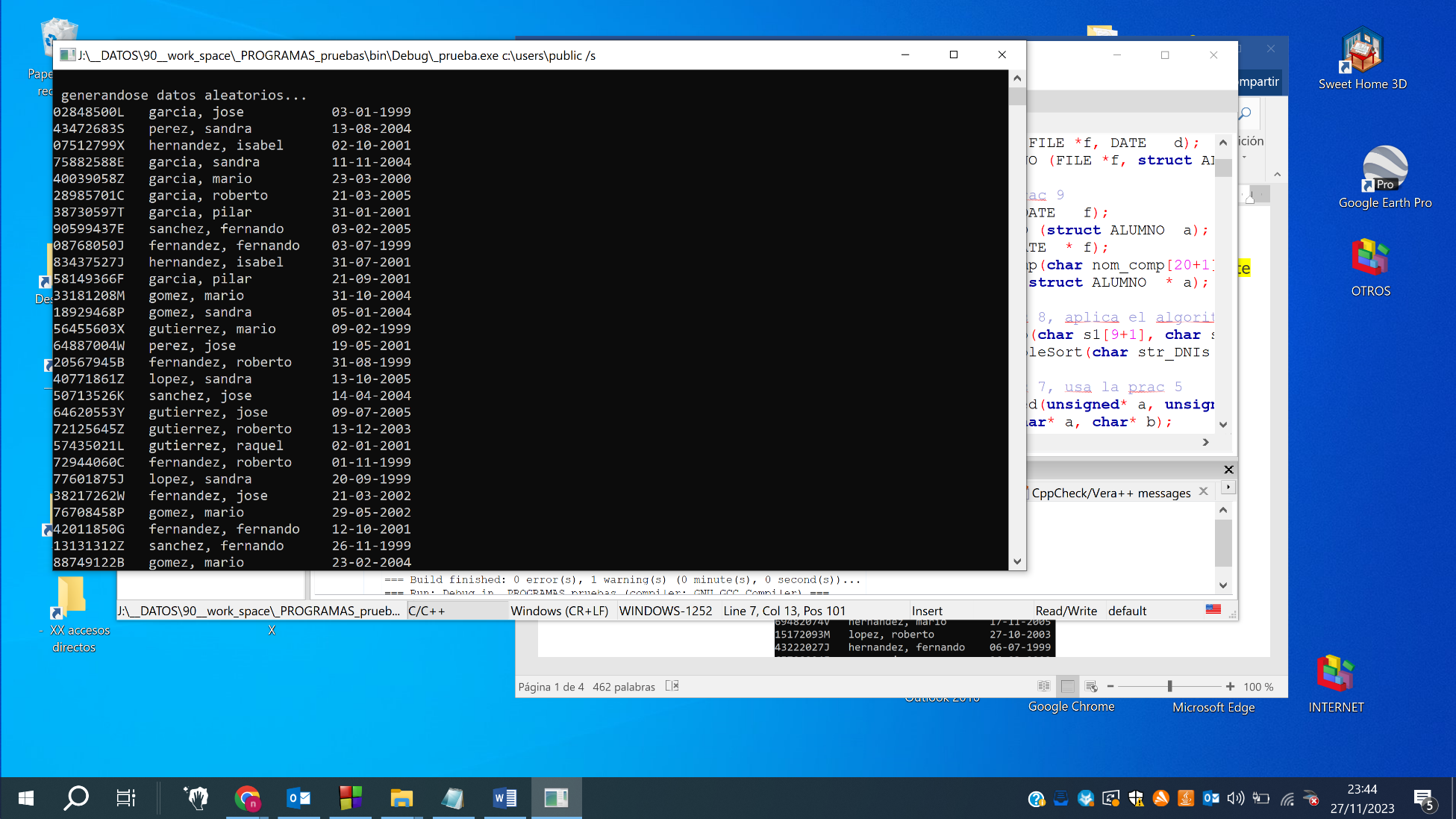
En esta práctica se pueden usar los **elementos de C permitidos** en prácticas anteriores, y hay que seguir teniendo en cuenta las **restricciones** de funcionamiento de esas prácticas.

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”.



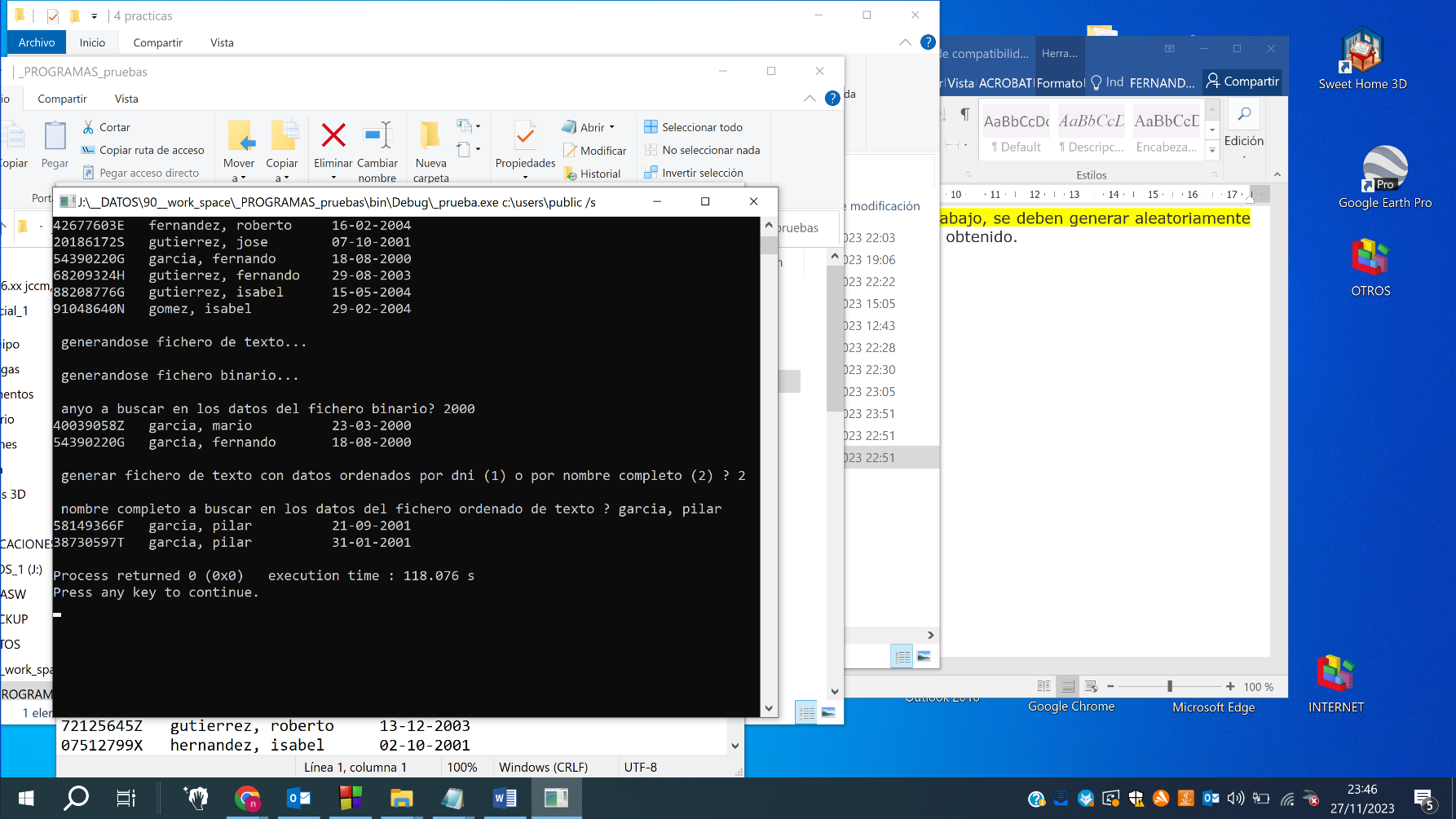


Figura 2. Ejemplo de ejecución del programa. Parte segunda.

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

//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);

}