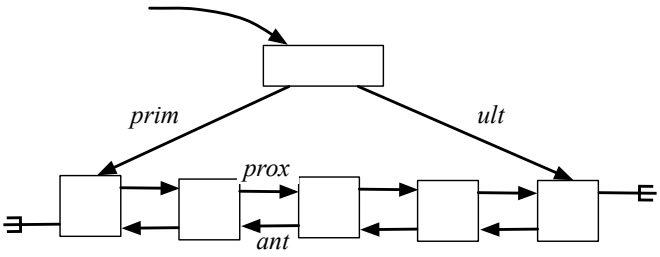
INF 1007 – Programação II

TRABALHO

*Este trabalho valerá um bônus de 0.5 ponto a ser adicionado à nota da P3. O trabalho é para ser feito individualmente. Trabalhos com alto grau de similaridade não serão considerados.*

Na estrutura de dados conhecida como *lista duplamente encadeada*, cada nó armazena, além da informação associada, dois ponteiros para o encadeamento dos nós: um ponteiro para o nó anterior e um ponteiro para o próximo nó. Desta forma, é possível, por exemplo, dado um ponteiro para um nó qualquer, acessar tanto o nó precedente quanto o nó seguinte. Com esta lista, é possível percorrer os elementos na ordem direta (do primeiro ao último) ou na ordem inversa (do último ao primeiro), a partir de um ponteiro para o primeiro ou para o último elemento, respectivamente.

Além dos nós, pode-se adotar uma estrutura que representa a lista como um todo, armazenando um ponteiro para o primeiro nó e outro ponteiro para o último nó. O diagrama abaixo ilustra esta estruturação.




Os tipos que representam esta estrutura para armazenar valores reais podem ser dados por:

```
typedef struct no No;
struct no {
  float info;
  No* prox;
  No* ant;
};

typedef struct lista Lista;
struct lista {
  No* prim;
  No* ult;
};
```

Pede-se:

1. Usando estes tipos, escreva as seguintes funções:
    a. Função para criar uma lista vazia

```
Lista* cria (void);
```

    b. Função para inserir um novo elemento no início da lista

```
void insere_inicio (Lista* l, float x);
```

    c. Função para inserir um novo elemento no final da lista

```
void insere_final (Lista* l, float x);
```

    d. Função para remover o elemento no início da lista

```
void remove_inicio (Lista* l);
```

    e. Função para remover o elemento no final da lista

```
void remove_final (Lista* l);
```

    f. Função para imprimir os elementos do início ao final

```
void imprime (Lista* l);
```

    g. Função para imprimir os elementos do final ao início

```
void imprime_reverso (Lista* l);
```

    h. Função para liberar a lista e seus elementos

```
void libera (Lista* l);
```

2. Escreva uma função `main` para testar as suas funções. É fundamental que **todas** as funções implementadas sejam testadas.

PRAZO e FORMA DE ENTREGA: O prazo para entrega do trabalho é dia **24 de junho de 2013**. Para facilitar a correção, aluno deve enviar um único arquivo .c, que deve incluir a definição dos tipos, a implementação das funções de lista e a função `main`. O nome do arquivo deve ser formado pelo nome do aluno (ex. "joao_da_silva.c"). Deve-se enviar um e-mail para **celes@inf.puc-rio.br** com o arquivo anexado. Note que deve ser enviado apenas o arquivo .c.