En esta práctica se pueden usar los **elementos de C permitidos** en prácticas anteriores, y hay que seguir teniendo en cuenta las **restricciones** de funcionamiento de esas prácticas.

Esta práctica está diseñada para ejercitar el uso de **ficheros de texto y binarios**.

Se deben **usar todos y cada uno de los prototipos** indicados en esta práctica, y los elementos de prácticas anteriores que se requieran.

El funcionamiento de la práctica sería el reflejado en las figuras siguientes. El fichero de texto se denomina "alum.txt", el binario "alum.bin", y el de texto ordenado "alum_sort.txt".

```
generandose datos aleatorios...
02848500L   garcia, jose          03-01-1999
43472683S   perez, sandra         13-08-2004
07512799X   hernandez, isabel     02-10-2001
75882588E   garcia, sandra        11-11-2004
40039058Z   garcia, mario         23-03-2000
28985701C   garcia, roberto       21-03-2005
38730597T   garcia, pilar         31-01-2001
90599437E   sanchez, fernando     03-02-2005
08768050J   fernandez, fernando   03-07-1999
83437527J   hernandez, isabel     31-07-2001
58149366F   garcia, pilar         21-09-2001
33181208M   gomez, mario          31-10-2004
18929468P   gomez, sandra         05-01-2004
56455603X   gutierrez, mario      09-02-1999
64887004W   perez, jose           19-05-2001
20567945B   fernandez, roberto    31-08-1999
40771861Z   lopez, sandra         13-10-2005
50713526K   sanchez, jose         14-04-2004
64620553Y   gutierrez, jose       09-07-2005
72125645Z   gutierrez, roberto    13-12-2003
57435021L   gutierrez, raquel     02-01-2001
72944060C   fernandez, roberto    01-11-1999
77601875J   lopez, sandra         20-09-1999
38217262W   fernandez, jose       21-03-2002
76708458P   gomez, mario          29-05-2002
42011850G   fernandez, fernando   12-10-2001
13131312Z   sanchez, fernando     26-11-1999
88749122B   gomez, mario          23-02-2004
```

Figura 1. Ejemplo de ejecución del programa. Parte primera.

```
generandose fichero de texto...

generandose fichero binario...

anyo a buscar en los datos del fichero binario? 2000
40039058Z   garcia, mario          23-03-2000
54390220G   garcia, fernando       18-08-2000

generar fichero de texto con datos ordenados por dni (1) o por nombre completo (2) ? 2

nombre completo a buscar en los datos del fichero ordenado de texto ? garcia, pilar
58149366F   garcia, pilar          21-09-2001
38730597T   garcia, pilar          31-01-2001

Process returned 0 (0x0)   execution time : 118.076 s
Press any key to continue.
```

Figura 2. Ejemplo de ejecución del programa. Parte segunda.

```
//include
#include "stdio.h"
#include "time.h"  // time()
#include "stdlib.h" // srand(), rand()

//define
#define A 80


// struct
typedef struct
{  unsigned dia, mes, anyo;
```

```
} DATE;

struct ALUMNO
{ char DNI[9+1];
  char nom_comp [20+1];
  DATE nac;
};


//prototipos usados en practicas anteriores

void print_DATE(DATE ); //Listo
void print_ALUMNO (struct ALUMNO);
void rand_DATE(DATE  *); //Listo
void rand_nom_comp(char [20+1]); //Listo
void rand_ALUMNO(struct ALUMNO  *); //Listo
unsigned es_fecha_valida(DATE); //Listo

void rand_str_DNI(char [9+1]);

//prototipos de esta práctica
void all_swap(struct ALUMNO *, struct ALUMNO *);
void all_bubbleSort(struct ALUMNO  [A], unsigned);
void fprint_DATE(FILE *, DATE);
void fprint_ALUMNO (FILE *, struct ALUMNO);


//main

int main()
{
   int anyo_buscar,longitud;
   unsigned opcion;
   char subnombre [20+1];
   char leernombre [255+1];
   srand(time(NULL));

   FILE * fichero_alumno;
   FILE * binario;
   FILE * fichero_alumno_ordenado;

   struct ALUMNO alumnos [A];

   printf("generando datos aleatorios...\n");

   int i;
   for(i = 0;i < A;i++){
      rand_ALUMNO(&alumnos[i]);
      print_ALUMNO(alumnos[i]);
   }

   printf("\n\n");
   printf("generandose fichero de texto...\n");
```

```
fichero_alumno = fopen("alum.txt","w");

if(!fichero_alumno){
   printf("ERROR al abrir el fichero .txt");
   exit(1);
}


for(i = 0; i < 80;i++){
   fprint_ALUMNO(fichero_alumno,alumnos[i]);
   fprintf(fichero_alumno,"\n");
}

fclose(fichero_alumno);

printf("generandose fichero binario...\n");

binario = fopen("alum.bin","wb");

if(!binario){
   printf("ERROR al abrir el fichero .bin");
   exit(1);
}

fwrite(alumnos,sizeof(struct ALUMNO),80,binario);

fclose(binario);

printf("anyo a buscar en los datos del fichero binario? ");
scanf("%i",&anyo_buscar);
while(anyo_buscar < 1999 || anyo_buscar > 2005){
   printf("anyo a buscar en los datos del fichero binario? ");
   scanf("%i",&anyo_buscar);
}

binario = fopen("alum.bin","rb");

if(!binario){
   printf("ERROR al abrir el fichero .bin");
   exit(1);
}

struct ALUMNO aux;

while(fread(&aux,sizeof(struct ALUMNO),1,binario) != NULL){
   if(aux.nac.anyo == anyo_buscar){
      print_ALUMNO(aux);
   }
}

fclose(binario);
```

```
   printf("generar fichero de texto con los datos ordenados por DNI (1) o nombre completo (2) ? ");
   scanf("%u",&opcion);
   while(opcion < 1 || opcion > 2){
      printf("generar fichero de texto con los datos ordenados por DNI (1) o nombre completo (2) ? ");
      scanf("%u",&opcion);
   }

   fichero_alumno_ordenado = fopen("alum_sort.txt","w");

   if(!fichero_alumno_ordenado){
      printf("ERROR al abrir el fichero .txt");
      exit(1);
   }

   all_bubbleSort(alumnos,opcion);

   for(i = 0; i < 80;i++){
      fprint_ALUMNO(fichero_alumno_ordenado,alumnos[i]);
      fprintf(fichero_alumno_ordenado,"\n");
   }

   fclose(fichero_alumno_ordenado);

   fichero_alumno_ordenado = fopen("alum_sort.txt","r");

   if(!fichero_alumno_ordenado){
      printf("ERROR al abrir el fichero .txt");
      exit(1);
   }

   printf("\nnombre completo a buscar en los datos del fichero ordenado de texto ? ");

   fflush(stdin);
   gets(subnombre);

   longitud = strlen(subnombre);

   if(longitud > 0 && subnombre[longitud-1] == '\n'){
      subnombre[longitud-1] == '\0';
   }

   while(fgets(leernombre,256,fichero_alumno_ordenado) != NULL){
      if(strstr(leernombre,subnombre)!= NULL){
         printf("\n%s",leernombre);
      }
   }

   fclose(fichero_alumno_ordenado);
   return 0;
}
```

```
// definición de las funciones

void rand_str_DNI(char str_DNI[9+1]){
   int i,Digito,DNI = 0,resto,letraCalculada;
   char letraCalculadaDNI[] = {'T', 'R', 'W', 'A', 'G', 'M', 'Y', 'F', 'P', 'D', 'X',
   'B', 'N', 'J', 'Z', 'S', 'Q', 'V', 'H', 'L', 'C', 'K', 'E'};
   for(i = 0;i < 9;i++){
      Digito = rand() % 10;// numero del 0 - 9
      DNI = DNI * 10 + Digito;
      str_DNI[i] = Digito + '0' ;
   }
   resto = DNI % 23;
   letraCalculada = letraCalculadaDNI[resto];
   str_DNI[8] = letraCalculada;
   str_DNI[9] = '\0';
}


unsigned es_fecha_valida(DATE fecha) {

   unsigned Diasmes [12] = {31,28,31,30,31,30,31,31,30,31,30,31};
   if ((fecha.anyo % 4 == 0 && fecha.anyo % 100 != 0) || (fecha.anyo % 400 == 0)){
      Diasmes[1] = 29;
   }

   if(fecha.dia > Diasmes[fecha.mes - 1]){
      return 0;
   } else return 1;

}
void rand_DATE(DATE  *fecha){
   do{
      fecha->dia = (rand()%31)+1;
      fecha->mes = (rand() %12)+1;
      fecha->anyo = (rand() %7)+1999;
   }while(!es_fecha_valida(*fecha));
}

void print_DATE(DATE fecha){
   printf("%02u - %02u - %04u\n",fecha.dia,fecha.mes,fecha.anyo);
}


void rand_nom_comp(char nombre_completo[20+1]){ //strcat (concatena)
   nombre_completo [0] = '\0';
   strcat(nombre_completo,apellidos[rand()%8]);
   strcat(nombre_completo, ", ");
   strcat(nombre_completo,nombres[rand()%8]);
}

void rand_ALUMNO(struct ALUMNO  *alumno){
   rand_str_DNI(alumno->DNI);
```

```
   rand_nom_comp(alumno->nom_comp);
   rand_DATE(&alumno->nac);
}

void print_ALUMNO(struct ALUMNO alumno) {
   printf("%-15s%-32s", alumno.DNI, alumno.nom_comp);
   print_DATE(alumno.nac);
}


void all_swap(struct ALUMNO *alumno1, struct ALUMNO *alumno2){
   struct ALUMNO aux;
   aux = *alumno1;
   *alumno1 = *alumno2;
   *alumno2 = aux;
}

void all_bubbleSort(struct ALUMNO  alumno[A], unsigned opcion){
   int i,j;
   for(j = A-1;j>=0;j--){
      for(i=0;i<j;i++){
         if(opcion == 1){
            if(strcmp(alumno[i].DNI,alumno[i+1].DNI) > 0){
               all_swap(&alumno[i],&alumno[i+1]);
            }
         }else if(opcion == 2){
            if(strcmp(alumno[i].nom_comp,alumno[i+1].nom_comp) > 0){
               all_swap(&alumno[i],&alumno[i+1]);
            }
         }
      }
   }
}


void fprint_DATE(FILE * datos, DATE nac){
   fprintf(datos,"%02u - %02u - %04u",nac.dia,nac.mes,nac.anyo);
}

void fprint_ALUMNO (FILE * datos, struct ALUMNO alumno) {
   fprintf(datos,"%-15s%-32s",alumno.DNI, alumno.nom_comp);
   fprint_DATE(datos,alumno.nac);
}
```