

## Trabalho 3 – Listas ligadas

Prazo de entrega: 10/11/2014 no Run.Codes

(Trabalho Individual)

### Introdução

Um mini-mercado familiar deseja informatizar sua emissão de notas fiscais e contratou você para fazer seu sistema. Deseja-se armazenar informações sobre cada cliente e sobre todos os itens comprados. Como a família é muito organizada, ela deseja que todas as informações sejam armazenadas no sistema em ordem alfabética.

Neste trabalho, você deve:

1. Utilizar as estruturas abaixo para armazenar as informações do programa:

```
#define TAMANHO 100

typedef struct t_no_objeto {
    char objeto[TAMANHO];
    int quantidade;
    struct t_no_objeto *prox;
}No_Objeto;

typedef struct t_lista_objeto{
    No_Objeto *first;
    int nElementos;
}Lista_Objeto;

typedef struct t_no_compra{
    char nomeComprador[TAMANHO];
    int dia, mes, ano;
    char CPF[15];
    struct t_no_compra *prox;
    Lista_Objeto *listaObjetos;
}No_Compra;

typedef struct t_lista_compras{
    No_Compra *first;
    int nCompras;
}Lista_Compras;
```

2. Definir funções para as seguintes operações:

- a. Inicializar as estruturas de dados;
- b. Cadastrar compra (**em ordem alfabética de nome do cliente**);

- c. Cadastrar objetos comprados por compra (**em ordem alfabética por nome do objeto**);
- d. Imprimir relatório;
- e. Desalocar as estruturas de dados.

## Entrada

A primeira linha contém um número <nCompras>, que contém o número de compras a ser cadastradas no sistema.

Para cada compra (em <nCompras>), as próximas linhas contém:

- O nome do cliente (string sem espaços);
- O cpf do cliente (string sem espaços);
- O dia, mês e ano da compra (inteiros);
- O número de objetos da compra <nObjetos>;

Para cada objeto em <nObjetos>, as próximas linhas contém:

- O nome do objeto comprado (string sem espaços);
- A quantidade de objetos.

Exemplo:

```
2
Jorge
372562333
15 10 2014
2
Batata
5
Maca
7

Ana
123456789
01 03 2014
3
Pera
5
Abacate
6
Sabao
3
```

## Saída

A saída - apresentada do dispositivo padrão – consiste de várias linhas, e contém um relatório de todas as compras efetuadas.

A primeira linha contém o texto: “Relatorio de Compras”

A próxima linha está em branco;

Para cada cliente (em ordem alfabética), as próximas linhas contém:

<Nome>

<CPF>

<Dia>/<Mês>/<Ano>

<Numero de objetos comprados>

Para cada objeto comprado (em ordem alfabética), as próximas linhas contém:

<Nome do objeto><espaço em branco><Quantidade>

Uma linha em branco (entre clientes)

Uma linha em branco (no final da saída)

Para a entrada apresentada no exemplo, a saída esperada é:

### **Relatorio de Compras**

**Ana**

**123456789**

**01/03/2014**

**3**

**Abacate 6**

**Pera 5**

**Sabao 3**

**Jorge**

**372562333**

**15/10/2014**

**2**

**Batata 5**

**Maca 7**

## Outras Informações Importantes

- O trabalho deve ser feito individualmente.
- O programa pode ser feito na linguagem C.
- Todas as submissões são checadas para evitar cópia/plágio/etc.
- Comente o seu código com uma explicação rápida do que cada função, método ou trecho importante de código faz (ou deveria fazer). Os comentários e a modularização do código (".c", ".h", "Makefile") serão checados e valem nota.
- Entradas/saídas devem ser lidas/escritas a partir dos dispositivos padrão, ou seja, use as funções "*printf(...)*" e "*scanf(...)*". Para testar, arquivos podem ser redirecionados para/de seu programa na linha de comando utilizando os operadores < e >.
- Exemplo:

```
# ./trab3 < entrada.txt > saida.txt
```