



Universidade Federal do Ceará
Centro de Ciências
Departamento de Computação

Disciplina Programação (CK0226)
Tarefa de Laboratório nº 12
Lista Circular Duplamente Encadeada Polimórfica

Prof. Miguel Franklin

OBJETIVOS:

- Implementação de estrutura de dados Lista Circular Duplamente Encadeada.
- Prática de divisão de projeto em diversos arquivos-fonte.
- Prática de ponteiros genéricos.

ENUNCIADO:

Criar uma lista duplamente encadeada que possa receber, dentro da mesma lista, valores inteiros (int), pontos flutuantes (float), caracteres (char) e cadeias de caracteres (char *), indexados por um valor inteiro chamado *chave*, que deve ser único. Na lista, os elementos devem ser inseridos sempre mantendo a ordem crescente da *chave*. As funções devem ser implementadas no arquivo `lista_polimorfica.c`, com interfaces no arquivo `lista_polimortica.h`. Uma aplicação utilizando as funções implementadas através de um menu de opções deve ser implementada no arquivo `main.c`. Na informação de tipo “cadeia de caractere”, só deve ser alocada quantidade de memória suficiente para armazenar o conteúdo da cadeia passado pelo usuário. Não utilizar enumeradores. Não inserir no registro informações que não serão utilizadas em um dado momento. As seguintes funções deverão ser implementadas:

void inicializar (Lista **l);

Inicializa a lista.

int inserir_inteiro (Lista **l, int chave, int val);

Insere um valor inteiro mantendo a ordem crescente de *chave* na lista. A função deve retornar 0 caso haja alguma falha na inserção, seja por falta de memória ou pela tentativa de inserir um elemento cuja chave já está presente na lista.

int inserir_p_flutuante (Lista **l, int chave, float val);

Insere um valor ponto flutuante mantendo a ordem crescente de *chave* na lista. A função deve retornar 0 caso haja alguma falha na inserção, seja por falta de memória ou pela tentativa de inserir um elemento cuja chave já está presente na lista.

int inserir_c_caracteres (Lista **l, int chave, char *val);

Insere uma cadeia de caracteres mantendo a ordem crescente de *chave* na lista. A função deve retornar 0 caso haja alguma falha na inserção, seja por falta de memória ou pela tentativa de inserir um elemento cuja chave já está presente na lista.

int inserir_caractere (Lista **l, int chave, char val);

Insere um caractere mantendo a ordem crescente de *chave* na lista. A função deve retornar 0 caso haja alguma falha na inserção, seja por falta de memória ou pela tentativa de inserir um elemento cuja chave já está presente na lista.

int obter_tipo (Lista *l, int chave);

Obtém o tipo de dado armazenado com o índice *chave* na lista.

int obter_inteiro (Lista *l, int chave);

Obtém o valor inteiro armazenado com o índice *chave* na lista. Se não existir esta chave, ou se a chave representar um tipo diferente de inteiro, a função deve retornar o valor -99999 e exibir uma mensagem de erro.

float obter_p_flutuante (Lista *l, int chave);

Obtém o valor ponto flutuante armazenado com o índice *chave* na lista. Se não existir esta chave, ou se a chave representar um tipo diferente de ponto flutuante, a função deve retornar o valor -99999.0 e exibir uma mensagem de erro.

char *obter_c_caracteres (Lista *l, int chave);

Obtém a cadeia de caracteres armazenada com o índice *chave* na lista. Se não existir esta chave, ou se a chave representar um tipo diferente de cadeia de caracteres, a função deve retornar NULL e exibir uma mensagem de erro.

char obter_caractere (Lista *l, int chave);

Obtém o caractere armazenado com o índice *chave* na lista. Se não existir esta chave, ou se a chave representar um tipo diferente de caractere, a função deve retornar '\0' e exibir uma mensagem de erro.

void listar_elementos (Lista *l);

Apresenta na saída padrão (tela) todos os elementos na lista, um por linha, da primeira à última posição.

A entrega (upload) deverá ser realizada através do Google Classroom.