Universidade Federal do Pará
Disciplina: Programação I
Profa: Paula Cardoso
Aluno: Fábio Barbosa 202004940021

Relatório: Sistema para Cadastro de Animais

Belém PA-2024

1. Introdução

A princípio, o projeto possui vários requisitos a serem cumpridos, a introdução expõe a forma em esses requisitos foram interpretados, pois sabe-se que para construir um produto sempre devemos iniciar do planejamento. Como o projeto foca, principalmente, em manipulação de dados, foi necessário a construção de uma base de dados inicial, de forma que essa base deve estar presente em um documento *csv*. Nesse contexto, dentro de suas requisições, também está a possibilidade de exportar o arquivo, se estiver em *csv*, para binário, se estiver em binário, para *csv*.

Ademais, usando estruturas de agrupamento presentes na linguagem de programação *python*, como listas e bibliotecas, foi carregado cada elemento presente na base de dados na memória. O tipo de configuração estrutural feito foi uma lista, na qual cada elemento é uma biblioteca, e cada biblioteca possui seis chaves e seis valores. Nesse sentido, cada biblioteca presente na lista, é uma linha do meu arquivo *csv*, e cada linha também é um cadastro do meu sistema.

Feito todo o processo de carregamento dos dados, os requisitos adiante abordam a manipulação dos mesmos e da estrutura de conjuntos, na qual foram carregados. Esse controle, abrange a inclusão de novos elementos na base de dados, a exclusão de um cadastro existente, a busca por posições onde encontra-se uma determinada chave, a exibição, de várias formas diferentes, dos elementos, e qualquer outra forma de manipulação que for jugada válida. Se faz necessário o uso de estruturas de repetição para percorrer entre os elementos, que estão presentes no conjunto estrutural.

2. Descrição Prática

Primeiramente, utilizei 4 módulos do python, que são: *sys* para executar a função principal e sair da aplicação; *pickle* para transformar o conteúdo do arquivo “base_de_dados.csv” em binário; *time* para pausar por 3 segundos quando um usuário sair da aplicação, isso apresenta um efeito

interessante; e *os* para modificar a extensão da “base_de_dados.csv” para “.bin”.

- Função **getDataBase(caminho do arquivo)**: é usada para retornar a estrutura principal, onde são carregados os dados, é a primeira função utilizada na função principal “main()”.
- Função **exportaArquivo(caminho do arquivo)**: Implementada com o intuito de mudar o tipo de um arquivo e retornar o novo arquivo, com as opções de *csv* ou *bin*, ela também usa a função “mudar_extensão(caminho do arquivo, nova extensão)”.
- Função **gerarElemento()**: solicita os dados de um elemento (nome, tipo, idade, descrição, data e hora, e preço), e retorna uma biblioteca (registro ou elemento) com as informações passadas, foi usada para incluir um novo elemento.
- Função **getPosicaoKey(chave, estrutura com os elementos)**: essa função imprimi todas as posições, dentro da estrutura que guarda os elementos, que possuem a chave passada por parâmetro.
- Função **imprimir(estrutura de dados, posição inicial, posição final)**: nessa função, os parâmetros de posição inicial e posição final possuem valores padrões, no caso de não ser incluso nenhum dos dois, a função irá imprimir todos os registros, no caso de só a posição inicial ser inclusa, ela irá imprimir apenas o elemento da posição inicial, em último caso, no qual os dois são inclusos, ela irá imprimir todos os elementos no intervalo [posição inicial – posição final].

Todas as estruturas acima conversam dentro da função “main()”, na qual, foi criado um menu com as seguintes opções: 0- Exportar para csv ou binário; 1- Inserir; 2- Buscar; 3- Imprimir; 4- Remover; 5- Sair. Nesse contexto, as únicas opções que não têm suporte de uma função é a de remover e a de sair. A de sair, simplesmente interrompe a aplicação pois existe uma estrutura de repetição “while”, que repete o laço enquanto a variável sentinela “op” for diferente de 5. Já a opção de remover, inclui a

posição do elemento que será removido em uma lista, quando for inserido um novo elemento, o elemento na posição 0 dessa lista é removido da estrutura principal.

3. Conclusão

O projeto possui vários pontos a se melhorar, iniciando pelo fato de que ele não grava dentro do arquivo as novas modificações, além disso, não há um tratamento de erros tão eficiente, sendo suscetível, facilmente, a bugs. Entretanto, para o objetivo apresentado nos requisitos, ele alcançou bons resultados, fazendo a maioria dos requisitos corretamente, caso os inputs também estejam corretos. Cada opção do menu principal apresentado no console quando a aplicação é executada, funciona de maneira técnica.