cadastro, que realiza todas as operações solicitadas, além de um programa auxiliar para filtrar animais de acordo com a classe, veterinário e/ou tratador associado.

Foram criados dois CSVs para armazenar os Animais e os Funcionários cadastrados conforme solicitado.

Dificuldades encontradas

A principal dificuldade encontrada foi durante os processos de cadastro e leitura dos arquivos CSVs, pois não conseguimos reduzir a quantidade de código repetido durante o cadastro por meio de uma função para os campos em comum que os diversos animais compartilham, a solução utilizada foi deixar a massiva repetição de código sem simplificação, além desse problema, tivemos certa dificuldade para utilizar os recurso do C++11, atrasando bastante o progresso na implementação dos métodos idealizados, sendo necessário buscar soluções alternativas e menos eficientes. Por fim acreditamos ter encontrado o motivo da falta dos recursos do C++11 devido a uma pequena troca nas opções de compilação, onde estava sendo utilizada a seguinte sequencia:

```
g++ -Wall -pedantic -std=c++11 -ansi
```

Com o -ansi localizado após o -std=c++11, após termos compilado cada objeto .o individualmente e nenhum erro ter sido encontrado. Decidimos refazer completamente o makefile, consertando o problema. Por fim as opções de compilação ficaram nessa ordem:

```
g++ -Wall -pedantic -ansi -std=c++11
```

O que não foi feito

Devido aos atrasos encontrados, não tivemos tempo de implementar todas as devidas exceções que podem ocorrer, assim com realizar os devidos testes de situações onde essas exceções podem aparecer, criamos apenas algumas exceções básicas para a leitura de arquivos.

Considerações Finais

Acreditamos que mesmo o programa estando funcional,ainda está muito “bruto”, podendo ser grandemente melhorado, e muitos trechos simplificados por meio de funções, evitando a repetição do código, além de que poderíamos ter explorado melhor as exceções do C++.

Por fim, gostaríamos de ter organizado melhor o código, porém conciliar os estilos de programação dos dois integrantes foi um processo difícil, assim não foi utilizado um padrão para o código.