UNIVERSIDAD TECNOLÓGICA NACIONAL

Facultad Regional Buenos Aires

–2020–

MATERIA: ALGORITMOS Y ESTRUCTURAS DE DATOS

DOCENTE: PABLO DAMIÁN MENDEZ

Localización [CAMPUS / MEDRANO]								Curso: K1024				
Legajo	**Apellido y Nombre**							**DNI**				**Correos**
176.175-4	Lara Alyster							95.727.771				alaramartnez@est.frba.utn.edu.ar
175.402-6	Gómez Agustín Ezequiel							42.314.422				gomez.agustin1011@gmail.com
175.446-4	Gonzalez Fleitas María Agustina							43.242.294				mgonzalezfleitas@est.frba.utn.edu.ar
Entrega / Revisión												
Fechas de entrega	19/09/2020				03/10/2020							
Fecha de calificación												
Calificación	A	B	C	D	A	B	C	D	A	B	C	D
Firma del Docente												
OBSERVACIONES:												

En este informe explicaremos las hipótesis y soluciones que utilizamos para resolución del Trabajo Practico Nro. 1 de Algoritmos y Estructuras de Datos, también se incluirán diagramas de Lindsay de algunos de los subprogramas.

Para comenzar, mostraremos el funcionamiento en el que interpretamos las consignas. Dividimos las consignas en 8 puntos fundamentales, las cuales serán explicadas a continuación:

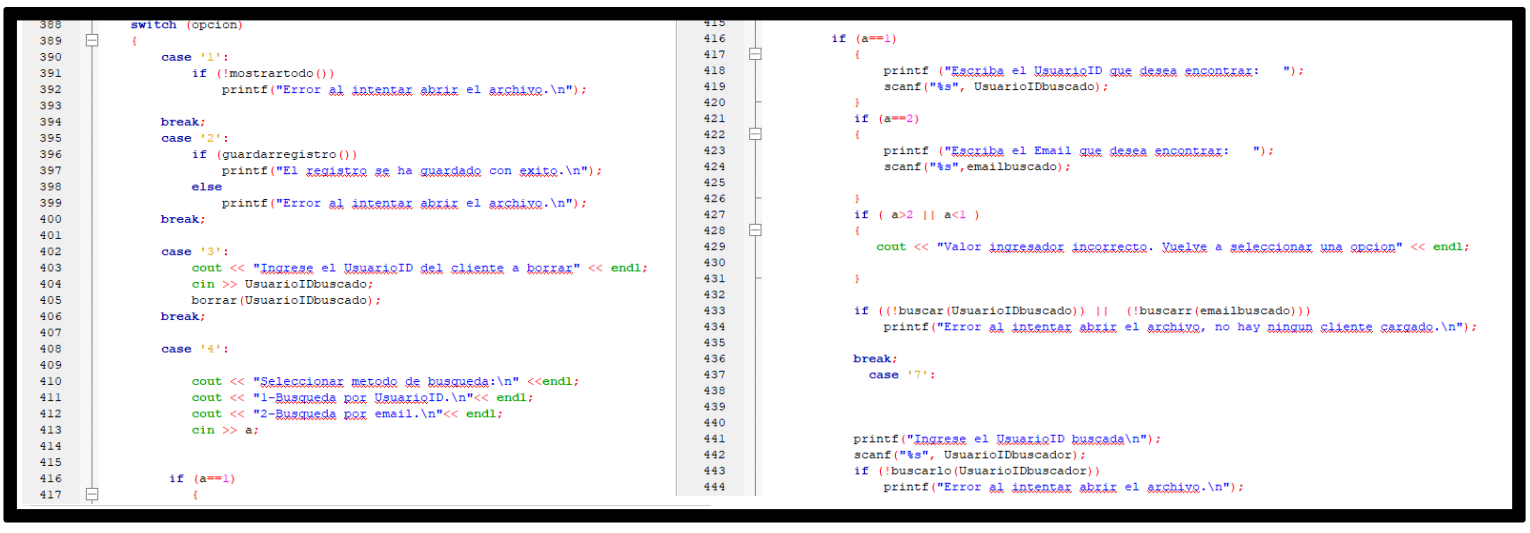
1) Crear el menú con las correspondientes opciones.
2) Cargar clientes.
3) Búsqueda de clientes
4) Borrar clientes.
5) Listar clientes.
6) Procesar lote de compras.
7) Mostrar compras del usuario.
8) Finalizar jornada.

1) Crear el menú con las correspondientes opciones.

Para este punto, creamos subprograma correspondiente a "**Char** menú", dentro de este creamos dos variables llamadas "**int** a" y "**char** opción". Procedimos a crear las diferentes opciones del menú por medio de continuos "cout" tales como: *"Levantar los clientes", "Cargar un nuevo cliente", "Desactivar un cliente existente", "Buscar un cliente por ID o por mail", "Listar todos los clientes activos ordenados por total del importe", "Procesar un lote de compras", "Mostrar todas las compras realizadas" y "Finalizar jornada".*

Continuamos con la sentencia de ciclo postcondicionado "**do – while** " donde el "**Do**" contiene "opcion = getch();" para que el usuario pueda acceder a opción del menú que desee por medio de la lista de números del 1 al 8, luego tenemos el "**While**" que contiene *"!*(opcion > '0' && opcion < '10') && (opcion != 27)" esto lo pensamos para que el cliente tuviera dos "caminos", uno de estos es elegir una opción entre el 1 y el 9, y el otro camino es salir del programa con la tecla ESC.

Para finalizar, creamos una sentencia de múltiples casos "***Switch***" con los respectivos casos referentes a cada opción del menú como se puede apreciar en la siguiente imagen.



2) Cargar clientes.

Para ese punto, primero creamos un "**struct**" llamado cliente cuyo contenido es:

```
11    struct cliente
12    {
13        char UsuarioID [10];
14        char email[40];
15        int dia;
16        int mes;
17        int anio;
18        bool borrado = false;
19    };
```

```
#define NOMBREARCHIVO "Clienteee.ayde"
#define NOMBRE "Procesados.ayde"
```

Luego, creamos un subprograma con la variable "**bool**" llamado "*guardarregistro*". dentro de este creamos el archivo que definimos previamente al inicio del programa como "*NOMBREARCHIVO*", continuamos invocando la sentencia de decisión sin caso complementario "**If**" cuya expresión lógica es "f=fopen(NOMBREARCHIVO,"ab") , esto significa que cuando el programa se ejecute y el usuario cargue un cliente nuevo, se abrirá un archivo de tipo "ab" (Permite la modificación de datos), llamado "*clienteee.ayde*". Dentro del "**If**" agregamos dos mensajes para el usuario a través de "**cout**", donde éste deberá ingresar su Usuario ID y su Email por medio de "**cin**".
Para concluir con este punto, declaramos un "**fwrite**" para que el usuario pueda escribir sus datos en el "**cin**" y finalizamos haciendo un "**fclose**", para cerrar el archivo.

3) Búsqueda de clientes

En este punto decidimos crear dos subprogramas, uno para la búsqueda de por Usuario ID y otro para la búsqueda por Email, los contenidos son los siguientes:

```
127  int buscar(char UsuarioID[])
128  {
129      FILE *f;
130      int encontrado =0;
131      struct cliente p;
132      if (f=fopen(NOMBREARCHIVO,"rb"))
133      {
134          while(fread(&p,sizeof(struct cliente),1,f) && !encontrado)
135          {
136              if (strcmp(UsuarioID, p.UsuarioID)==0)
137              {
138                  encontrado = 1;
139                  cout << "****** Datos del cliente *******" << endl;
140                  cout << "UsuarioID " << p.UsuarioID << endl;
141                  cout << "Fecha de creacion: " << p.anio << "/" << p.mes << "/" <<p.dia << endl;
142                  if(p.borrado == 1) {  cout << "Activo:  True " << endl;}
143                  if(p.borrado == 0) {  cout << "Activo: False " << endl;}
144                  cout << "Email: " << p.email << endl;
145                  cout << "Total de importe de compras: " << importeTotal (p.UsuarioID) << endl;
146              }
147              if (strcmp(UsuarioID, p.UsuarioID)==1)
148              {
149                  cout << "No se encontro el cliente buscado" << endl;
150              }
151          }
152          fclose(f);
153          return 1;
154      }
155      return 0;
156  }
```

Luego de abrir el archivo previamente definido al inicio del programa, creamos una sentencia repetitiva de ciclo precondicionado "**While**" cuya expresión lógica es primero leer el archivo y luego que encontrado sea == 0.

Aquí indicamos que si la variable "Usuario ID" ==0 , entrara en la secuencia de decisión sin caso complementario "**If**" donde el usuario se encontrara con sus datos personales y de compra.

En el caso de que el Usuario ID sea ==1, se mostrara por medio de una sentencia externa de salida el mensaje "No se encontró el cliente buscado"

Ocurre lo mismo con el subprograma que busca a los clientes por Email, pero con las siguientes diferencias:

```
int buscarr(char email[])
{
    FILE *f;
    int encontradoo =0;
    struct cliente p;
    if (f=fopen(NOMBREARCHIVO,"rb"))
    {
        while(fread(&p,sizeof(struct cliente),1,f) && !encontradoo)
        {
            if (strcmp(email, p.email)==0)
            {
                encontradoo = 1;
                cout << "****** Datos del cliente *******" << endl;
                cout << "UsuarioID " << p.UsuarioID << endl;
                cout << "Fecha de creacion: " << p.anio << "/" << p.mes << "/" <<p.dia << endl;
                if(p.borrado == 1) {  cout << "Activo: True " << endl;}
                if(p.borrado == 0) {  cout << "Activo: False " << endl;}
                cout << "Email: " << p.email << endl;
                cout << "Total de importe de compras: " << importeTotal (p.UsuarioID) << endl;
            }
            if (strcmp(email, p.email)==1)
            {
                cout << "No se encontro el cliente buscado" << endl;
            }
        }
        fclose(f);
        return 1;
    }
    return 0;
}
```

En este usamos la variable "**Char email []**" del **struct** *cliente* porque lo que necesitamos es buscar al cliente por el mail

La variable del subprograma; "**int** *encontradoo*" parece ser la misma que utilizamos en el subprograma para buscar por Usuario ID pero al llevar una letra **o** de más en el nombre, es una variable totalmente distinta

Para concluir con este punto las dos opciones, tanto la búsqueda por UsuarioID como la de Email, las "unimos" al "case '4'", brindándole al usuario un mini menú donde podrá elegir por cual método buscar.

```
382         case '4':
383
384             cout << "Seleccionar metodo de busqueda:\n" <<endl;
385             cout << "1-Busqueda por UsuarioID.\n"<< endl;
386             cout << "2-Busqueda por email.\n"<< endl;
387             cin >> a;
388
389             if (a==1)
390             {
391                 printf ("Escriba el UsuarioID que desea encontrar: ");
392                 scanf("%s", UsuarioIDbuscado);
393             }
394             if (a==2)
395             {
396                 printf ("Escriba el Email que desea encontrar: ");
397                 scanf("%s",emailbuscado);
398             }
399             if ( a>2 || a<1 )
400             {
401                cout << "Valor ingresado incorrecto. Vuelve a seleccionar una opcion" << endl;
402             }
403
404             if ((!buscar(UsuarioIDbuscado)) ||  (!buscarr(emailbuscado)))
405                 printf("Error al intentar abrir el archivo, no hay ningun cliente cargado.\n");
406
407         break;
```

Aquí pensamos que debíamos crear una variable (la cual llamamos "**a**") que tuviera la función de seleccionar la opción dependiendo de los intereses del usuario, por eso creamos 4 sentencias de decisión sin caso complementario, "**If**":

1) El primer "**if**" tiene como condición (a==1), lo que sería la búsqueda por UsuarioID.

2) El segundo "**If**" tiene como condición (a==2), lo que sería la búsqueda por Email.

3) El siguiente "**If**" contiene la condición (a>2 || a<1), por lo tanto, le impide al cliente ingresar cualquier "**a**" que no sea "1" o "2"

4) Y el último, el "**If**" contiene la condición ((!buscar(UsuarioIDbuscado)) || (!buscarr(emailbuscado)) , este "**if**" es muy importante, debido a que si el usuario intenta buscar un cliente que no existe (tanto por email como por UsuarioID) el programa le devolverá un Error

4) Borrar clientes.

Para este punto lo que planteamos fue lo siguiente:

I. Creamos un subprograma con una variable "**int**" llamada "*borrar*" cuyo parámetro es la cadena de caracteres "**char**" llamada "*UsuarioID [ ]"*

II. Dentro de este subprograma creamos el archivo utilizado previamente (*NOMBREARCHIVO*), creamos una variable de tipo "**int**" llamada "*encontrado*" la igualamos a 0, también utilizamos el **struct** de *cliente*.

III. Establecimos una secuencia de decisión sin caso complementario "**If**" la cual tiene como expresión lógica abrir el archivo que creamos. Este archivo será de tipo rb+, lo que nos permitirá escribir, leer y lo más importante, actualizarlo.

IV. Dentro del "**If**" creamos un "**While**" con la siguiente condición: "!encontrado && fread(&p,sizeof(cliente),1,f" lo que quisimos lograr con esto es que el Usuario borre un cliente previamente ingresado, para ello pusimos ésta restricción, es decir, mientras que encontrado sea falso y el cliente registro del cliente pertenezca al archivo "*clienteee.ayed*" se va a ejecutar el "**If**" que está dentro de este "**while**".

V. Dentro del "**While**" creamos un "**If**". Aquí nosotros planteamos la siguiente situación:

- Hicimos un "**fseek** *(f, (-1)*sizeof(cliente), SEEK_CUR)*" esto lo pensamos de la siguiente forma: Para poder sobrescribir los datos del cliente que queremos desactivar necesitamos reposicionar el cursor, porque en el "**while**" anterior, el "**fread**" empezó a leer todos los Usuarios ID y detuvo su cursor al final del cliente que el Usuario está buscando.

- Llegamos a la conclusión de que debíamos multiplicar (-1) por la cantidad de bytes de "*cliente*" para los bytes resultantes retrocedan después el punto de referencia "***Seek_Cur***" es decir, desde la posición actual. Al retroceder, el cursor queda posicionado al comienzo del registro que el usuario desea borrar.

- A continuación, utilizamos "**fwrite**" para que se sobrescriban los mismos datos de **struct** UsuarioID (nuestro buffer) pero con la diferencia de que ahora se muestra un mensaje a través de una sentencia de tipo de salida "**cout**", diciendo que el cliente fue desactivado.

VI. Para finalizar este subprograma creamos otra sentencia de tipo "**If**" indicando que si en esa búsqueda realizada del cliente a borrar, no se encuentra la persona que buscamos, saldrá un mensaje a través de una sentencia de salida "**Cout**", informándole al usuario que no se a encontrado al cliente.

```
189  int borrar (char UsuarioID[])
190  {
191      FILE *f;
192      int encontrado =0;
193      cliente p;
194      if (f=fopen(NOMBREARCHIVO,"rb+"))
195      {
196          while(!encontrado && fread(&p,sizeof(cliente),1,f))
197          {
198              if (strcmp(UsuarioID, p.UsuarioID)==0)
199              {
200                  encontrado = 1;
201                  p.borrado = 0;
202                  fseek(f,(-1)*sizeof(cliente), SEEK_CUR);
203                  fwrite(&p,sizeof(cliente),1,f);
204                  cout << "El cliente ha sido desactivado satisfactoriamente" << endl;
205              }
206          }
207          if (!encontrado)
208              cout << "No se encontro el cliente a desactivar" << endl;
209          fclose(f);
210          return 1;
211      }
212      return 0;
213  }
```

Misma variable de tipo "**Char**" ahora dentro del parámetro de borrar

```
368          case '3':
369              cout << "Ingrese el UsuarioID del cliente a desactivar:" << endl;
370              cin >> UsuarioIDbuscado;
371              borrar(UsuarioIDbuscado);
372
373          break;
```

Variable de tipo "Char" creada al inicio del subprograma "**Int** *menú*"

Aquí está la opción borrar del menú, indicada como case '3' de la sentencia repetitiva de ciclo pre condicionado "**while**", con lo hablado previamente.

5) Listar clientes.

Para comenzar con este punto, explicaremos el porqué de la creación de la variable "**Int**" llamada "*importeTotal*" a partir del **struct** "*compra*":

```
22   struct compra
23   {
24       char CompraID [10];
25       char UsuarioIDD [10];
26       int diaa;
27       int mess;
28       int anioo;
29       int horass;
30       int minutoss;
31       int monto;
32       int NroArticulo;
33   };
```

Para llevar a cabo este ítem, creamos una variable de tipo "**Int**" la cual llamamos "*Importe Total*" y seguimos los siguientes pasos:

I. Como siempre creamos el archivo, y aclaramos que "**struct**" vamos a usar, en este caso es el de "*Compra*"

II. Creamos una sentencia "**If**" cuya condición es abrir el archivo "*Procesados.ayde*" de tipo "Rb" lo que nos permite leer datos.

III. Dentro del anterior "**If**" incorporamos una variable "**Int**" llamada "*montoTemp*" y indicamos que el archivo debe leer el buffer de compras

IV. Continuamos agregando un "**While**" para que lea todos los datos del archivo cuya condición es !feof(f), este nos permite saber si queda algún carácter más que leer en el archivo.

V. Dentro de la sentencia "**While**" agregamos otro "**If**". Lo que contiene el "**If**" significa lo siguiente: Si la comparación entre cadena de caracteres es la misma, entonces "UsuarioIDD ==0" por lo tanto el "*montoTemp*" se suma con "p.*monto*" creando así el importe total.

```
37  int importeTotal (char UsuarioIDD[])
38  {
39      FILE *f;
40      struct compra p;
41      if (f=fopen(NOMBRE,"rb"))
42      {
43          int montoTemp;
44          fread(&p,sizeof(struct compra),1,f);
45          while(!feof(f))
46          {
47              if (strcmp(UsuarioIDD, p.UsuarioIDD)==0)
48              {
49                  montoTemp += p.monto;
50              }
51              fread(&p,sizeof(struct compra),1,f);
52          }
53          fclose(f);
54          return montoTemp;
55      }
56      return 0;
57  }
```

Una vez que ya tenemos este subprograma fundamental, creamos otro cuya función es listar los clientes:

I. Comenzamos nuevamente creando el archivo, luego especificamos el "**struct**" que vamos a usar, en este caso es "*Cliente".*

II. Creamos la sentencia "**If**" la cual se encarga de abrir el archivo "*Clienteee.ayde*" de tipo lectura.

III. Podemos observar en la imagen que, luego del "**If**" mencionado anteriormente, viene una sentencia de tipo "**While**" cuya condición es leer el buffer del archivo.

IV. Próximamente podemos apreciar como dentro del "**While**" nos escontramos otra sentencia "**If**" la cual tendra la funcion de comprovar en que estado esta el cliente, activo o inactivo; en caso de que este activo, o sea: "p.borrado == 1", entrara al "**If**" en cuestion. Este mostrara los datos del cliente y además, el "*totaldeImporte*" (visto en el subprograma explicado anteriormente).

V. En caso de que "*p.borrado*" sea "== 0", es decir, el cliente esta inctivo o es inexistente y mostrara por mantalla a traves de un "**cout**" que no se encontró el cliente buscado.

```
int listarclientes()
{
    FILE *f;
    cliente p;
    if (f=fopen(NOMBREARCHIVO,"rb"))
    {
        while(fread(&p,sizeof(struct cliente),1,f))
        {
            if (p.borrado == 1)
            {
                cout << "**********Datos del cliente**********" << endl;
                cout << "Usuario ID: " << p.UsuarioID << endl;
                cout << "Fecha de creacion: " << p.anio << "/" << p.mes << "/" <<p.dia << endl;
                if(p.borrado == 1) {  cout << "Activo: True " << endl;}
                if(p.borrado == 0) {  cout << "Activo: False " << endl;}
                cout << "Email: " << p.email << endl;
                cout << "Total de importe de compras: " << importeTotal (p.UsuarioID) << endl;
            }
            if (p.borrado == 0)
            {
                cout << "No se encontro el cliente buscado" << endl;
            }
        }
        fclose(f);
        return 1;
    }
    return 0;
}
```

Después, creamos esta opción en el "**Case'5'**" de la sentencia "**Switch**". Al elegir esa opcion mostrara a todos los clientes listados.

En caso de que listar clientes sea "= False", el programa mostrara por pantalla, a traves de una sentencia externa de salida "**Cout**", "No hay clientes cargados en el archivo".

```
        case '5':
            if (!listarclientes())
            {
                cout << "No hay clientes cargados en el archivo" << endl;
            }
        break;
```

6)Procesar lote de compras.

Para llevar a cabo este ítem, creamos un subprograma cuya variable es "**Bool**" y se llama "*guardarRegistroDeCompra*". Los pasos que seguimos son los siguientes:

I. Como siempre creamos el archivo, y aclaramos que "**struct**" vamos a usar, en este caso es el de "*Compra*"

II. Creamos una sentencia "**If**" cuya condición es abrir el archivo "*Procesados.ayde*" de tipo "Ab" lo que nos permite agregar datos.

III. Dentro de este "**If**" encontramos continuas sentencias de asignación externa, tanto de salida como, de entrada "**Cout**" y "**Cin**". Lo que quisimos hacer con este subprograma fue darle al Usuario las opciones para que ingrese sus datos de compra, tales como "Ingrese UsuarioID", "Ingrese CompraID", "Ingrese Monto" e "Ingrese NroArticulo" y queden guardados en el archivo correspondiente.
También agregamos la fecha y la hora y como se puede apreciar en la imagen.

```
204   bool guardarregistrodecompra ()
205   {
206       FILE *f;
207       compra p;
208       if (f=fopen(NOMBRE,"ab"))
209       {
210           cout << "Ingrese UsuarioID:" << endl;
211           cin >>  p.UsuarioIDD;
212           cout << "Ingrese CompraID:" << endl;
213           cin >> p.CompraID;
214           cout << "Ingrese Monto:" << endl;
215           cin >> p.monto;
216           cout << "Ingrese NroArticulo:" << endl;
217           cin >> p.NroArticulo;
218
219
220           time_t tiempoahora;
221           time(&tiempoahora);
222           struct tm *mitiempoo =localtime (&tiempoahora
223           int anioo;
224           int mess;
225           int diaa;
226           int minutoss;
227           int horass;
228
229           diaa = mitiempoo -> tm_mday;
230           mess = mitiempoo -> tm_mon+1;
231           anioo = mitiempoo-> tm_year+1900;
232           horass= mitiempoo -> tm_hour;
233           minutoss= mitiempoo -> tm_min;
234
235           p.anioo = anioo;
236           p.mess = mess;
237           p.diaa = diaa;
238           p.horass = horass;
239           p.minutoss = minutoss;
240
241           fwrite(&p, sizeof(compra),1,f);
242
243           fclose(f);
244           return true;
245       }
246       return false;
247   }
```

IV. Para concluir, especificamos con un "**fwrite**" que los datos de escritura ingresados anteriormente se guarden en el archivo.

```
410           case '6':
411               if (guardarregistrodecompra())
412                   printf("El lote de compra de ha procesado exitosamente.\n");
413               else
414                   printf("Error al intentar abrir el archivo.\n");
415           break;
```

Desde luego esto lo podemos encontrar en el "**Case '6'**" de la sentencia "**Switch**" del menú.
Aquí agregamos una sentencia de decisión con caso complementario "**If – Else**" cuya función la interpretamos de este modo: El usuario debe ingresar los datos que se piden, al terminar se muestra por pantalla que el "el lote de compra se ha procesado exitosamente".

7) Mostrar compras del usuario.

Para este ítem, consideramos crear un subprograma a partir de una variable "**Int**", la cual llamamos "*Buscarlo*" y seguimos los siguientes pasos:

I. Como siempre creamos el archivo, y aclaramos que "**struct**" vamos a usar, en este caso es el de "*Compra*", También creamos una variable de tipo "**Int**" la cual la llamamos "*Encontrador*" y la igualamos a 0.

II. Añadimos una sentencia "**If**" cuya condición es abrir el archivo "*Procesados.ayde*" de tipo "Rb" lo que nos permite leer los datos agregados con anterioridad.

III. Dentro del anterior "**If**" agregamos un "**While**" para que lea todos los datos del archivo. Incluimos **"!feof(f)"** luego de una primera instancia, donde el programa mostraba solo la primera compra realizada por un usuario y no todas. En cambio una vez agregado el **"!feof(f)"** logramos que aparecieran todas las compras realizadas por el UsuarioID buscado previamente.

IV. Dentro del "**While**" anterior agregamos otro "**If**" indicando que si el "UsuarioIDD es == 0", el "Encontrador == 1" y le deberían aparecer por pantalla al Usuario los datos de la compra.

V. En caso de que "*Encontrador*" sea "==0", se mostrara por pantalla "*El UsuarioID no tiene ninguna compra realizada*"

```
249  int buscarlo (char UsuarioIDD[])
250  {
251      FILE *f;
252      int encontrador =0;
253      struct compra p;
254      if (f=fopen(NOMBRE,"rb"))
255      {
256          while(fread(&p,sizeof(struct compra),1,f),!feof(f))
257          {
258              if (strcmp(UsuarioIDD, p.UsuarioIDD)==0)
259              {
260              encontrador = 1;
261              cout << "****** Datos compra *******" << endl;
262              cout << "UsuarioID: " << p.UsuarioIDD << endl;
263              cout << "Fecha de creacion: " << p.anioo << "/" << p.mess << "/"  << p.diaa << " " << p.horass << ":" << p.minutoss << endl;
264              cout << "CompraID: " << p.CompraID << endl;
265              cout << "Monto: " << p.monto<< endl;
266              cout << "Nro Articulo: " << p.NroArticulo<< endl;
267              }
268          }
269          if (!encontrador)
270              printf("El UsuarioID no tiene ninguna compra realizada.\n");
271          fclose(f);
272          return 1;
273      }
274      return 0;
275  }
```

Por supuesto que esto lo podemos encontrar en el "**Case '7'**" de la sentencia "**Switch**" del menú.

```
418          case '7':
419              printf("Ingrese el UsuarioID buscado.\n");
420              scanf("%s", UsuarioIDbuscador);
421              if (!buscarlo(UsuarioIDbuscador))
422                  printf("Error al intentar abrir el archivo.\n");
423          break;
424
```

Al presionar la tecla del número 7, aparecerá por pantalla "*Ingrese el UsuarioID buscado*", en caso de que el UsuarioID tenga alguna compra realizada, se muestra. Por el contrario, se le indicara al usuario que no ha realizado ninguna compra.

8) Finalizar jornada.

Para este último ítem, consideramos crear un subprograma a partir de una variable "**Int**", la cual llamamos "*Finalizar Jornada*" y seguimos los siguientes pasos:

I. Creamos el archivo, y aclaramos que "**struct**" vamos a usar, en este caso es el de "*Cliente*".

II. Comenzamos creando una variable "**If**" la cual abre el archivo "*Clienteee.ayde*" que es de tipo "Rb" lo que nos permite leer los datos agregados con anterioridad.

III. Dentro del anterior "**If**" agregamos un "**While**" para que lea todos los datos del archivo.

IV. A continuación, agregamos un "**If**" con la condición "p.borrado == 1". En caso de serlo el programa actualizara el importe total.

```
308    int finalizarjornada()
309    {
310        FILE *f;
311        cliente p;
312        if (f=fopen(NOMBREARCHIVO,"rb"))
313        {
314            while(fread(&p,sizeof(struct cliente),1,f))
315            {
316                if (p.borrado == 1)
317                {
318                    importeTotal (p.UsuarioID);
319                }
320            }
321            fclose(f);
322            return 1;
323        }
324        return 0;
325    }
```

Desde luego esto lo podemos encontrar en el "**Case '8'**" de la sentencia "**Switch**" del menú.

```
425        case '8':
426            finalizarjornada();
427            cout << "Se actualizaron los saldos de las cuentas exitosamente" << endl;
428            exit(1);
429        break;
```

Al tocar el número 8, aparecerá por pantalla "*Se actualizaron los saldos de las cuentas exitosamente*" y dará al usuario la opción de salir del programa tacando cualquier tecla.

Diagramas de Lindsay

Diagrama del subprograma Buscar

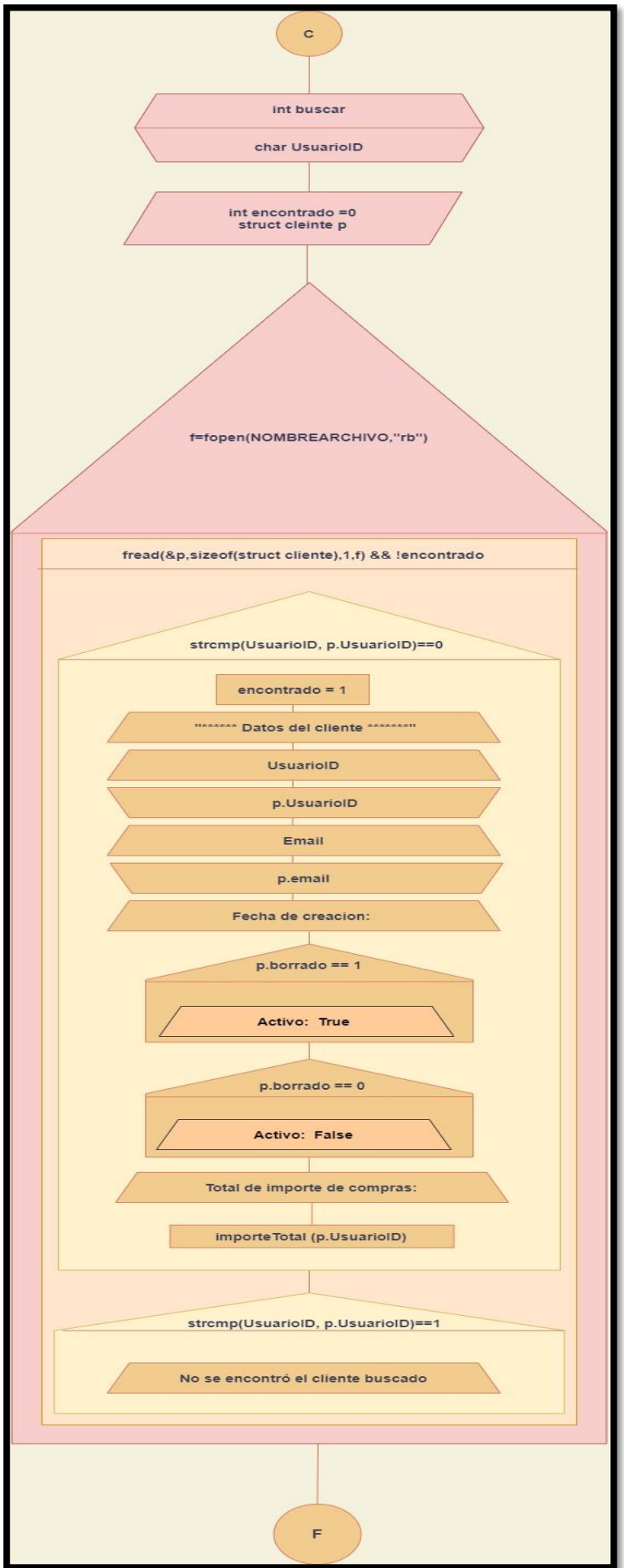


Diagrama del subprograma GuardarRegistro

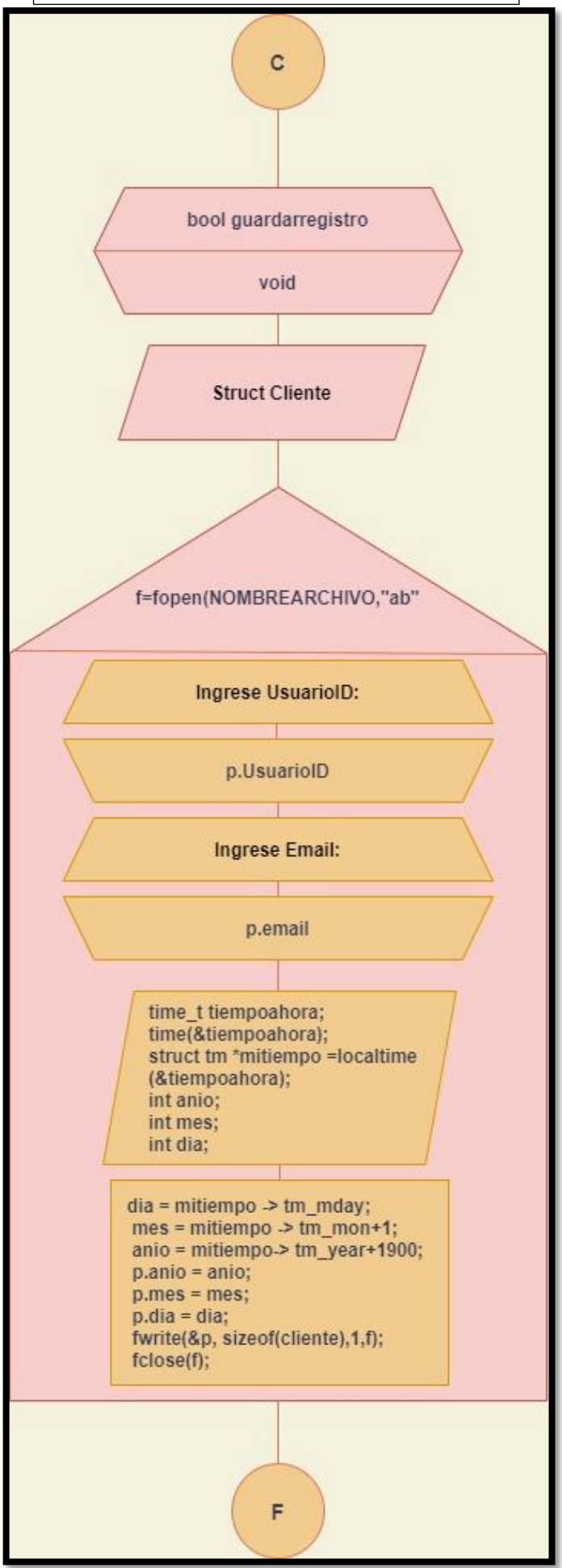


Diagrama del subprograma Borrar

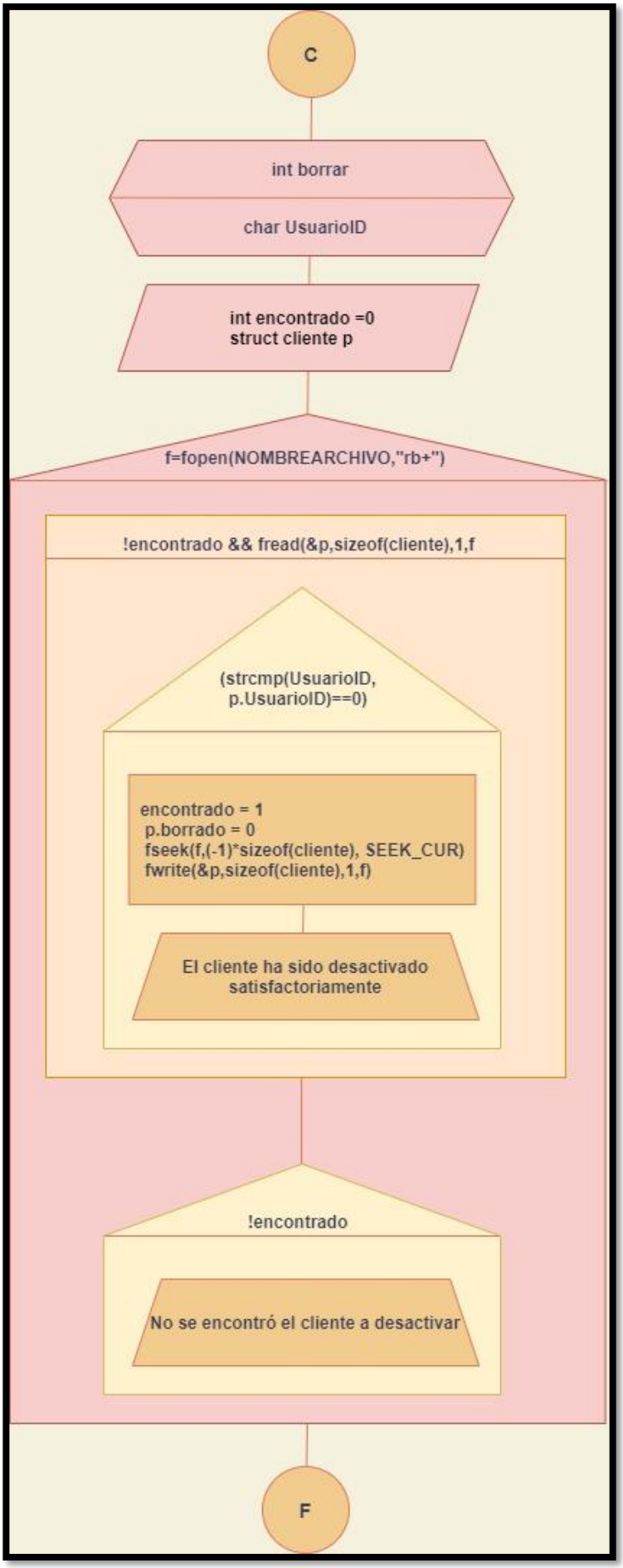


Diagrama del subprograma FinalizarJornada

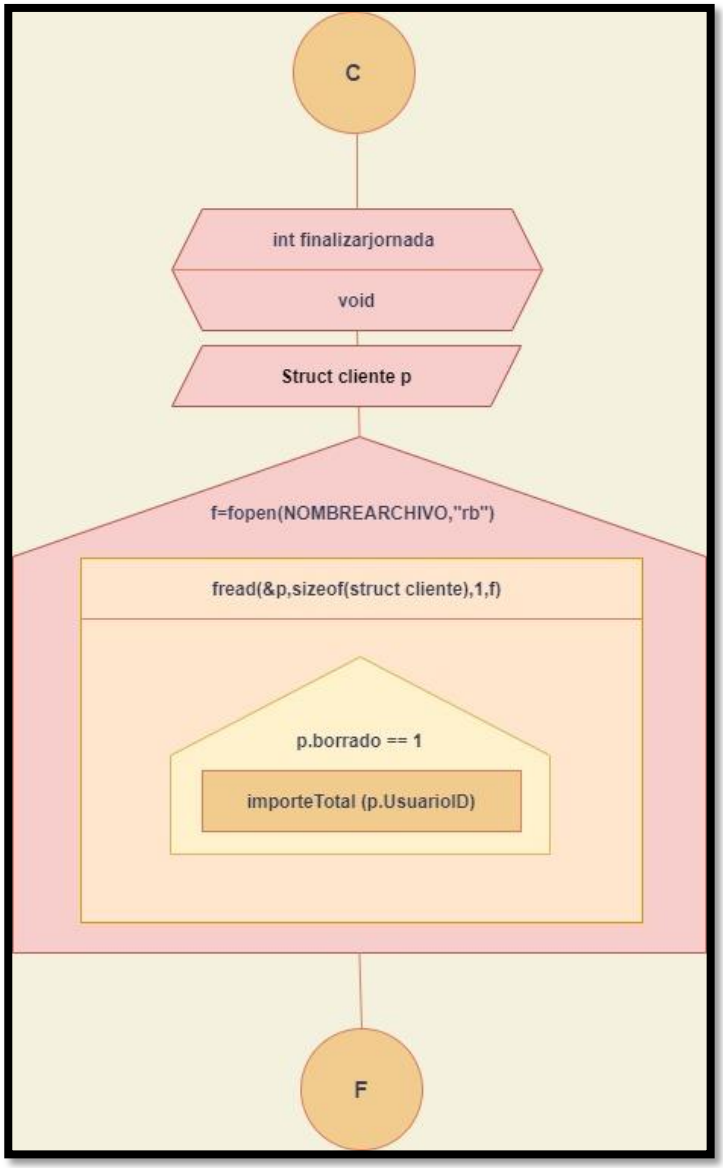


Diagrama del subprograma guardar registro decompra