**ÚLTIMA ACTUALIZACIÓN: 10/06/2021 14:00**

| **#include <stdio.h>** **#include <stdlib.h>** **#include <string.h>**  */\*\*   Lista - Sruct - Archivo de Texto - Recursividad    Pasa el contenido del archivo arch.txt a una lista Carga la lista en forma ordenada imprime la lista  Luego ordena la list con otro criterioOrden imprime la lista   \*/*  **struct** **sFecha**{*//12*  **int** dd,mm,aaaa;  }; **typedef** **struct** **sFecha** **tFecha**; **struct** **sContent**{*//24*  **int** dni;  **char**\* apellido;  **char**\* nombre;  tFecha fechaNac; }; **typedef** **struct** **sContent** **tContent**;  **struct** **sNodo**{*//28*  tContent content;  **struct** **sNodo**\* **sig**; }; **typedef** **struct** **sNodo**\* **tNodo**;    **void** **insertarEnListaUltimo**(tNodo\* , tContent ); **void** **insertarEnListaOrdenado**(tNodo\* nodo, tContent contenido, **int** nCampo, **int** asc); **void** **imprimirListaR**(tNodo); **int** **escribirArchDesdeLista**(**char**\* ,tNodo ); **int** **cargarListaOrdenadaDesdeArch**(tNodo\* lista,**char**\* archNom, **int** nCampo, **int** asc); **int** **criterioOrden**(tContent,tContent,**int** nCampo,**int** asc); **void** **leerCadDeArch**(FILE\*,**char**\*\*,**char**); **void** **ordenarLista**(tNodo ,**int** nCampo,**int** asc);  **void** **ordenarElemento**(tNodo nodo, tNodo lista,**int** nCampo,**int** asc);  **int** **main**(){   tNodo lista=NULL;   printf("Cargar la lista insertando ordenado por un campo, desde la fuente de un archivo\n");  cargarListaOrdenadaDesdeArch(&lista,"arch.txt",2,0);   imprimirListaR(lista);    printf ("\n");  printf("Cambiar el orden de la lista, dejandola ordenada por otro campo\n");  ordenarLista( lista ,1,1);  imprimirListaR(lista);    *//escribirArchDesdeLista("archOrd.txt", lista);*   **return** 0; }  **int** **escribirArchDesdeLista**(**char**\* nomArch,tNodo lista){  FILE\* arch = fopen(nomArch,"w");  **if** (arch==NULL){**return** -1;}  tContent auxCont;    **while**(lista!=NULL){  auxCont = lista->content;  *//fprintf(arch,"%d,",auxCont.dni);*  fprintf(arch,"%s,",auxCont.apellido);  fprintf(arch,"%s,",auxCont.nombre);  fprintf(arch,"%d",auxCont.fechaNac.aaaa);  fprintf(arch,"%d",auxCont.fechaNac.mm);  fprintf(arch,"%d\n",auxCont.fechaNac.dd);  lista = lista->sig;  }   fclose(arch);  **return** 0; }   **int** **cargarListaOrdenadaDesdeArch**(tNodo\* lista,**char**\* archNom, **int** nCampo, **int** asc){    FILE\* arch = fopen(archNom,"r");  **if** (arch==NULL){**return** -1;}  *//tContent auxCont={0};*  tContent auxCont={0,NULL,NULL,{0,0,0}};  **int** auxI;     **while**(!feof(arch)){   fscanf(arch, "%d,",&(auxCont.dni));   leerCadDeArch(arch,&(auxCont.apellido),',');  leerCadDeArch(arch,&(auxCont.nombre),',');   fscanf(arch, "%d\n",&auxI);     *//separo el número que contiene la fecha en*  *// aaaa mm dd y lo almaceno en la estructura*  auxCont.fechaNac.dd=auxI%100;  auxCont.fechaNac.mm=(auxI/100)%100;  auxCont.fechaNac.aaaa=auxI/10000;    *//insertarEnListaUltimo(lista, auxCont);*  insertarEnListaOrdenado(lista, auxCont,nCampo, asc);  }   fclose(arch);  **return** 0; }  **void** **insertarEnListaUltimo**(tNodo\* nodo, tContent contenido){  *//insertar <al final de la lista>*  **if** ( \*nodo == NULL){  \*nodo = (tNodo) malloc(**sizeof**(struct sNodo));  (\*nodo)->content = contenido;   (\*nodo)->sig = NULL;  }  **else**{  insertarEnListaUltimo(&((\*nodo)->sig), contenido);   } }  **void** **imprimirListaR**(tNodo lista) {  *// Imprimir Recursivo*  tContent auxCont;  **if**(lista!=NULL){  auxCont = lista->content;  *//printf("%10d",&(lista->content));*  printf("%4d",auxCont.dni);  printf("%18s",auxCont.apellido);  printf("%10s",auxCont.nombre);  printf("%5d",auxCont.fechaNac.aaaa);  printf("%3d",auxCont.fechaNac.mm);  printf("%3d\n",auxCont.fechaNac.dd);    imprimirListaR(lista->sig);  } }  **void** **insertarEnListaOrdenado**(tNodo\* nodo, tContent contenido, **int** nCampo, **int** asc){  tNodo aux=NULL;  **if** ( \*nodo == NULL){ */\*\*INSERTO ULTIMO\*/*  \*nodo = (tNodo) malloc(**sizeof**(struct sNodo));  (\*nodo)->content = contenido;   (\*nodo)->sig = NULL;  }  **else** {  *//if ( ((\*nodo)->content).dni > contenido.dni){*  **if** ( criterioOrden (((\*nodo)->content),contenido,nCampo,asc)){  aux = (tNodo) malloc(**sizeof**(struct sNodo));  aux->sig=(\*nodo);  aux->content = contenido;   (\*nodo)=aux;  }  **else**{  insertarEnListaOrdenado(&(\*nodo)->sig, contenido,nCampo,asc);  }  } }  **int** **criterioOrden**(tContent conA,tContent conB,**int** nCampo,**int** asc){  *// es una función booleana que retorna 1 de acuerdo a los*  *// contenido conA y conB y de acuerdo a los criterios*  *// nCampo es el núemrod de campo y asc es 1 significa ascendente*  **int** res=0;  **if** (nCampo==0){  **if** (asc==0){  **if** (conA.dni<conB.dni)  res=1;  }**else**{  **if** (conA.dni>conB.dni)  res=1;  }  }  **else** **if** (nCampo==1){  **if** (asc==0){  **if** (strcmp(conA.apellido,conB.apellido)==-1)  res=1;  }**else**{  **if** (strcmp(conA.apellido,conB.apellido)==1)  res=1;  }   }  **else** **if** (nCampo==2){  **if** (asc==0){  **if** (strcmp(conA.nombre,conB.nombre)==-1)  res=1;  }**else**{  **if** (strcmp(conA.nombre,conB.nombre)==1)  res=1;  }  }  **else** **if** (nCampo==3){  *// convierto los campos separados de fecha a un numero único*  **int** aucA = conA.fechaNac.dd + conA.fechaNac.mm\*100 + conA.fechaNac.aaaa\*10000;  **int** aucB = conB.fechaNac.dd + conB.fechaNac.mm\*100 + conB.fechaNac.aaaa\*10000;    **if** (asc==0){  **if** (aucA<aucB)  res=1;  }**else**{  **if** (aucA>aucB)  res=1;  }   }   **return** res; }  **void** **leerCadDeArch**(FILE\* arch,**char**\*\* arrC,**char** sep){  *// Lee cadena de un archivo arch hasta encontrar separador sep*  *// la cadena leída la guarda en un arreglo dinámico arrC como string*  **char** car;  **int** i=0;  \*(arrC)=malloc(**sizeof**(**char**));  **if** (!feof(arch))  car=fgetc(arch);  **while**(!feof(arch) && car!=sep){  (\*(arrC))[i]=car;  i++;  \*(arrC)=realloc(\*(arrC),(i+1)\***sizeof**(**char**));  car=fgetc(arch);  }  (\*(arrC))[i]='\0'; }     **void** **ordenarElemento**(tNodo nodo, tNodo lista,**int** nCampo,**int** asc){  tContent contAux;   **while**(lista!=NULL && nodo!=NULL){   *//if (nodo->content.dni > lista->content.dni ) {*  **if** ( criterioOrden (lista->content ,nodo->content,nCampo,asc)){  contAux =lista->content ;   lista->content = nodo->content;  nodo->content = contAux;   }   lista = lista->sig;  } } **void** **ordenarLista**(tNodo lista,**int** nCampo,**int** asc){  **while**(lista!=NULL){  ordenarElemento(lista, lista->sig,nCampo,asc);   lista = lista->sig;  }  } |
| --- |
|  |
|  |

**SALIDA**

| Cargar la lista insertando ordenado por un campo, desde la fuente de un archivo  800 Santander Marcos 2003 10 12  700 Cohen Rodriguez Luis 2004 10 14  100 Laprida Juan 1977 11 13  600 Ruiz Ana 2001 9 11  300 Sesto Alberto 1977 11 12  Cambiar el orden de la lista, dejando la lista ordenada por otro campo  300 Sesto Alberto 1977 11 12  800 Santander Marcos 2003 10 12  600 Ruiz Ana 2001 9 11  100 Laprida Juan 1977 11 13  700 Cohen Rodriguez Luis 2004 10 14  << El programa ha finalizado: código de salida: 0 >>  << Presione enter para cerrar esta ventana >> |
| --- |

**arch.txt**

| 800,Santander,Marcos,20031012 100,Laprida,Juan,19771113 700,Cohen Rodriguez,Luis,20041014 600,Ruiz,Ana,20010911 300,Sesto,Alberto,19771112 |
| --- |