Lista Avaliativa 2

Este trabalho consiste no desenvolvimento de um programa utilizando a linguagem C e deve ser realizado em dupla ou individualmente. O código-fonte (arquivo .c) deve ser enviado pelo portal didático. Basta um dos alunos da dupla postar o arquivo. Favor, incluir no início do arquivo um comentário com o(s) nome(s) do(s) autor(es) para que eu possa identificar quem são.

O programa consiste em um sistema de agendamento de consultas veterinárias. O sistema armazena informações sobre os animais, os donos, os veterinários e as consultas. Cada questão abaixo define um registro ou função do sistema.

Obrigatoriamente, todas as strings devem ser ponteiros para char e alocadas dinamicamente;

É permitido utilizar todos os recursos vistos na disciplina e outros que desejar, exceto se esses recursos já entregarem uma solução pronta para o trabalho todo ou parte dele. Por exemplo, você pode usar a biblioteca string.h, pois não foi solicitado no trabalho que você implemente soluções para manipulação de string, mas você não pode usar uma biblioteca de lista com alocação dinâmica porque você tem que criar uma no trabalho;

Trabalhos considerados semelhantes (por exemplo, mas não somente, que se diferenciam apenas pelos nomes das variáveis e textos de printf) acarretam a punições previstas na UFSJ. Isso também é aplicado caso dois ou mais alunos apresentem programas com trechos iguais copiados da mesma fonte na Internet.

Favor, indicar por meio de comentários no código onde começa o código referente a cada questão.

Questão 01 [1,0]

Crie o registro Dono, que possui os atributos: nome (string); e telefone (string).

Crie o registro Animal, que possui os atributos: nome (string); idade (int); peso (float); e dono (ponteiro para Dono).

Crie o registro Veterinario, que possui os atributos: nome (string); e CFMV (registro no conselho de medicina veterinária) (int).

Crie o registro Consulta, que possui os atributos: data (string); horário (int); animal (ponteiro para Animal); e veterinário (ponteiro para Veterinario).

Questão 02 [1,0]

Crie a função imprimirDono, que não tem retorno e recebe um Dono por referência e imprime os seus dados.

Crie a função imprimirVeterinario, que não tem retorno e recebe um Veterinario por referência e imprime os seus dados.

Crie a função imprimirAnimal, que não tem retorno e recebe um Animal por referência e imprime os seus dados. Imprima apenas o nome do dono para o campo dono.

Questão 03 [1,0]

Crie a função buscarDono, que recebe o array de donos, a quantidade de donos cadastrados (int) e o nome do dono que se deseja encontrar (string). Esta função procura no array o dono que possui o nome passado e retorna o índice (int) dele no array. Caso não encontre, a função retorna -1.

Faça também as funções buscarAnimal e buscarVeterinario seguindo a mesma lógica.

Questão 04 [1,0]

Crie a função cadastrarDono, que não tem retorno e recebe o array de donos e a quantidade de donos cadastrados por referência (int*). Esta função verifica se o array está cheio e, caso afirmativo, avisa ao usuário que não pode cadastrar e encerra. Caso exista espaço no array, a função solicita os dados do novo Dono ao usuário, armazena no array recebido e atualiza a quantidade de donos cadastrados.

Faça também a função cadastrarVeterinario seguindo a mesma lógica.

Questão 05 [1,0]

Crie a função cadastrarAnimal, que não tem retorno e recebe um array de animais, a quantidade de animais cadastrados por referência (int*), o array de donos e a quantidade de donos cadastrados (int). A função verifica se o array de animais está cheio e, caso afirmativo, a função avisa que não pode cadastrar e encerra. Caso exista espaço no array, a função solicita o nome do dono do animal ao usuário e busca os dados desse dono. Caso não encontre o dono, a função avisa ao usuário e pergunta se deseja buscar o dono novamente ou desistir de cadastrar o animal, encerrando a função. Isso se repete até o usuário encontrar o dono ou desistir e encerrar a função. Caso tenha sucesso em encontrar o dono, a função deve solicitar os demais dados do animal, armazenar tudo no array de animais e atualizar a quantidade de animais cadastrados.

Questão 06 [1,0]

Crie a função recursiva horarioDisponivel, que recebe o array de consultas, a quantidade de consultas cadastradas (int), uma data (string) e um horário (int). Esta função verifica se já existe uma consulta marcada na data e horário passados e retorna 1, se não existir (então o horário está disponível), e 0, caso já exista.

Questão 07 [1,0]

Crie a função agendarConsulta, que não tem retorno e recebe o array de consultas, a quantidade de consultas cadastradas por referência (int*), os arrays de animais e veterinários e suas quantidades (int). A função verifica se o array de consultas está cheio e, caso afirmativo, avisa que não pode cadastrar e encerra. Caso exista espaço no array, a função solicita o nome do veterinário ao usuário e busca os dados dele. Caso não encontre o veterinário, a função avisa ao usuário e pergunta se deseja buscar o dono novamente ou desistir de agendar a consulta, encerrando a função. Isso se repete até o usuário encontrar o veterinário ou desistir e encerrar a função. Caso tenha sucesso em encontrar o veterinário, a função deve solicitar o nome do animal, repetindo os mesmos passos relativos ao veterinário. Em seguida, a função solicita a data e o horário da consulta e verifica se há disponibilidade. Caso não haja, a função avisa ao usuário e pergunta se deseja indicar outra data e horário ou desistir de agendar a consulta, encerrando a função. Isso se repete até o usuário encontrar o veterinário ou desistir e encerrar a função. Caso tenha sido encontrado um horário disponível, a função deve armazenar todos os dados no array de consultas e atualizar a quantidade de consultas cadastradas.

Questão 08 [2,0]

Crie a função visualizarAgenda, que não tem retorno e recebe o array de consultas e a quantidade de consultas cadastradas (int). Esta função solicita uma data (string) e imprime a data, hora e os nomes do veterinário, animal e dono das consultas naquela data, conforme o exemplo mostrado abaixo em que aparecem 2 consultas.

Questão 09 [1,0]

Implemente a função main, em que são declarados os arrays de donos, animais, veterinários e consultas. A função apresenta um menu que possui as opções (a ordem não importa): sair, cadastrar dono, cadastrar animal, cadastrar veterinário, buscar dono, buscar animal, buscar veterinário, agendar consulta e visualizar consultas do dia. Ao selecionar uma opção, o programa deve utilizar a função correspondente, solicitando/passando todas as informações necessárias para executá-las e voltar para o menu. No caso das funções de busca, o programa solicita o nome ao usuário, busca e imprime todos os dados do dono, animal ou veterinário ou informa que não encontrou. Após a impressão resultante das buscas e da agenda de consultas, o programa deve pausar. Para isso, use a função system(“PAUSE”); da biblioteca stdlib.h.