Universidade Federal do Rio Grande do Sul
Instituto de Informática
INF01202 - Algoritmos e Programação

Trabalho Final

Prof. Vinícius Garcia Pinto

2019-1

Especificação do Trabalho Prático

O trabalho final extra-classe tem por objetivo agregar os conteúdos tratados na disciplina INF01202. Ele é requisito parcial para aprovação.

Descrição do Problema

Você foi contratado pela LocAU! (LOCadora de AUtomóveis Ltda.) para desenvolver um sistema gerenciador de locações de automóveis. O sistema deve ser desenvolvido em linguagem C e será utilizado apenas no computador da locadora. Todas as inserções de dados por parte do usuário devem ser validadas de forma a evitar uma situação inconsistente nos arquivos de dados.

Fonte: `https://pxhere.com/en/photo/1349397`

Dados de Entrada

Arquivo Texto `clientes.csv`

O arquivo `clientes.csv`, é um arquivo texto que contém os dados dos clientes da locadora onde a primeira linha contém o nome das colunas (cabeçalho) e as linhas seguintes contêm as informações sobre cada cliente.

- Exemplo:

```
codigo_cliente,nome,cnh,ddd,telefone
1,Daniel Osvaldo Araújo,11371110700,55,981928571
2,Isabela Sophia Novaes,14096054168,51,982024781
3,Diego Nicolas Rezende,82788772250,11,997588232
4,Emilly Raquel Porto,48354971870,48,992102813
```

Arquivo Texto `carros.csv`

O arquivo `carros.csv`, é um arquivo texto que contém os dados dos carros da locadora onde a primeira linha contém o nome das colunas e as linhas seguintes contêm as informações sobre cada carro.

- Exemplo:

```
codigo_carro,marca,modelo,ano,placa,valor_diaria,valor_seguro,quantidade
1,Ford,Focus 2.0 Aut.,2018,HJO4752,80,30,5
2,Fiat,Strada Celeb. 1.4,2018,GVP9221,140,40,3
3,Fiat,Uno 1.4,2017,HKZ4848,60,30,10
4,BMW,X1 2.0 Aut.,2019,HAR8260,280,60,1
```

Arquivo Binário `alugueis.bin`

O arquivo `alugueis.bin`, é um arquivo binário que contém os dados de todos aluguéis realizados onde cada linha contêm as informações sobre cada aluguel (Esse arquivo não possui cabeçalho!).

- Exemplo (Após leitura do binário):

```
1,2,4,1,5,2019,1,340,S,E
2,4,3,2,5,2019,3,180,N,E
3,3,2,4,5,2019,2,280,N,E
4,1,1,6,5,2019,5,550,S,L
5,2,4,8,5,2019,2,680,S,L
```

Estruturas e Bibliotecas

Devem ser definidas, pelo menos, as seguintes estruturas (`struct`) para armazenar as informações contidas nos arquivos:

- `cliente` possui cliente possui um `código` (número inteiro), um `nome` (*string*), uma `CNH` (*string*), um DDD (*string*) e um telefone (*string*).
- `carro` possui um `código` (número inteiro), uma `marca` (*string*), um `modelo` (*string*), um `ano` (número inteiro), uma `placa` (*string*), um `valor de diária` (número real), um `valor de seguro` (número real) e uma `quantidade` (número inteiro).
- `aluguel` possui um `código` (número inteiro), um `código do cliente` (número inteiro), um `código do carro` (número inteiro), um `dia` (número inteiro), um `mês` (número inteiro), um `ano` (número inteiro), um `número de diárias` (número inteiro), um `valor` (número real), um `seguro` (char): `'S'` ou `'N'` e uma `situação` (char): `'L'` (locado) ou `'E'` (entregue).

Cada uma das estruturas deve ser implementada dentro de uma biblioteca (ex.: cliente.h). As funções para manipulação dos dados, leituras, escritas, inclusões, relatórios, entre outras, devem ser implementadas no respectivo arquivo de implementação da biblioteca (ex.: cliente.c).

Vetores

Utilize vetores de estruturas para armazenar os dados lidos dos arquivos durante a execução do programa. Defina um tamanho MAX (Ex.: 100) para os vetores e mantenha um contador MAX_ATUAL para cada vetor estrutura sendo assim possível saber quantos clientes, carros e empréstimos existem no momento. O programa deve ler os arquivos, armazenar nos vetores e após mostrar o menu para o usuário. Quando o usuário decidir sair o programa deve salvar todos clientes, carros e alugueis nos respectivos arquivos e finalizar.

Funções

Implemente, no mínimo, as seguintes funções:

Leituras:

- `leitura_clientes` que leia o arquivo `clientes.csv` e preencha o vetor de clientes.
- `leitura_carros` que leia o arquivo `carros.csv` e preencha o vetor de carros.
- `leitura_alugueis` que leia o arquivo binário `alugueis.bin` e preencha o vetor de alugueis.

Informação:

- `info_cliente` que receba o código de um cliente e o vetor contendo todos os clientes, realize a busca do cliente e imprima na tela todas as informações sobre o mesmo.
- `info_carro` que receba o código de um carro e o vetor contendo todos os carros, realize a busca do carro e imprima na tela todas as informações sobre o mesmo.
- `info_aluguel` que receba o código de um aluguel e o vetor contendo todos os alugueis, realize a busca do aluguel e imprima na tela todas as informações sobre o mesmo.

Inclusão

- `novo_cliente` que receba o vetor contendo todos os clientes, solicita ao usuário as informações do cliente (o código do cliente deve ser preenchido automaticamente com um valor sequencial) e armazena no vetor de clientes.
- `novo_carro` que receba o vetor contendo todos os carros, solicita ao usuário as informações do carro (o código do carro deve ser preenchido automaticamente com um valor sequencial) e armazena no vetor de carros.
- `novo_aluguel` que receba o vetor contendo todos os alugueis, solicita ao usuário as informações do aluguel (o código do aluguel deve ser preenchido automaticamente com um valor sequencial e a situação com o caractere `'L'`) e armazena no vetor de alugueis. Lembre de só alugar um carro que esteja disponível no momento.

Atualização

- `devolucaoo` que receba o código do aluguel e o vetor contendo todos os alugueis, realize a busca do aluguel e modifique o campo situação para `'E'`.

Relatórios

- `lista_carros` que receba a marca e o vetor contendo todos os carros e exiba a lista de carros que correspondem a marca informada.
- `lista_carros_preco` que receba a marca e o vetor contendo todos os carros e exiba a lista de carros que correspondem a marca informada ordenada por preço decrescente.
- `lista_clientes` que receba o vetor contendo todos os clientes e exiba a lista de clientes em ordem alfabética do primeiro nome, não é necessário considerar nomes compostos.
- `lista_alugueis` que receba o código de um cliente e o vetor contendo todos os alugueis e exiba a lista de alugueis deste cliente.
- `lista_atrasados` que receba o vetor contendo todos os alugueis e o vetor contendo todos os clientes e exiba as informações de todos os clientes (uma vez por cliente) que possuem carros não entregues.

Gravação

- `grava_clientes` que receba o vetor contendo todos os clientes e grave no arquivo clientes.csv.
- `grava_carros` que receba o vetor contendo todos os carros e grave no arquivo carros.csv.
- `grava_alugueis` que receba o vetor contendo todos os alugueis e grave no arquivo binário alugueis.bin.

Funcionalidades Extra

Cada aluno pode implementar livremente funcionalidade adicionais no seu trabalho. A única exigência é que as funcionalidades adicionais **não façam** uso de recursos específicos de um outro sistema operacional. Procure utilizar bibliotecas padrão da linguagem C ou que tenham implementação para vários sistemas operacionais. Em caso de dúvida, consulte o professor.

Exemplos de funcionalidades adicionais

- interface gráfica
- categorias para os carros (entrada, luxo, suv, picape, furgão, van, etc)
- promoções (sorteio de diária extra ou *upgrade*)
- custos adicionais (cobrança por quilometragem extra ou por devolução sem tanque cheio)
- programa de fidelidade (registra o total de locações de cada usuário e concede algum benefício a cada n locações
- remoção de cliente/carro
- concessão de descontos para diárias de longa duração (pode ser escalonado, exemplo: 15 dias $\rightarrow$ 10%, 30 dias $\rightarrow$ 20%, etc)

Avaliação do Trabalho

A avaliação levará em conta os seguintes critérios:

- **corretude** do programa, ou seja, o programa mostra o resultado correto para uma dada entrada;
- **correspondência** ao **enunciado**, o programa atende a todos os requisitos presentes no enunciado da questão;
- **validação das entradas** quando solicitado no enunciado. Exemplo: programa só deve aceitar valores maiores que 1;
- programas que **não compilarem** receberão **nota 0**. Caso seja necessário enviar uma resposta parcial ou incompleta, comente (preferível) ou retire eventuais linhas de código que estejam impedindo a compilação;
- **identação** do código;
- uso das estruturas e funcionalidades vistas em aula;
- **modularização** e organização do código;
- **usabilidade**, por exemplo: informações claras ao usuário do que ele deve digitar;

Instruções de envio

- Comentário no cabeçalho de cada arquivo informando o nome completo do aluno e o número do cartão UFRGS.

```
/*
   Universidade Federal do Rio Grande do Sul
   INF01202 - Trabalho Final
   Turma X
   Nome do Aluno: Meu nome completo
   Cartao UFRGS:  00XXXXXX
*/

/* Descrição sobre o problema e o que faz o código. */

#include<stdio.h>

...

int main(){
  ...
}
```

Exemplo de arquivo a ser enviado.

- Enviar na entrada correspondente no Moodle Acadêmico.

Prazo de Entrega

O trabalho é **individual** e possui três etapas obrigatórias:

1) Andamento

- **06/06/2019**
  - O código parcial deve ser enviado no Moodle até 23h55
- **07/06/2019** - apresentação em aula
  - Breve relato (+- 3min) para explicar o andamento do trabalho e mostrar o que já está feito
  - É obrigatório ter alguma parte do código do trabalho funcional e que possibilite uma demonstração

- O não cumprimento desta etapa implicará em desconto de 10% da nota final obtida pelo aluno!

2) Entrega Final

- **27/06/2019** às 23h55

3) Apresentação em Aula (obrigatória)

- **28/06/2019** às 13h30 (turma I)
- **28/06/2019** às 15h30 (turma J)
- O aluno deve ser capaz de explicar todos os recursos da linguagem, comandos e bibliotecas utilizados no seu código. É permitido usar recursos não vistos em aula desde que o aluno saiba explicar como funcionam e para que servem.

Verificação anti-plágio

A detecção de plágio em qualquer atividade implicará penalidades (nota zero) a todos os envolvidos!

- **todos** os materiais entregues (práticas e trabalho) são submetidos a verificação anti-plágio

Dúvidas

Neste trabalho contaremos com o auxílio do doutorando do PPGC Matheus Serpa que está realizando Estágio Docência na INF01202 neste semestre. O Matheus participará da avaliação dos trabalhos e estará disponível para esclarecer dúvidas relacionadas tanto à especificação do trabalho quanto para dúvidas de programação referentes ao desenvolvimento do mesmo. Eventualmente, as dúvidas também podem ser esclarecidas com o professor Vinícius ou com o monitor Gustavo.

Para contatar o Matheus, envie mensagem via e-mail

- E-mail: ***matheus.serpa@inf.ufrgs.br***
- Assunto: TF - INF01202 - Turma ? - 2019/1

ou pessoalmente

- Prédio: 43413 (67)
- Sala: 201