```c
#include "estrutura.h"
#include<stdio.h>
#include<string.h>
#include<stdlib.h>



int ordAlf(char *nova, char *existe){
        int i;

        if(strcmp(nova,existe)==0) return -1;

        for(i=0;i<(strlen(nova));i++){
                if(toupper(nova[i])>toupper(existe[i])) return 0;
                if(toupper(nova[i])<toupper(existe[i])) return 1;
        }

        return 1;
}

List guardaDoc(Docs doc, List lista){

        List newDocSave = (List) malloc(sizeof(struct listDocs));

        //inserir cenas
        newDocSave->doc=doc;


        if(lista==NULL) {
                newDocSave->next=NULL;
                return newDocSave;
        }

        List ant = lista, aux = lista;

        int res=ordAlf(doc->titulo,lista->doc->titulo);
        if(res==-1) return lista;
        if(res){
                newDocSave->next=lista;
                return newDocSave;
        }

        while(aux!=NULL){
                res=ordAlf(doc->titulo,aux->doc->titulo);//função de ordenar
                if(res==-1) return lista;
                if(res){
                        ant->next=newDocSave;
                        newDocSave->next=aux;
                        return lista;
                }
                ant=aux;
                aux=aux->next;
        }

        ant->next=newDocSave;
        newDocSave->next=NULL;

        return lista;
}

//inicializar o documento

Docs initDocument(){

        Docs document = (Docs)malloc(sizeof(struct Document));
        document->nInter=0;
        document->linkInter=NULL;
```

```
        document->nExt=0;
        document->linkExt=NULL;
        document->nSec=0;
        document->linkSec=NULL;

        return document;
}



Docs inserLinkInter(char* link, Docs doc){
        Links nLink = (Links)malloc(sizeof(struct links));

        //inserir cenas
        char *link1 = (char *)malloc(sizeof(char *[strlen(link)+1]));
        strcpy(link1,link);
        nLink->nomeLinks=link1;


        if(doc->linkInter==NULL) {
                doc->linkInter=nLink;
                nLink->next=NULL;
                doc->nInter++;
                return doc;
        }

        Links ant = doc->linkInter, aux = doc->linkInter;

        int res=ordAlf(link,aux->nomeLinks);
        if(res==-1) return doc;
        if(res){
                doc->linkInter = nLink;
                nLink->next=aux;
                doc->nInter++;
                return doc;
        }

        while(aux!=NULL){
                res=ordAlf(link,aux->nomeLinks);//função de ordenar
                if(res==-1) return doc;
                if(res){
                        ant->next=nLink;
                        nLink->next=aux;
                        doc->nInter++;
                        return doc;
                }
                ant=aux;
                aux=aux->next;
        }

        ant->next=nLink;
        nLink->next=NULL;
        doc->nInter++;

        return doc;
}

Docs inserLinkExt(char* link, Docs doc){
        Links nLink = (Links)malloc(sizeof(struct links));

        //inserir cenas
        char *link1 = (char *)malloc(sizeof(char *[strlen(link)+1]));
        strcpy(link1,link);
        nLink->nomeLinks=link1;


        if(doc->linkExt==NULL) {
```

```
                doc->linkExt=nLink;
                nLink->next=NULL;
                doc->nExt++;
                return doc;
        }

        Links ant = doc->linkExt, aux = doc->linkExt;

        int res=ordAlf(link,aux->nomeLinks);
        if(res==-1) return doc;
        if(res){
                doc->linkExt = nLink;
                nLink->next=aux;
                doc->nExt++;
                return doc;
        }

        while(aux!=NULL){
                res=ordAlf(link,aux->nomeLinks);//função de ordenar
                if(res==-1) return doc;
                if(res){
                        ant->next=nLink;
                        nLink->next=aux;
                        doc->nExt++;
                        return doc;
                }
                ant=aux;
                aux=aux->next;
        }

        ant->next=nLink;
        nLink->next=NULL;
        doc->nExt++;

        return doc;
}


Docs inserLinkSec(char* link, Docs doc){
        Links nLink = (Links)malloc(sizeof(struct links));

        //inserir cenas
        char *link1 = (char *)malloc(sizeof(char *[strlen(link)+1]));
        strcpy(link1,link);
        nLink->nomeLinks=link1;


        if(doc->linkSec==NULL) {
                doc->linkSec=nLink;
                nLink->next=NULL;
                doc->nSec++;
                return doc;
        }

        Links ant = doc->linkSec, aux = doc->linkSec;

        int res=ordAlf(link,aux->nomeLinks);
        if(res==-1) return doc;
        if(res){
                doc->linkSec = nLink;
                nLink->next=aux;
                doc->nSec++;
                return doc;
        }

        while(aux!=NULL){
                res=ordAlf(link,aux->nomeLinks);//função de ordenar
```

```
                if(res==-1) return doc;
                if(res){
                        ant->next=nLink;
                        nLink->next=aux;
                        doc->nSec++;
                        return doc;
                }
                ant=aux;
                aux=aux->next;
        }

        ant->next=nLink;
        nLink->next=NULL;
        doc->nSec++;

        return doc;
}

Docs inserTitulo(char* titulo, Docs doc){
        char *titulo1 = (char *)malloc(sizeof(char *[strlen(titulo)+1]));
        strcpy(titulo1,titulo);
        doc->titulo=titulo1;

        return doc;
}

Docs inserAutorLastRec(char* autor, Docs doc){
        char *autor1 = (char *)malloc(sizeof(char *[strlen(autor)+1]));
        strcpy(autor1,autor);
        doc->autorLastRev=autor1;

        return doc;
}

Docs inserDateLastRec(char* date, Docs doc){
        char *date1 = (char *)malloc(sizeof(char *[strlen(date)+1]));
        strcpy(date1,date);
        doc->dateLastRev=date1;

        return doc;
}

/*
int main(){

        Docs doc = NULL;
        doc = initDocument();

        //doc = inserTitulo("Miguel", doc);

        doc = inserLinkExt("b", doc);
        doc = inserLinkExt("a", doc);
        doc = inserLinkExt("bc", doc);
        doc = inserLinkExt("c", doc);

        while(doc->linkExt!=NULL){
                printf("%s\n", doc->linkExt->nomeLinks);
                doc->linkExt=doc->linkExt->next;
        }

        return 1;
}*/
```