```c
/*
 * LISTA DUPLAMENTE ENCADEADA
 * 1) Estude o código abaixo
 * 2) Escreva uma função 'remover', análoga à função 'inserir'
 * 3) Monte seu TAD "lista duplamente encadeada", criando os respectivos arquivos .h e .c
 *
 */

#include <stdio.h>
#include <stdlib.h>
#include <string.h>

typedef struct bloco {
    char nome[50];
    struct bloco *ant, *prox;
} no;

typedef struct {
    no *ini, *fim;
} Lista;

void criar (Lista *);
void destruir (Lista *);
void inserir (Lista *, char[], int, int *);
void imprimir (Lista *);

int main (void) {
    int pos, erro;
    char nome[50];
    Lista L;

    criar (&L);

    printf ("Digite um nome ou \'fim\' para terminar: ");
    scanf ("%s", nome);
    if (strcmp (nome, "fim") ) {
        printf ("Digite a posicao em que quer inserir: ");
        scanf ("%d", &pos);
    }
    while (strcmp (nome, "fim") ) {
        inserir (&L, nome, pos, &erro);
        if (erro) {
            printf ("Nao foi possível inserir %s\n", nome);
            break;
        }
        printf ("Digite um nome ou \'fim\' para terminar: ");
        scanf ("%s", nome);
        if (strcmp (nome, "fim") ) {
            printf ("Digite a posicao em que quer inserir: ");
            scanf ("%d", &pos);
        }
    }

    printf ("Os elementos da lista sao:\n");
    imprimir (&L);

    destruir (&L);
    return 0;
}


void criar (Lista *L) {
    L->ini = NULL;
    L->fim = NULL;
}

void destruir (Lista *L) {
    no *p;
    p = L->ini;
    while (p != NULL) {
        L->ini = L->ini->prox;
        free (p);
```

```
        p = L->ini;
    }
}

void imprimir (Lista *L) {
    no *p;
    p = L->ini;
    while (p != NULL) {
        printf ("%s\n", p->nome);
        p = p->prox;
    }
}

void inserir (Lista *L, char x[], int posicao, int *erro) {
    no *p, *aux1, *aux2;

    p = (no*) malloc (sizeof(no));
    if (p == NULL)
        *erro = 1;
    else {
        *erro = 0;
        strcpy (p->nome, x);
        if (L->ini == NULL) {
            L->ini = p;
            L->fim = p;
            p->ant = NULL;
            p->prox = NULL;
        } else {
            aux1 = L->ini;
            while ((aux1 != NULL) && (posicao > 1)) {
                aux1 = aux1->prox;
                posicao--;
            }
            if (aux1 == L->ini) {
                p->ant = NULL;
                p->prox = L->ini;
                L->ini->ant = p;
                L->ini = p;
            } else if (aux1 == NULL) {
                p->ant = L->fim;
                p->prox = NULL;
                L->fim->prox = p;
                L->fim = p;
            } else {
                aux2 = aux1->ant;
                aux1->ant = p;
                p->prox = aux1;
                aux2->prox = p;
                p->ant = aux2;
            }
        }
    }
}
```