Exercício 3 –Construir o arquivo header Elemento.h. Testar todas as funções.

/* TAD TipoElemento exemplo 3*/
/* TipoElemento guarda os dados referentes a um aluno: nome e RA. */

```
#ifndef _Elemento_h
        #define _Elemento_h
typedef struct{
      char * nome;
      char *RA;
}TipoElemento;

    TipoElemento criarElemento(char *, char *);
    TipoElemento criarNovoElemento();
    TipoElemento criarFantasma();
    char* get_Nome(TipoElemento);
    char* get_RA(TipoElemento );
    void mostrarElemento(TipoElemento);
    void set_Nome(TipoElemento *, char *);
    void set_RA(TipoElemento *, char *);

//implementações
TipoElemento criarElemento(char *s, char * r){
        TipoElemento reg;
        reg.nome = s; reg.RA = r;
        return reg;
}

TipoElemento criarNovoElemento(){
        TipoElemento novo;
        novo.nome = "MARIA"; novo.RA = “00000001”;
        return novo;
}

TipoElemento criarFantasma(){
        TipoElemento f;
        f.nome = "fantasma"; f.RA = “00000000”;
        return f;
}

void set_Nome(TipoElemento *reg, char *s){
      reg->nome = s;
}

void set_RA(TipoElemento *reg, char *m){
   reg->RA = m;
}

char * get_Nome(TipoElemento reg){
   char * s;
   s = reg.nome;
   return s;
}

char get_RA(TipoElemento reg){
   char * m;
   m = reg.RA;
   return m;
}

void mostrarElemento(TipoElemento reg){
   printf(" nome = %s   ",reg.nome);      printf("   RA%s   ", reg.RA);
}

#endif
```

Exercício 4

Construir o TAD Lista usando a estrutura a seguir. Testar todas as funções básicas de lista. Definir o que for necessário.

```
typedef struct {
   int tamanho;
   TipoElemento  tab[Maximo];
} Lista;
```