CENTRO PAULA SOUZA GOVERNO DO ESTADO DE SÃO PAULO FATEC CARAPICUÍBA

Professor:	Ciro Cirne Trindade
Disciplina:	Linguagem de Programação
Curso:	Análise e Desenvolvimento de Sistemas
Grupos:	3 alunos
Data de entrega:	04/06/2014

2º Trabalho Prático – Concessionária de Automóveis

Implemente uma aplicação em C para o controle de vendas em uma concessionária de automóveis.

Um automóvel possui as seguintes características: código, marca, modelo, ano, preço e um campo que indica se ela já foi vendido ou não.

Um vendedor possui os seguintes dados: código e nome.

Quando um automóvel é vendido, guarda-se as informações da data da venda, código do automóvel e código do vendedor. O valor da venda é sempre igual ao preço do automóvel.

O sistema deve possuir as seguintes funcionalidades:

a) **Cadastrar automóvel**: o sistema deve gerar o código do automóvel de forma automática e sequencial e solicitar sua marca, modelo, ano e preço. Todo automóvel que é cadastrado a princípio ainda não foi vendido. Os automóveis cadastrados devem ser armazenados num arquivo cujo nome é definido pela constante `ARQ_AUTOMOVEIS` declarada no arquivo *automovel.h* (vide adiante);

b) **Listar todos os automóveis**: o sistema deve gerar uma listagem no formato de tabela contendo o código, marca, modelo e preço de todos os automóveis cadastrados, tenham eles sido vendidos ou não;

c) **Listar os automóveis a venda**: idêntica a listagem anterior, só que aparecem apenas os veículos que não foram vendidos ainda. O usuário deve ter a opção de gerar essa listagem em um arquivo do tipo texto cujo nome é definido pela constante `ARQ_A_VENDA` declarada no arquivo *automovel.h* (vide adiante);

d) **Cadastrar vendedor**: o sistema deve gerar o código do vendedor de forma automática e sequencial e depois solicitar seu nome. Os vendedores cadastrados devem ser armazenados em um arquivo cujo nome é definido pela constante `ARQ_VENDEDORES` declarada no arquivo *vendedor.h* (vide adiante);

e) **Listar vendedores**: o sistema deve gerar uma listagem no formato de tabela de todos os vendedores cadastrados;

f) **Cadastrar venda**: o sistema deve solicitar o código do automóvel vendido, mostrar os dados desse automóvel (marca, modelo, ano e preço), e se ele ainda não foi vendido, solicitar o código do vendedor e mostrar seu nome, e solicitar a data da venda. O sistema deve mostrar mensagens adequadas se o veículo já foi vendido ou quando o código do automóvel ou do vendedor forem inválidos. As vendas devem ser armazenadas em um arquivo cujo nome é definido pela constante `ARQ_VENDAS` declarada no arquivo *venda.h* (vide adiante);

g) **Listar todas as vendas realizadas em um mês/ano**: o sistema deve solicitar um mês e ano e depois gerar uma listagem em formato de tabela contendo a data da venda,

marca, modelo, ano e preço do automóvel e nome do vendedor de todas as vendas neste mês/ano. No final, o sistema deve exibir o valor total das vendas nesse mês/ano;

h) **Listar todos os automóveis vendidos por um vendedor**: o sistema deve solicitar o código do vendedor, mostrar seu nome, e exibir uma listagem no formato de tabela ordenada pela data da venda contendo a data da venda, marca, modelo, ano e preço de todos os automóveis vendidos pelo vendedor. No final, o sistema deve exibir o valor total das vendas do vendedor. Se o código do vendedor for inválido, o sistema deve exibir uma mensagem.

A seguir são fornecidos os arquivos *concessionaria.c*, *automovel.h*, *vendedor.h* e *venda.h*. Estes arquivos devem ser seguidos rigorosamente em não podem ser modificados. O grupo deve fornecer a implementação dos arquivos *automovel.c* (contendo a implementação das funções definidas em *automovel.h*), *vendedor.c* (contendo a implementação das funções definidas em *vendedor.h*) e *venda.c* (contendo a implementação das funções definidas em *venda.h*).

Arquivo ***concessionaria.c***:

```
/* concessionaria.c
 *
 * Implementação de um sistema de controle de vendas de uma
 * concessionária de automóveis.
 *
 * Ciro Cirne Trindade
 * 15/05/2014
 */

#include <stdio.h>
#include "automovel.h"
#include "vendedor.h"
#include "venda.h"

#define CAD_AUTOMOVEL         1
#define LIST_TODOS_AUTOS      2
#define LIST_AUTOS_A_VENDA    3
#define CAD_VENDEDOR          4
#define LIST_VENDEDOR         5
#define CAD_VENDA             6
#define LIST_VENDAS_MES       7
#define LIST_VENDAS_VENDEDOR  8
#define SAIR                  0

/* exibe o menu de opções do sistema e devolve a escolha do usuário */
int menu(char * [], int);

int main() {
      char * opcoes[] = { "Cadastrar automóvel",
                          "Listar todos os automóveis",
                          "Listar os automóveis a venda",
                          "Cadastrar vendedor",
                          "Listar vendedores",
                          "Cadastrar venda",
                          "Listar todas as vendas de um mês/ano",
                          "Listar todas as vendas de um vendedor",
```

```
                        "Sair do programa" };
      int op;

      do {
            op = menu(opcoes, sizeof(opcoes) / sizeof(char *));
            switch (op) {
                  case CAD_AUTOMOVEL:
                        cadastrar_automovel();
                        break;
                  case LIST_TODOS_AUTOS:
                        listar_todos_automoveis();
                        break;
                  case LIST_AUTOS_NAO_VEND:
                        listar_automoveis_a_venda();
                        break;
                  case CAD_VENDEDOR:
                        cadastrar_vendedor();
                        break;
                  case LIST_VENDEDOR:
                        listar_vendedores();
                        break;
                  case CAD_VENDA:
                        cadastrar_venda();
                        break;
                  case LIST_VENDAS_MES:
                        listar_vendas_mes();
                        break;
                  case LIST_VENDAS_VENDEDOR:
                        listar_vendas_vendedor();
                        break;
                  case SAIR:
                        break;
                  default:
                        printf("\n\tOpção inválida!\n");
            } // fim do switch
      } while (op != SAIR);
      return 0;
}

int menu(char * opcoes[], int num) {
      int i, op;

      printf("\n\n\t\tCONCESSIONÁRIA DE AUTOMÓVEIS\n\n");
      for (i = 0; i < num-1; i++) {
            printf("\t%2d - %s\n", i + 1, opcoes[i]);
      }
      printf("\t%2d - %s\n", SAIR, opcoes[i]);
      printf("\tOpção: ");
      scanf("%d", &op);
      return op;
}
```

Arquivo ***automovel.h***:

```
/* automovel.h
```

```
 *
 * Define o tipo automovel e os protótipos das operações realizadas
 * sobre esse tipo.
 *
 * Ciro Cirne Trindade
 * 15/05/2014
 */

#ifndef _AUTOMOVEL_H
#define _AUTOMOVEL_H

#include <stdbool.h>

#define ARQ_AUTOMOVEIS  "automoveis.dat"
#define ARQ_A_VENDA     "avenda.txt"

typedef struct {
      int codigo; // código do automóvel
      char marca[21]; // marca do automóvel, por exemplo, Fiat, Ford, VW
      char modelo[21]; // modelo do automóvel, por exemplo, Palio,
Fiesta, Gol
      int ano; // ano de fabricação do automóvel
      float preco; // preço de venda do automóvel
      bool vendido; // indica se o automóvel foi vendido ou não
} automovel;

/* função que cadastra um automóvel no arquivo */
void cadastrar_automovel();

/* função que lista todos os automóveis cadastrados */
void listar_todos_automoveis();

/* função que lista os automóveis a venda */
void listar_automoveis_a_venda();

#endif
```

Arquivo ***vendedor.h***:

```
/* vendedor.h
 *
 * Define o tipo vendedor o os protótipos das operações realizadas
 * sobre esse tipo.
 *
 * Ciro Cirne Trindade
 * 15/05/2014
 */

#ifndef _VENDEDOR_H
#define _VENDEDOR_H

#define ARQ_VENDEDORES "vendedores.dat"

typedef struct {
      int codigo; // código do vendedor
```

```
	char nome[41]; // nome do vendedor
} vendedor;

/* função que cadastra um vendedor no arquivo */
void cadastrar_vendedor();

/* função que lista todos os vendedores cadastrados */
void listar_vendedores();

#endif
```

Arquivo ***venda.h***:

```
/* venda.h
 *
 * Define o tipo venda e os protótipos das operações realizadas sobre
 * esse tipo.
 *
 * Ciro Cirne Trindade
 * 15/05/2014
 */

#ifndef _VENDA_H
#define _VENDA_H

#define ARQ_VENDAS "vendas.dat"

typedef struct {
	int dia;
	int mes;
	int ano;
} data;

typedef struct {
	int cod_automovel; // código do automóvel
	int cod_vendedor; // código do vendedor
	data dt; // data da venda
} venda;

/* função que cadastra uma venda no arquivo */
void cadastrar_venda();

/* função que lista todas as vendas realizadas em um mês/ano */
void listar_vendas_mes();

/* função que lista todas as vendas realizadas por um vendedor */
void listar_vendas_vendedor();

#endif
```

Informações importantes sobre Trabalho

1. Critérios de avaliação:
   a) Corretude: 70%
   b) Interface: 20%
   c) Legibilidade: 10%
2. Os trabalhos serão testados no sistema operacional Ubuntu 10.10 e serão compilados pelos seguinte arquivo *makefile*:

```
trab2: concessionaria.o automovel.o vendedor.o venda.o
	gcc -o trab2 concessionaria.o automovel.o vendedor.o venda.o
concessionaria.o: concessionaria.c automovel.h vendedor.h venda.h
	gcc -c concessionaria.c
automovel.o: automovel.c automovel.h
	gcc -c automovel.c
vencedor.o: vendedor.c vendedor.h
	gcc -c vendedor.c
venda.o: venda.c venda.h automovel.h vendedor.h
	gcc -c venda.c
```

3. Todos os fontes do trabalho gerados pelo grupo devem possuir no cabeçalho a identificação dos autores;
3. Trabalhos copiados (com ou sem eventuais disfarces) terão a nota dividida pelo número de cópias (inclusive o original);
4. Trabalhos atrasados não serão aceitos;
5. Trabalhos com erros de sintaxe (ou seja, erros de compilação) receberão nota ZERO;
6. É muito importante que seu programa tenha comentários e esteja bem identado, ou seja, digitado de maneira a ressaltar a estrutura de subordinação dos comandos do programa. A legibilidade do código será levada em consideração pelo critério de avaliação do trabalho;
7. Você deve enviar os fontes da aplicação para o seguinte e-mail: ciroct@gmail.com
8. Guarde uma cópia do e-mail enviado ao professor para o caso de extravio. Esta cópia pode ser solicitada pelo professor em caso de alguma dúvida.