Laboratório de Listas Encadeadas

1) Implemente o TAD ListaEncadeada utilizando um nó descritor como o mostrado abaixo

Padronização de erros

Sugestão para padronizar códigos de erro

#define SUCCESS 0

#define OUT_OF_MEMORY -1

#define ELEM_NOT_FOUND -2

….

// ListaEncadeada.c

```
typedef struct lista_no Lista_no;

struct lista{
   Lista_no *head;
};

struct lista_no{
    struct aluno dado;
    Lista_no *prox;
};
```

// ListaEncadeada.h

```
typedef struct lista Lista;

Lista* cria_lista();
void libera_lista(Lista* li);
int consulta_lista_pos(Lista* li, int pos, struct aluno *al);
int consulta_lista_mat(Lista* li, int mat, struct aluno *al);
int insere_lista_final(Lista* li, struct aluno al);
int insere_lista_inicio(Lista* li, struct aluno al);
int insere_lista_pos(Lista* li, int pos, struct aluno al);
int insere_lista_ordenada(Lista* li, struct aluno al);
int remove_lista(Lista* li, int mat);
int remove_lista_inicio(Lista* li);
int remove_lista_final(Lista* li);
```

```
int tamanho_lista(Lista* li);
int lista_cheia(Lista* li);
int lista_vazia(Lista* li);
void imprime_lista(Lista* li);
```

2) Crie um novo TAD a partir do TAD do exercício anterior que inclua as seguintes modificações no nó descritor

```
struct lista {
   Lista_no *head;
   int qte; // quantidade de elementos na lista
   int ordenada; // indica se a lista é ordenada
};
```

Caso uma lista seja considerada ordenada, o usuário só poderá utilizar a função *insere_lista_ordenada* para inserção. Caso contrário, só estarão disponíveis as funções de inserção no início, fim e em uma posição específica. Caso o usuário tente utilizar uma função não disponível, a função deve retornar um código de erro.