UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

Profesor: Ernesto Alcántara Concepción

Fundamentos de Programación

Proyecto en lenguaje C

HABITUS

Integrantes:

- Arias Jiménez Alejandro
- García Olvera José Ignacio
- Gónzalez Falcón Luis Adrián
- López Morales Fernando Samuel
- Ortega Ortega Genaro Raziel

2024-1

ÍNDICE

DEFINICIÓN

Descripción del proyecto

Aplicación para manejar hábitos; compete el crear, gestionar y manejar cada uno de ellos para poder mejorar la productividad del usuario.

Justificación

Recae en los estudiantes de la universidad una responsabilidad de hacer todas sus tareas, proyectos y asignaciones para poder avanzar en la carrera y poder adquirir conocimiento. Por ello, es útil poder organizar todos sus hábitos.
Se propone una aplicación para poder manejar hábitos, para que el usuario pueda retenerlos todos en una aplicación y pueda gestionarlos, procurando siempre el cumplimiento de aquellos que fueron configurados para el día de la semana específico.

Dinámica y elección de rumbo del proyecto

El manejo de hábitos requiere del concepto de **CRUD** *(Create, Read, Update & Delete)* para el manejo de la información, por ello y en conjunto con el requisito obligatorio de ser un proyecto desarrollado en **lenguaje C**, se propone un modelo de manejo de archivos manipulados en forma binaria para poder almacenar toda información requerida por la aplicación.
El manejo de los hábitos se presta para la elaboración de una interfaz gráfica, por lo que, para poder manejar visualmente e interpretar la información de una manera más cómoda, se propone trabajar con la **biblioteca Allegro 5**, utilizada comúnmente para el desarrollo de videojuegos; esta biblioteca nos proveerá en esencia de la capacidad de manejar figuras primitivas para poder crear figuras complejas, eventos del teclado y temporizadores para la elaboración de una aplicación.

REQUISITOS PREVIOS

Herramientas

Para poder desarrollar una aplicación que utiliza recursos de la biblioteca Allegro 5, el equipo en conjunto se valió de los servicios gratuitos que proporciona ***JetBrains*** para la educación en asociación con ***Github for Education***, prestando sus servicios gratuitamente para comunidades estudiantiles, en este caso, de la UNAM, para poder utilizar gratuitamente el IDE CLion para compilar y manejar todo el proyecto.

Para la instalación de la biblioteca **Allegro 5** y del **IDE CLion**, así como información útil relacionada al proyecto y el proceso previo de trabajo, se proporciona el siguiente manual elaborado por un integrante del equipo:

Manual Instalaciones Allegro y Clion_Proyecto Final FP.pdf

REPOSITORIO

Se utilizaron también el manejador de versiones Git, en conjunto con un repositorio en línea en GitHub, con el que el equipo trabajó durante todo el proyecto:
https://github.com/TheUniqueFersi/Control-de-Habitos.git

ANÁLISIS

ENTRADA

Información de un hábito (Nombre del hábito, Notas del hábito, días de la semana que se repetirá, dificultad) ingresados mediante el teclado.

SALIDA

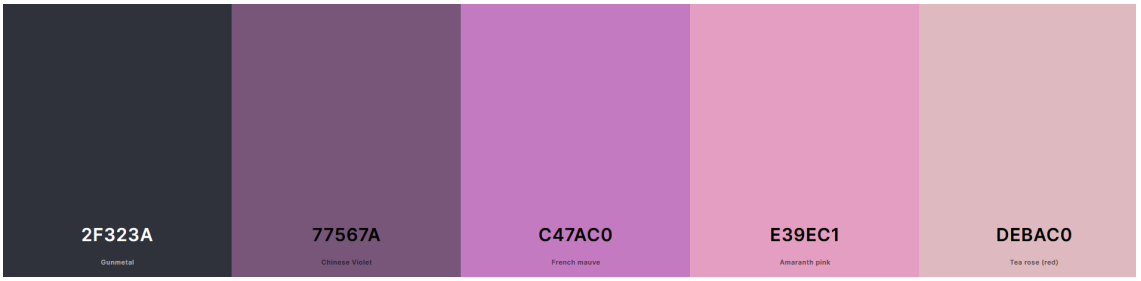
Hábito con información visualmente acomodada
Indicadores visuales que facilitan la información de cuantos hábitos se tiene que hacer ese día en específico
Indicadores visuales que marcan que días de la semana cuentan con más hábitos por hacer.

RESTRICCIONES

- Dificultades predefinidas por el programa
- No se cuenta con fecha límite para cada hábito, en caso de ya no requerirlo, el usuario se tiene que borrarlo.

DISEÑO

El equipo decidió en conjunto usar la siguiente paleta de colores:
https://coolors.co/2f323a-77567a-c47ac0-e39ec1-debac0

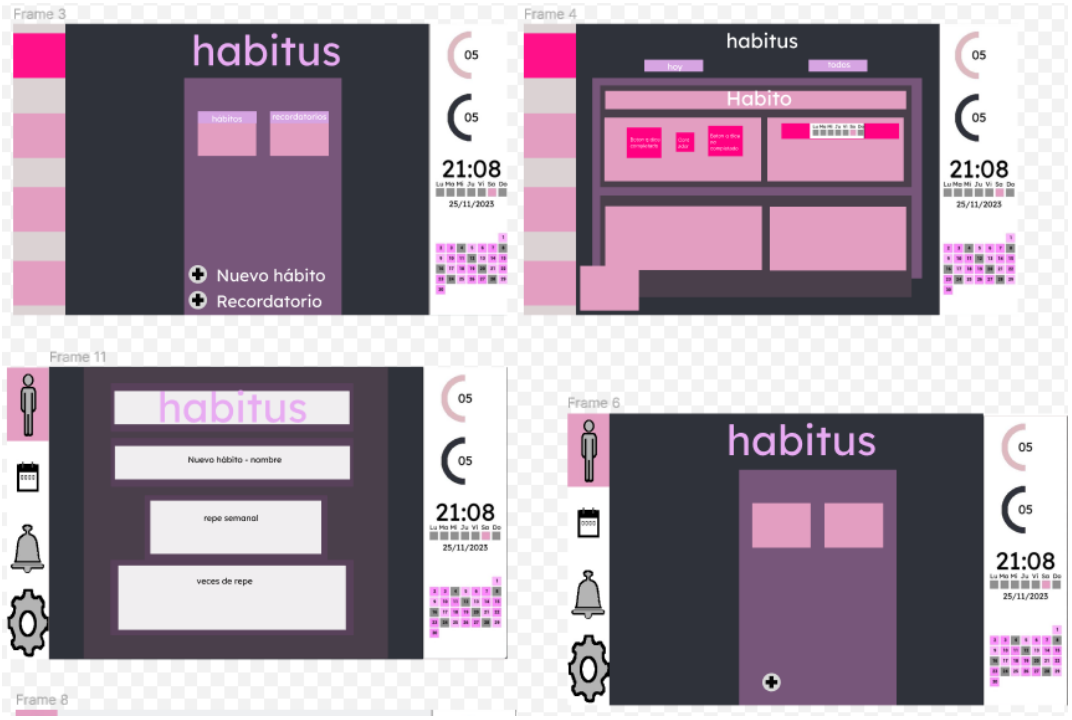


INTERFAZ

Se utilizó **Figma** para la elaboración de la Guía de estilos y las vistas principales que tendría la aplicación.

Se optó por utilizar las fuentes: **Lexend** para títulos; `Roboto` para textos.

HABITUS



CONTROLES

Esencialmente se cuenta con el **teclado** como manera de desplazarse entre los 2 menús de la aplicación:

Tecla 1: para ir al menú de manejo de hábitos

Tecla 4: para ir al menú de cambio de nombre de usuario

Dentro del menú 1 (hábitos), se pueden ver todos los hábitos desplazandote verticalmente, esto se logra con las **teclas de dirección arriba y abajo**.

Para crear un hábito, se presiona la **tecla A**, y se siguen las instrucciones, siempre accediendo al siguiente paso con **ENTER**.

Para eliminar un hábito, se enfoca el hábito y se presiona la **tecla B**, seguido de una confirmación que dará las indicaciones en caso de cancelar o confirmar eliminación.

El ingreso de textos es mediante el teclado.

ENTIDADES

Para poder diseñar la manera en como el código **recuperará, insertará, editará y eliminará** datos de los archivos de almacenamiento de información, se diseñaron varias entidades, de las cuáles terminaron siendo utilizadas completamente 3:

USUARIO	
PK	
ID_usuario	nombre
INT	CHAR(30)

DIFICULTAD	
PK	
ID_dificultad	dificultad
INT	CHAR(11)
1	Muy fácil
2	Fácil
3	Intermedio
4	Difícil
5	Muy difícil

HABITO								
PK					FK	FK		
ID_habito	nombre	nota	repeticion_semanal	repeticion	ID_tipo	ID_dificultad	racha	fecha_ini
INT	CHAR(40)	CHAR(180)	CHAR(7)	INT	INT	INT	INT	

ESTRUCTURA DEL PROYECTO (carpetas)

En el repositorio, se cuentan con 8 directorios, que se explicarán a continuación:

config: Carpeta para almacenar archivos de configuración de proyecto.

data: Carpeta donde se almacenan los archivos de información de la aplicación para su funcionamiento así como las “bases de datos” que guardarán la información del usuario y de sus hábitos correspondientes”.

docs: Carpeta de documentación del proyecto.

include: Carpeta que guarda archivos “.c” que tendrán código necesario para el funcionamiento de la aplicación; se separa del archivo principal de ejecución para organizar de mejor manera el proyecto y poder tener en archivos separados ciertas secciones o funciones a modo de bibliotecas personalizadas del equipo.

libs: Almacena todos los archivos de la biblioteca Allegro 5.

media: Almacena aspecto gráficos; fuentes de texto e imágenes.

src: Almacena el archivo fuente: “main.c”.

tests: Carpeta para hacer pruebas personales o grupales de tipo tests de depuración o búsqueda de errores.

Archivos en la carpeta raíz:

.dll : Necesarios para la ejecutción de un programa que utiliza la biblioteca Allegro 5

.gitignore : Configuración del repositorio par ignorar archivos

README.md: Documentación de proyecto

.exe (ejecutable): Ejecutable de la aplicación. (Cualquier dispositivo windows puede correr la aplicación, siempre y cuando el antivirus no lo reconozca como amenaza).

DIAGRAMA DE FLUJO

Dada la compleja estructura del proyecto, así como el uso de funciones propias de la biblioteca Allegro 5, el diagrama de flujo tiene una extensión grande.
A continuación se proporciona el link del mimo:

https://app.diagrams.net/#G1qK9I_KCsgkW0GWfPojT1I76yRdzcNJ_t

PSEUDOCÓDIGO

Dada la misma situación previamente mencionada, el pseudocódigo se resumió de manera que explicara aspectos muy claves del programa:

```
Funcion modoHabitos()
   //EVENTOS DE TECLADO PARA REGISTRAR
   Si estado=0 Entonces
         //Desplazarse por el menú
   SiNo
         Si estado=1 Entonces
               //Crear Habitos
         SiNo
               Si estado=2 Entonces
                     //EDITAR HABITO
               SiNo
                     Si estado=3
                           //ELIMINAR HABITO
                     Fin Si
               Fin Si
         Fin Si
   Fin Si
FinFuncion
Funcion main_habitus()
   pedirUsuario();//pide el nombre del usuario
   cargarRegistros();
```

```
   Mientras !ventanaCerrada Hacer
        activarEventos();//Se inicializan los evento para cada Caso
                                        //(eventos de teclado, de ventana, etc)
        Segun momento Hacer
            -1:
                depuracion(); //Depuracion de registros
            0:
                pedirNombreUsuario();//Registrar el nombre de usuario
            1:
                modoHabitos(); // Gestor de habitos
            2:
                horarios();//GESTOR HORARIOS
            3:
                recoradtorios();//GESTOR RECORDATORIOS
            4:
                ajustes();
            De Otro Modo:
        Fin Segun
   Fin Mientras

Fin Funcion


Funcion inicializarVentanas()
   // ... CÓDIGO DONDE SE INICIALIZA VENTANAS Y EVENTOS
   main_habitus();
   Escribir "Inicializar ventana";
Fin Funcion

Algoritmo control_de_Habitos
   Si inicializaAllegro Entonces
        inicializarVentanas();//DESCRIPCION
   SiNo
        Escribir "Ocurrió un error";
   Fin Si
FinAlgoritmo
```

CÓDIGO

Dada la "***ESTRUCTURA DEL PROYECTO (carpetas)***" previamente explicada, se presentan los archivos y el código con el que el programa funciona

** Salvo las funciones alusivas a Allegro, todo el código a continuación y sus funciones fueron elaboradas por el equipo.*

Archivo ***`include\structs.c`***

Define las estructuras que se utilizarán a lo largo del proyecto (algunas estructuras fueron descartadas dado el tiempo de desarrollo, pero se conservan para mantener un área de oportunidad y de desarrollo posterior)

```c
#include <time.h>
typedef struct tm FECHA;
typedef struct {
	int ID_dificultad;
	char dificultad[11];
} DIFICULTAD;
typedef struct {
   int ID_usuario;
   char nombre[30];
} USUARIO;
typedef struct {
   int ID_tipo;
   char tipo[50];
} TIPO;
typedef struct {
   int ID_habito;
   char nombre[40];
   char nota[180];
   char repeticion_semanal[7];
   int repeticion;
   TIPO *ptr_fk_tipo;
   TIPO fk_tipo;        //-----
   DIFICULTAD *ptr_fk_difi;
   DIFICULTAD fk_difi; //-----
   int racha;
   time_t tiempo;
   FECHA fecha_ini;
} HABITO;
```

```
typedef struct {
   int ID_RH;
   HABITO *ptr_fk_habito;
   HABITO fk_habito; //------
   time_t tiempo;
   FECHA fecha;
   int completado;
   int no_completado;
} REGISTRO_HABITOS;
typedef struct {
   int ID_horario;
   char nombre[40];
   char repeticion_semanal[7];
   TIPO *ptr_fk_tipo;
   TIPO fk_tipo; //-----
   time_t tiempo;
   FECHA fecha_ini;
   FECHA fecha_final;
   FECHA alerta;
} HORARIO;
typedef struct {
   int ID_HH;
   HORARIO *ptr_fk_horario;
   HORARIO fk_horario; //-------
   time_t tiempo;
   FECHA dia_h_ini;
   FECHA h_final;
} HORA_HORARIO;
typedef struct {
   int ID_recordatorio;
   char recordatorio[180];
   TIPO *ptr_fk_tipo;
   TIPO fk_tipo; //-------
   time_t tiempo;
   FECHA fecha;
   short int estado_comp;
} RECORDATORIOS;
typedef struct {
   int ID_product;
   time_t tiempo;
   FECHA fecha;
```

```
    int habit;
    int racord;
} PRODUCTIVIDAD;
```

Archivo ***include\CRUD.c***

Contiene las funciones de "***SELECT; INSERT, SUPER_INSERT; UPDATE & DELETE***", así como otras funciones dirigidas al manejo de los archivos *.dat* para el almacenamiento de información.

```
/* COMANDOS PARA MANEJAR CRUD EN LOS ARCHIVOS */
#include <stdio.h>
#include <string.h>
#include "structs.c"
/*OTRAS COSAS*/

int contadorBytesArch(char *ruta){
    FILE *archi = fopen(ruta, "rb");
    int retorno = 0, n_bytes;
    if(archi != NULL){
        fseek(archi, 0, SEEK_END);
        n_bytes = ftell(archi);
        //printf("Numero de bytes de %s: %i\n",ruta, n_bytes);
        rewind(archi);
        fclose(archi);
        retorno = n_bytes;
    }
    else{
        printf("El archivo %s no fue encontrado (contadosBytesArch)\n", ruta);
        retorno = -1; //ERROR
    }
    return retorno;
}
int desplazarAUTOINCREMENT(FILE *arch){ //requiere de la apertura en el modo:
rb ||
    int variableDestino=0, retorno;
    fseek(arch, 0, SEEK_SET);
    fread(&variableDestino, sizeof(int), 1, arch);
    //printf("-- AUTO_INCREMENT leido: %i --\n", variableDestino);
    retorno = (variableDestino > 0)? variableDestino: -1;
```

```
    return retorno;
}

int INSERT(char *, void *, size_t, size_t);
int SELECT(char *, void *, size_t, size_t, int id);
int UPDATE(char *, void *, size_t, size_t, int id);
int DELETE(char *, void *, size_t, size_t, int id);

int manejarAUTOINCREMENT(char *ruta){//Solamente ejecutable dentro de INSERT,
    int AUTO_INCREMENT=0;
    FILE *arch = fopen(ruta, "ab+");
    if(arch != NULL){
        if(contadorBytesArch(ruta)==0){
            printf("\tEl archivo esta vacío\n");
            AUTO_INCREMENT=1;
            fwrite(&AUTO_INCREMENT, sizeof(int), 1, arch);
        }
        else{
            fclose(arch);
            FILE *archi2 = fopen(ruta, "rb+");
            //rewind(arch),
            AUTO_INCREMENT = desplazarAUTOINCREMENT(archi2);
            if(AUTO_INCREMENT > 0){
                AUTO_INCREMENT++;
                rewind(archi2);
                fwrite(&AUTO_INCREMENT, sizeof(int), 1, archi2);
                fclose(archi2);
            }
            else{
                printf("Ocurrio un error en la devolucion, AUTOINCREM: %i:
\n", AUTO_INCREMENT);
            }
        }
        fclose(arch);
    }
    else{
        printf("OCurrio un error (manejarAUTOINCREMENT)\n");
    }
    //printf("Nuevo Autoincremento: %i\n", AUTO_INCREMENT);
    return AUTO_INCREMENT;
}
```

```
int SUPER_INSERT(int * ID, char *ruta, void *registro, size_t tam_elem,
size_t num_elem){
   *ID = manejarAUTOINCREMENT(ruta);
   int nuevoID = *ID;
   INSERT(ruta, registro, tam_elem, num_elem);
   return nuevoID;
}
int INSERT(char *ruta, void *registro, size_t tam_elem, size_t num_elem){
   int retorno;
   if(strcmp("./data/app.dat", ruta)==0){
       //printf("ENTRE A APP.DAt");
       FILE *arch = fopen(ruta, "wb");
       if(arch!=NULL){
           fwrite(registro, tam_elem, num_elem, arch);
           fclose(arch);
       }else{
           printf("ERROR al abrir archivo wb app.dat\n");
       }
   }
   else{
       FILE *arch = fopen(ruta, "rb+");
       if(arch!=NULL){
           fseek(arch, 0, SEEK_END);//Se posiciona al final de to_do registro
           //printf("POSICICON %li\t", ftell(arch));
           fwrite(registro, tam_elem, num_elem, arch);
           if(fclose(arch) != 0){
               printf("Hubo un problema cerrando %s\n", ruta);
           }
       }
       else{
           printf("La base de datos no existe (INSERT)\n");
       }
   }
   return retorno;
}
int SELECT(char *ruta, void *registro_en_codigo, size_t tam_elem, size_t
num_elem, int id){
   //printf("FuncionSELECT\n");
   if(strcmp("./data/app.dat", ruta)==0){
       //printf("ENTRE A APP.DAt");
       FILE *arch = fopen(ruta, "rb");
```

```
        if(arch!=NULL){
            rewind(arch);
            //printf("Antes de lectura: %li\t", ftell(archivo));
            fread(registro_en_codigo, tam_elem, num_elem, arch);
            fclose(arch);
        }else{
            printf("ERROR al abrir archivo wb app.dat\n");
        }
    }else{
        FILE *archivo = fopen(ruta, "rb");  //requiere de la apertura en el
modo: rb ||
        if(archivo != NULL){
            rewind(archivo);
            desplazarAUTOINCREMENT(archivo);
            fseek(archivo, (id - 1) * ((long)tam_elem), SEEK_CUR);
            //printf("Antes de lectura: %li\t", ftell(archivo));
            fread(registro_en_codigo, tam_elem, num_elem, archivo);
            fclose(archivo);
        }else
            printf("La base de datos %s no existe (SELECT)\n", ruta);
    }
    return 0;
}

int UPDATE(char *ruta, void *registro_act, size_t tam_elem, size_t num_elem,
int id){
    //printf("\nFuncion UPDATE\n");
    FILE *archivo = fopen(ruta, "rb+");
    if(archivo!=NULL ){
//       fseek(ptrCF, 0, SEEK_END);
        rewind(archivo);
        desplazarAUTOINCREMENT(archivo);
        fseek(archivo, (id-1)*((long)tam_elem), SEEK_CUR);
        //printf("\tUPDATE: %li\n", ftell(archivo));
        fwrite(registro_act, tam_elem, num_elem, archivo);
        fclose(archivo);
    }else{
        printf("La base de datos no existe (UPDATE)\n");
    }
}
```

```
int DELETE(char *ruta, void *registro_a_elim, size_t tam_elem, size_t
num_elem, int id){
   //printf("\nFuncionDELETE\n");
   FILE *archivo = fopen(ruta, "rb+");
   HABITO hab, habNULL={'\0'};/*SE CREA habNULL PARA SOBRESCRIBIR EL REGISTRO
QUE INDIQUE EL id*/
   if(archivo!=NULL ){
//      fseek(ptrCF, 0, SEEK_END);
       rewind(archivo);
       desplazarAUTOINCREMENT(archivo);
       fseek(archivo, (id-1)*((long)tam_elem), SEEK_CUR);
       //printf("\tUPDATE: %li\n", ftell(archivo));
       fwrite(&habNULL, tam_elem, num_elem, archivo);
       fclose(archivo);
   }else{
       printf("La base de datos no existe (DELETE)\n");
   }
   fclose(archivo);
}
// -- CONSIDERACIONES --
//INSERT INTO tabla VALUES (1,1,1,1); ** SOLO SE PUEDEN METER REGISTROS
COMPLETOS
//SELECT * FROM tabla;
//UPDATE tabla SET col = val WHERE columna = valor;
//DELETE FROM tabla WHERE col = valor;
```

Archivo ***include\resources.c***

Inicializa los recursos gráficos usando Allegro

```
#include <allegro5/allegro.h>
ALLEGRO_COLOR principal_pale_chestnut;
ALLEGRO_COLOR secundario_pastel_magenta;
ALLEGRO_COLOR neutro1_tinta_de_pulpo;
ALLEGRO_COLOR neutro2_african_violet;
ALLEGRO_COLOR neutro3_french_lilac;
ALLEGRO_COLOR texto_white;
ALLEGRO_COLOR texto_black;
ALLEGRO_COLOR fondo_gris1;
ALLEGRO_COLOR fondo_principal_oscuro;
```

```
ALLEGRO_FONT * lexend_regular[60];
ALLEGRO_FONT * lexend_bold[60];
ALLEGRO_FONT * lexend_thin[60];
ALLEGRO_FONT * roboto_bold[60];
ALLEGRO_FONT * roboto_italic[60];
ALLEGRO_FONT * roboto_regular[60];
ALLEGRO_FONT * roboto_thin[60];
ALLEGRO_FONT * roboto_black[60];

ALLEGRO_BITMAP *RECORDS;
ALLEGRO_BITMAP *HABITOS;
ALLEGRO_BITMAP *CALENDARIOBLANCO;
ALLEGRO_BITMAP *AJUSTES;

ALLEGRO_BITMAP *CALENDARIOROSA;
ALLEGRO_BITMAP *AJUSTESROSA;
ALLEGRO_BITMAP *RECORDSROSA;
ALLEGRO_BITMAP *HABITOSROSA;
ALLEGRO_BITMAP *LOGO;
ALLEGRO_BITMAP *EDITARHABITO;
ALLEGRO_BITMAP *BORRARHABITO;
ALLEGRO_BITMAP *NUEVOHABITO;
ALLEGRO_BITMAP *FLECHAS;
int init_resources(){
   int inicializado_correctamente=1;

   fondo_principal_oscuro= al_map_rgb(47, 50, 58);
   fondo_gris1 = al_map_rgb(143,143,143);
   principal_pale_chestnut = al_map_rgb(222, 186, 192);
   secundario_pastel_magenta = al_map_rgb(227,158,193);
   neutro1_tinta_de_pulpo = al_map_rgb(47,50,58);
   neutro2_african_violet = al_map_rgb(222, 186, 192);

   neutro3_french_lilac = al_map_rgb(222, 186, 192);
   //neutro3_french_lilac = al_map_rgb(56, 38, 56);
   texto_white = al_map_rgb(255,255,255);
   texto_black = al_map_rgb(0,0,0);

   for(int i=0; i<60; i++) {
       lexend_regular[i] =
al_load_font("./media/fuentes/lexend/Lexend-Regular.ttf", i, 0);
```

```
       lexend_bold[i] =
al_load_font("./media/fuentes/lexend/Lexend-Bold.ttf", i, 0);
       lexend_thin[i] =
al_load_font("./media/fuentes/lexend/Lexend-Thin.ttf", i, 0);
       roboto_bold[i] =
al_load_font("./media/fuentes/roboto/Roboto-Bold.ttf", i, 0);
       roboto_italic[i] =
al_load_font("./media/fuentes/roboto/Roboto-Italic.ttf", i, 0);
       roboto_regular[i] =
al_load_font("./media/fuentes/roboto/Roboto-Regular.ttf", i, 0);
       roboto_thin[i] =
al_load_font("./media/fuentes/roboto/Roboto-Thin.ttf", i, 0);
       roboto_black[i] =
al_load_font("./media/fuentes/roboto/Roboto-Black.ttf", i, 0);

       if(!lexend_regular[i] || !lexend_bold[i] || !lexend_thin[i]){
           inicializado_correctamente = 0;
           printf("ERROR: %i\n", al_get_errno());
           printf("ERROR: Hubo un problema al cargar las fuentes de
lexend\n");
       }
       if(!roboto_regular[i] || !roboto_black[i] || !roboto_bold[i] ||
!roboto_italic[i] || !roboto_thin[i]){
           inicializado_correctamente = 0;
           printf("ERROR: %i\n", al_get_errno());
           printf("ERROR: Hubo un problema al cargar las fuentes de
roboto\n");
       }
   }
   RECORDS = al_load_bitmap("./media/img/recordatorio_blanco.jpg");
   HABITOS = al_load_bitmap("./media/img/habitosblanco.jpg");
   CALENDARIOBLANCO = al_load_bitmap("./media/img/cal_blanco.jpg");
   AJUSTES = al_load_bitmap("./media/img/ajustes_blanco.jpg");

   CALENDARIOROSA = al_load_bitmap("./media/img/cal_rosa.jpg");
   RECORDSROSA = al_load_bitmap("./media/img/recordatorio_rosa.jpg");
   HABITOSROSA = al_load_bitmap("./media/img/habitosrosa.jpg");
   AJUSTESROSA = al_load_bitmap("./media/img/ajustes_rosa.jpg");
   HABITOSROSA = al_load_bitmap("./media/img/habitosrosa.jpg");
   //HABITOSROSA = al_load_bitmap("./media/img/habitosrosa.jpg");
   LOGO = al_load_bitmap("./media/img/logo.png");
```

```
   EDITARHABITO = al_load_bitmap("./media/img/editar.jpg");
   BORRARHABITO = al_load_bitmap("./media/img/borrar.jpg");
   NUEVOHABITO = al_load_bitmap("./media/img/nuevo.png");
   FLECHAS = al_load_bitmap("./media/img/flechas2.png");

   if(!RECORDS || !HABITOS || !CALENDARIOBLANCO || !AJUSTES ||
!CALENDARIOROSA || !RECORDSROSA || !HABITOSROSA || !AJUSTESROSA || !LOGO ||
!NUEVOHABITO || !BORRARHABITO || !EDITARHABITO || !FLECHAS){
       inicializado_correctamente = 0;
       printf("ERROR: %i\n", al_get_errno());
       printf("ERROR: Hubo un problema al cargar las imagenes de img\n");
       printf("\n");
   }
   return inicializado_correctamente;
}
```

Archivo fuente ***src\main.c***

Es el código fuente y principal para el funcionamiento del proyecto, desde aquí conecta e incluye a los demás archivos descritos anteriormente.

Dada la extensión del código, se invita al lector a visualizar el código de manera más cómoda en el repositorio del proyecto:

https://github.com/TheUniqueFersi/Control-de-Habitos/blob/master/src/main.c

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <math.h>
#include <time.h>
#include <allegro5/allegro.h>
#include <allegro5/allegro_image.h>
#include <allegro5/allegro_primitives.h>
#include <allegro5/allegro_font.h>
#include <allegro5/allegro_ttf.h>
#include <allegro5/allegro_audio.h>
#include <allegro5/allegro_acodec.h>
//#include "../include/structs.c"
#include "../include/CRUD.c"
#include "../include/resources.c"
#include "../tests/depuracion.c"
/* ----->  <---- */
/* ----> Prototipos <---- */
```

```
int inicializar_allegro();//INICIALIZA TODO LO NECESARIO PARA QUE ALLEGRO
FUNCIONÉ
void main_habitus(int, int);
void actualizar_display();
int init_resources();
void IUSD();
void llamarINSERT();
void llamarUPDATE();
void llamarSELECT();
void llamarDELETE();
void creacionEstructuras();
void analizar_fecha_habitos();
void verificarREGISTROS();
void rellenarRegistrosHabitos(int);
void CARGAR_TODOS_LOS_REGISTROS();
//termina;
/* ----> ALLEGRO {TIPO DE DATOS} <---- */
// Displays
ALLEGRO_DISPLAY *disp;

// Eventos
ALLEGRO_EVENT_QUEUE *cola_eventos;
ALLEGRO_EVENT evento;

//SONIDO
ALLEGRO_SAMPLE_ID id_SAMPLE;
ALLEGRO_SAMPLE *SAMPLE;

//BITMAPS:
ALLEGRO_BITMAP *BITMAP;

//TIMERS
ALLEGRO_TIMER *AFK;
/* ---- termina; ---- */
/* ----> ESTRUCTURAS GLOBALES <---- */
struct APP{
   int init;
   int ID_last_user;
};
USUARIO usuario ={1,""},usuarioprueba={0};
struct APP app_recibe = {0};
struct APP reseteoAPP = {0,1};
//ESTRUCTURAS VACIAS (Para DELETE y otros):
HABITO habNULL={0};
USUARIO usuNULL={0};
HORARIO  horarioNULL={0};
RECORDATORIOS recordNULL={0};
DIFICULTAD difNULL = {0};
TIPO tipoNULL = {0};
REGISTRO_HABITOS reg_habNULL = {0};
HORA_HORARIO hor_horarioNULL = {0};
PRODUCTIVIDAD productNULL = {0};
//ESTRUCTURAS GLOBALES
```

```
//TODOS LOS REGISTROS:
DIFICULTAD * Dificultades;
TIPO * Tipos;
HABITO * Habitos, * habitosNULL ;
REGISTRO_HABITOS * Reg_habitos, * Numero_Registros;
HORARIO * Horarios;
HORA_HORARIO * Hora_horarios;
RECORDATORIOS * Recordatorios;
PRODUCTIVIDAD * Productividad;
FECHA miFecha;
int n_reg_dificultades;
int n_reg_tipo;
int n_reg_habitos;
//int n_reh_habitos2;/*PRUEBA LUILLILOL*/
int cantidadREGISTROS_HABITOS;
int n_reg_reg_hab;
int n_reg_horario;
int n_reg_hora_horario;
int n_reg_recordatorios;
int n_reg_productividad;
int n_cantidad_registros_disponibles=0;
/* ---- termina; ---- */

/* ----> VARIABLES GLOBALES <---- */
int fin=0;
int momento=0, estado=0; /*0: Inicio primera vez*/
int estado_creando = 0;
int tamArrPos=6, loc=0;
int * arrPos, * arrHab;
int registrosInicializados = 0;
int altura_Y_registros=180;
int mensajeAdvertencia=0;
//Nos permiten navegar en el momento 1
/*
* //Estado: 0-> Leer, 1 -> Crear, 2-> Modificar, 3-> Eliminar
*              */
/* ----> Nombres/RUTAS de archivos que manejar <---- */
char rutaAPP[100] = {"./data/app.dat"};
char rutaUSUARIO[100] = {"./data/usuario.dat"};

//char frag_1rutas[100] = {""};
char frag_2rutaDIFICULTAD[100] = {"/dificultad.dat"};
char frag_2rutaTIPO[100] = {"/tipo.dat"};
char frag_2rutaHABITO[100] = {"/habito.dat"};
char frag_2rutaREGISTROHABITO[100] = {"/registro_habito.dat"};
char frag_2rutaHORARIO[100] = {"/horario.dat"};
char frag_2rutaHORA_HORARIO[100] = {"/hora_horario.dat"};
char frag_2rutaRECORDATORIO[100] = {"/recordatorio.dat"};
char frag_2rutaPRODUCTIVIDAD[100] = {"/productividad.dat"};


char rutaDIFICULTAD[100] = {"./data/usuarios/"};
char rutaTIPO[100] = {"./data/usuarios/"};
```

```
char rutaHABITO[100] = {"./data/usuarios/"};
char rutaREGISTROHABITO[100] = {"./data/usuarios/"};
char rutaHORARIO[100] = {"./data/usuarios/"};
char rutaHORA_HORARIO[100] = {"./data/usuarios/"};
char rutaRECORDATORIO[100] = {"./data/usuarios/"};
char rutaPRODUCTIVIDAD[100] = {"./data/usuarios/"};
/*
char rutaDIFICULTAD[100] = {"./data/usuarios/1/dificultad.dat"};
char rutaTIPO[100] = {"./data/usuarios/1/tipo.dat"};
char rutaHABITO[100] = {"./data/usuarios/1/habito.dat"};
char rutaREGISTROHABITO[100] = {"./data/usuarios/1/registro_habito.dat"};
char rutaHORARIO[100] = {"./data/usuarios/1/horario.dat"};
char rutaHORA_HORARIO[100] = {"./data/usuarios/1/horario.dat"};
char rutaRECORDATORIO[100] = {"./data/usuarios/1/recordatorio.dat"};
char rutaPRODUCTIVIDAD[100] = {"./data/usuarios/1/productividad.dat"};
*/
/* ---- termina; ---- */

char nombre[30] = {0};
char Titulo[30] = {0};
char notas[30] = {0};
char semana[7]="0000000";
int y=400,y2=400,x3=475,y3=305;
int dificultadHabito = 1;
int diaDeLaSemana = 0;
int cantiHabitosHoy = 0;
void cargarDiaDeLaSemana() {
   time_t t;
   struct tm *fechaa;
   time(&t);
   fechaa = localtime(&t);
   diaDeLaSemana = fechaa->tm_wday;
}
int obtenerHabitosHoy(){
   cantiHabitosHoy=0;
   for(int i = 0; i<n_reg_habitos; i++){
       if(Habitos[i].ID_habito != 0 &&
Habitos[i].repeticion_semanal[diaDeLaSemana] == '1' && Habitos[i].racha ==
0){
           cantiHabitosHoy++;
       }
   }
   mensajeAdvertencia = (cantiHabitosHoy==0 &&
n_cantidad_registros_disponibles>0)? 1:0;
   //printf("Cantidad de Hábitos para hoy: %i\n", cantiHabitosHoy);
   return cantiHabitosHoy;
}
void colorearDia(int x, int y, int valor) {
   if(valor==1){
       al_draw_filled_rectangle(x,y,x+20,y+20,principal_pale_chestnut);
   }else{
       al_draw_filled_rectangle(x,y,x+20,y+20,neutro1_tinta_de_pulpo);
   }
```

```
}
void Dia(int dia){
   al_draw_text(lexend_regular[15], texto_black, 1015, 375,
ALLEGRO_ALIGN_LEFT, "Do Lu  Ma Mi  Ju  Vi  Sa");
   for (int i = 0; i < 7; ++i) {
       if (i == dia) {
           al_draw_filled_rectangle(1015 + i * 25, 395, 1035 + i * 25, 415,
secundario_pastel_magenta);
       } else {
           al_draw_filled_rectangle(1015 + i * 25, 395, 1035 + i * 25, 415,
fondo_gris1);
       }
   }
}
void Pendientes(){
   //int CantHabHoy= atoi(CantHabitosHoy)+1;
   int CantHabHoy = cantiHabitosHoy;
   //-ALLEGRO_PI/2.0 SE UTILIZA PARA INICIAR EN LA PARTE SUPERIOR DE LA
CIRCUNFERENCIA
   //theta se trabaja en radianes
   //PasoHabitos=(-2*ALLEGRO_PI)/CantHabitosHoy
   //printf("\nHabitos hoy: %d\n",CantHabHoy);
   al_draw_line(1000,25,1000,675,fondo_gris1,2);
   //PasoRecordatorios=(-2*ALLEGRO_PI)/CantRecordatoriosHoy
   float theta=0;
   if(CantHabHoy==0)
       theta = -ALLEGRO_PI*2.0;
   else
       theta = -ALLEGRO_PI*2/(CantHabHoy+1);
   al_draw_arc(1100,175,
50,-ALLEGRO_PI/2.0,theta,neutro1_tinta_de_pulpo,15.0);

al_draw_textf(lexend_regular[30],texto_black,1100,155,ALLEGRO_ALIGN_CENTER,
"%i", cantiHabitosHoy);
}
/*Crear registro en registro habitos con el dia de hoy
* Para racha checar con la fecha inicial
* */
void calendario(int dia_semana, int mes,int primero,int anio){
   srand(time(NULL));
   int FILAS = 6;
   int COLUMNAS = 7;
   int CELDA=25;
   int tipomes=mes%2;
   int dias_en_mes;
   int primerafila=0;
   int bisiesto=anio%4;

   if(mes==2){
       if(bisiesto!=0){
           dias_en_mes=28;
       }else{
           dias_en_mes=29;
```

```
            //printf("dias en mes: %d",dias_en_mes);
        }
    }else if(tipomes==0){
        dias_en_mes=31;
    }else if(tipomes==1){
        dias_en_mes=30;
    }
    float transparencia;
    for (int fila = 0; fila < FILAS; ++fila) {
        for (int columna = 0; columna < COLUMNAS; ++columna) {
            if(primerafila==0){
                columna=primero;
                primerafila=1;
            }
            int dia_calendario = fila * COLUMNAS + columna + 1 - primero;
            //printf("Dia en calendario %i\n", columna);
            //Aqui va la lógica para poder hacer la transparencia en base a la
cantidad de actividades que tuvo,solo que ocupo la cantidad
            if (dia_calendario >= 1 && dia_calendario <= dias_en_mes) {
                //transparencia=((rand())%25)*10;
                transparencia = 40;

                if(Habitos != NULL) {
                    for (int pos = 0; pos < n_cantidad_registros_disponibles;
pos++) {
                        //printf("Depurando ando: %s\n",
Habitos[arrHab[pos]].repeticion_semanal);
                        //printf("Depurando ando: %c\n",
Habitos[arrHab[pos]].repeticion_semanal[columna]);
                        if (Habitos[arrHab[pos]].repeticion_semanal[columna]
== '1') {
                            if(transparencia+15 < 255){
                                transparencia+=15;
                            }
                        }
                    }
                }
                if(transparencia<(255/6)){
                    transparencia=0;
                }
                // Dibujar calendario
                    al_draw_filled_rectangle(1012 + columna * CELDA+3, 500 + 3
+ fila * CELDA,
                                             1012 + (columna + 1) * CELDA, 500
+ (fila + 1) * CELDA,
                                             fondo_gris1);
                if (transparencia != 0) {
                    al_draw_filled_rectangle(1012 + columna * CELDA+3, 500+ 3
+ fila * CELDA,
                                             1012 + (columna + 1) * CELDA, 500
+ (fila + 1) * CELDA,

al_map_rgba(248,107,234,transparencia));
```

```
                }
                // Puedes agregar el número del día del mes
                char texto_dia[3];
                snprintf(texto_dia, sizeof(texto_dia), "%d", dia_calendario);
                al_draw_text(roboto_black[10], texto_black,
                             1012 + 3 + columna * CELDA + CELDA / 2, 495 +
fila * CELDA + CELDA / 2,
                             ALLEGRO_ALIGN_CENTER, texto_dia);
            }
        }
    }
}
void ObtenerHora(){
    char hora_formateada[9];
    time_t Hora=time(NULL);
    struct tm *primerdia= localtime(&Hora);
    primerdia->tm_mday = 1;
    primerdia->tm_hour = 0;
    primerdia->tm_min = 0;
    primerdia->tm_sec = 0;
    mktime(primerdia);
    int primero=primerdia->tm_wday;
    time_t HoraActual = time(NULL);
    struct tm *info_tiempo = localtime(&HoraActual);
    char dia_formateado[60];
    strftime(hora_formateada, sizeof(hora_formateada), "%H:%M", info_tiempo);
    int dia = info_tiempo->tm_mday;
    int dia_semana = info_tiempo->tm_wday;
    int mes = info_tiempo->tm_mon + 1;
    int anio = info_tiempo->tm_year + 1900;
    sprintf(dia_formateado,"%02d/%02d/%d",dia,mes,anio);
    Pendientes();
    Dia(dia_semana);
    calendario(dia_semana,mes,primero,anio);
    al_draw_text(lexend_regular[59], texto_black, 1100, 310,
ALLEGRO_ALIGN_CENTER, hora_formateada);
    al_draw_text(lexend_regular[20], texto_black, 1100, 420,
ALLEGRO_ALIGN_CENTER, dia_formateado);
}
void ventanaActual(){
    int i=0;

    switch (momento) {
        case 0:
            if(estado==0){
                al_draw_filled_rectangle(0,0,1200,700,
fondo_principal_oscuro);
                al_draw_text(lexend_regular[40],
texto_white,600,300,ALLEGRO_ALIGN_CENTER,"Parece que es tu primera vez
abriendo Habitus");
                al_draw_text(lexend_regular[30],
texto_white,600,350,ALLEGRO_ALIGN_CENTER,"Presiona cualquier tecla para
continuar");
```

```
            }
            else{
                al_draw_filled_rectangle(0,0,1200,700,
fondo_principal_oscuro);
                al_draw_text(lexend_regular[25], texto_white, 600, 300,
ALLEGRO_ALIGN_CENTER, "Ingresa tu nombre:");
                al_draw_rectangle(280,340,920,380,texto_white,5);
                al_draw_text(lexend_regular[30], texto_white, 600, 340,
ALLEGRO_ALIGN_CENTER, nombre);
            }
            break;
        case 1:
            al_draw_scaled_bitmap(HABITOSROSA, 0, 0, 100, 300, 0, 0,100, 300,
0);
            //al_draw_scaled_bitmap(CALENDARIOBLANCO, 0, 0, 100, 300, 0,
175,100, 300, 0);
            al_draw_filled_rectangle(0, 175, 100, 350, texto_white);
            //al_draw_scaled_bitmap(RECORDS, 0, 0, 100, 300, 0, 350,100, 300,
0);
            al_draw_filled_rectangle(0, 350, 100, 525, texto_white);
            al_draw_scaled_bitmap(AJUSTES, 0, 0, 100, 300, 0, 525,100, 300,
0);

            al_draw_filled_rectangle(100,0,1000,700, fondo_principal_oscuro);
            al_draw_text(lexend_regular[40],al_map_rgba(0, 0, 0,
100),12,0,ALLEGRO_ALIGN_CENTER,"1");
            //al_draw_text(lexend_regular[40],al_map_rgba(0, 0, 0,
100),12,175,ALLEGRO_ALIGN_CENTER,"2");

            //al_draw_text(lexend_regular[40],al_map_rgba(0, 0, 0,
100),12,345,ALLEGRO_ALIGN_CENTER,"3");
            al_draw_text(lexend_regular[40],al_map_rgba(0, 0, 0,
100),12,513,ALLEGRO_ALIGN_CENTER,"4");
            al_draw_filled_rectangle(1000, 0, 1200, 700, al_map_rgb(255, 255,
255));

            al_draw_filled_rectangle(100, 0, 1000, 700,
fondo_principal_oscuro);
            creacionEstructuras();
            //al_draw_text(lexend_regular[40],al_map_rgba(0, 0, 0,
100),12,345,ALLEGRO_ALIGN_CENTER,"3");
            al_draw_text(lexend_regular[40],al_map_rgba(0, 0, 0,
100),12,513,ALLEGRO_ALIGN_CENTER,"4");
            /*Interfaz integrada para cada estado*/
            if(estado==1){
                al_draw_filled_rectangle(200,0,900,550, al_map_rgba(74, 63, 75
, 220));/*Ventana emergente*/
                al_draw_filled_rectangle(360,170,730,215, texto_white);

al_draw_text(lexend_regular[30],texto_black,540,175,ALLEGRO_ALIGN_CENTER,"HA
BITUS");
            }
            if(estado==5){
```

```
                //recuadro con transparencia y mensaje deseas eliminar yes or
no
                al_draw_filled_rounded_rectangle(225, 100, 890, 675, 25, 25,
al_map_rgba(0, 0,0, 160));//222, 186, 192, 1
                al_draw_text(lexend_regular[40],al_map_rgb(255, 255, 255),560,
300, ALLEGRO_ALIGN_CENTER,"¿Estás seguro que");//deseas eliminar este
elemento?
                al_draw_text(lexend_regular[40],al_map_rgb(255, 255,
255),560,340,ALLEGRO_ALIGN_CENTER,"deseas eliminar este elemento?");

                al_draw_filled_rounded_rectangle(425, 425, 525, 525, 25, 25,
al_map_rgba(0, 0,0, 160));//222, 186, 192, 1
                al_draw_filled_rounded_rectangle(625, 425, 725, 525, 25, 25,
al_map_rgba(0, 0,0, 160));//222, 186, 192, 1
                al_draw_text(lexend_regular[20],al_map_rgb(255, 255, 255),475,
450, ALLEGRO_ALIGN_CENTER,"Sí");
                al_draw_text(lexend_regular[20],al_map_rgb(255, 255, 255),475,
475, ALLEGRO_ALIGN_CENTER,"(Enter)");

                al_draw_text(lexend_regular[20],al_map_rgb(255, 255, 255),675,
450, ALLEGRO_ALIGN_CENTER,"No");
                al_draw_text(lexend_regular[20],al_map_rgb(255, 255, 255),675,
475, ALLEGRO_ALIGN_CENTER,"(Esc)");
            }
            if(mensajeAdvertencia==1){
                al_draw_filled_rounded_rectangle(225, 100, 890, 675, 25, 25,
al_map_rgba(0, 0,0, 160));//222, 186, 192, 1
                al_draw_text(lexend_regular[30],al_map_rgb(255, 255, 255),560,
300, ALLEGRO_ALIGN_CENTER,"¡Has hecho todos los hábitos de hoy!");//deseas
eliminar este elemento?
                al_draw_text(lexend_regular[20],al_map_rgb(255, 255,
255),560,340,ALLEGRO_ALIGN_CENTER,"Presiona ESP para salir");
            } else if(mensajeAdvertencia == 2){
                al_draw_filled_rounded_rectangle(225, 100, 890, 675, 25, 25,
al_map_rgba(0, 0,0, 160));//222, 186, 192, 1
                al_draw_text(lexend_regular[30],al_map_rgb(255, 255, 255),560,
300, ALLEGRO_ALIGN_CENTER,"No tienes que hacer este hábito hoy");//deseas
eliminar este elemento?
                al_draw_text(lexend_regular[20],al_map_rgb(255, 255,
255),560,340,ALLEGRO_ALIGN_CENTER,"Presiona ESP para salir");
            }
            //al_draw_text(lexend_regular[40],al_map_rgba(0, 0, 0,
100),12,335,ALLEGRO_ALIGN_CENTER,"3");
            //al_draw_text(lexend_regular[40],al_map_rgba(0, 0, 0,
100),12,503,ALLEGRO_ALIGN_CENTER,"4"); ---VERIFICAR
            al_draw_filled_rectangle(100, 0, 1000, 150,
fondo_principal_oscuro);
            al_draw_scaled_bitmap(LOGO, 0, 0, 516, 484, 488, 16,125, 125, 0);
            /*
            al_draw_filled_rectangle(300, 90, 500, 120, al_map_rgb(214, 164,
226));//Cuadro Hoy
            al_draw_filled_rectangle(600, 90, 800, 120, al_map_rgb(214, 164,
226));//Cuadro Todos
```

```
al_draw_text(lexend_regular[28],texto_black,390,85,ALLEGRO_ALIGN_CENTER,"Hoy"
);

al_draw_text(lexend_regular[20],texto_black,440,90,ALLEGRO_ALIGN_CENTER,"(H)"
);

al_draw_text(lexend_regular[28],texto_black,690,85,ALLEGRO_ALIGN_CENTER,"Todo
s");

al_draw_text(lexend_regular[20],texto_black,750,90,ALLEGRO_ALIGN_CENTER,"(T)"
);
            */

            //mensaje Haz completado el hábito
            //al_draw_text(lexend_regular[40],al_map_rgb(255, 255, 255),560,
300, ALLEGRO_ALIGN_CENTER,"Haz completado las acciones del día");//deseas
eliminar este elemento?

            /*
            //botones de recordatorios y texto
            al_draw_filled_rectangle(300, 90, 500, 150, al_map_rgb(214, 164,
226));//Cuadro realizado

al_draw_text(lexend_regular[28],texto_black,400,85,ALLEGRO_ALIGN_CENTER,"Real
izado");

al_draw_text(lexend_regular[28],texto_black,400,115,ALLEGRO_ALIGN_CENTER,"(Y)
");
            al_draw_filled_rectangle(600, 90, 800, 150, al_map_rgb(214, 164,
226));//Cuadro realizado

al_draw_text(lexend_regular[28],texto_black,705,85,ALLEGRO_ALIGN_CENTER,"No
realizado");

al_draw_text(lexend_regular[28],texto_black,705,115,ALLEGRO_ALIGN_CENTER,"(N)
");
            //al_draw_filled_rectangle(600, 90, 800, 120, al_map_rgb(214, 164,
226));//Cuadro no realizado*/

            //l_draw_filled_rectangle(150, 150, 950, 600, al_map_rgb(119, 86,
122));
            al_draw_filled_rectangle(100, 525, 165, 700, al_map_rgb(255, 255,
255));//222, 186, 192, 1
            al_draw_filled_rectangle(935, 525, 1110, 700,al_map_rgb(255, 255,
255));//222, 186, 192, 1

            //texto de los bitmaps nuevo, editar, borrar
            // ------------------- MERGE TODO
            /*

al_draw_text(lexend_regular[30],texto_black,195,508,ALLEGRO_ALIGN_CENTER,"A")
;
```

```
al_draw_text(lexend_regular[30],texto_black,195,574,ALLEGRO_ALIGN_CENTER,"E")
;

al_draw_text(lexend_regular[30],texto_black,195,642,ALLEGRO_ALIGN_CENTER,"B")
;
           al_draw_scaled_bitmap(NUEVOHABITO, 0, 0, 738, 740, 100, 496,75,
68, 0);
           al_draw_scaled_bitmap(EDITARHABITO, 0, 0, 740, 744, 100, 564,75,
68, 0);
           al_draw_scaled_bitmap(BORRARHABITO, 0, 0, 744, 740, 100, 632,75,
68, 0);
           al_draw_scaled_bitmap(FLECHAS, 0, 0, 360, 360, 850, 496,210, 210,
0);
            */
           if(estado==1){
               al_draw_filled_rectangle(100,0,1000,700,
al_map_rgba(47,50,58,200));
               al_draw_filled_rectangle(200,100,900,550, al_map_rgba(74, 63,
75 , 220));/*Ventana emergente*/
               al_draw_text(lexend_regular[20], texto_white, 550, 120,
ALLEGRO_ALIGN_CENTER, "Escribe el nombre de tu habito");
               al_draw_filled_rectangle(250,150,850,200, texto_white);
               al_draw_text(lexend_regular[30], texto_black, 550, 160,
ALLEGRO_ALIGN_CENTER, Titulo);
               al_draw_text(lexend_regular[20], texto_white, 550, 220,
ALLEGRO_ALIGN_CENTER, "Escribe una nota para tu habito (Puedes dejarlo en
blanco)");
               al_draw_filled_rectangle(250,250,850,300, texto_white);
               al_draw_text(lexend_regular[18], texto_white, 550, 320,
ALLEGRO_ALIGN_CENTER, "1      2      3      4       5       6      7");
               al_draw_text(lexend_regular[20], texto_white, 550, 340,
ALLEGRO_ALIGN_CENTER, "Do  Lu  Ma  Mi   Ju   Vi   Sa");
               int CalX=422,CalY=365;
               for (int i = 0; i < 4; ++i) {
                   for(int j=0;i<7;i++){
                       int valor=semana[i]-48;
                       colorearDia(CalX,CalY,valor);
                       CalX+=40;
                   }
               }
           }else if(estado==2){
               al_draw_filled_rectangle(100,0,1000,700,
al_map_rgba(47,50,58,200));
               al_draw_filled_rectangle(200,100,900,550, al_map_rgba(74, 63,
75 , 220));/*Ventana emergente*/
               al_draw_text(lexend_regular[20], texto_white, 550, 120,
ALLEGRO_ALIGN_CENTER, "Escribe el nombre de tu habito");
               al_draw_filled_rectangle(250,150,850,200, texto_white);
               al_draw_text(lexend_regular[30], texto_black, 550, 160,
ALLEGRO_ALIGN_CENTER, Titulo);
```

```
                al_draw_text(lexend_regular[20], texto_white, 550, 220,
ALLEGRO_ALIGN_CENTER, "Escribe una nota para tu habito(Puedes dejarlo en
blanco)");
                al_draw_filled_rectangle(250,250,850,300, texto_white);

al_draw_text(lexend_regular[30],texto_black,550,260,ALLEGRO_ALIGN_CENTER,not
as);
                al_draw_text(lexend_regular[18], texto_white, 550, 320,
ALLEGRO_ALIGN_CENTER, "1      2      3      4       5       6      7");
                al_draw_text(lexend_regular[20], texto_white, 550, 340,
ALLEGRO_ALIGN_CENTER, "Do  Lu  Ma  Mi   Ju   Vi   Sa");
                int CalX=422,CalY=365;
                for (int i = 0; i < 4; ++i) {
                    for(int j=0;i<7;i++){
                        int valor=semana[i]-48;
                        colorearDia(CalX,CalY,valor);
                        CalX+=40;
                    }
                }
            }else if(estado==3){
                al_draw_filled_rectangle(100,0,1000,700,
al_map_rgba(47,50,58,200));
                al_draw_filled_rectangle(200,100,900,550, al_map_rgba(74, 63,
75 , 220));/*Ventana emergente*/
                al_draw_text(lexend_regular[20], texto_white, 550, 120,
ALLEGRO_ALIGN_CENTER, "Escribe el nombre de tu habito");
                al_draw_filled_rectangle(250,150,850,200, texto_white);
                al_draw_text(lexend_regular[30], texto_black, 550, 160,
ALLEGRO_ALIGN_CENTER, Titulo);
                al_draw_text(lexend_regular[20], texto_white, 550, 220,
ALLEGRO_ALIGN_CENTER, "Escribe una nota para tu habito(Puedes dejarlo en
blanco)");
                al_draw_filled_rectangle(250,250,850,300, texto_white);

al_draw_text(lexend_regular[30],texto_black,550,260,ALLEGRO_ALIGN_CENTER,not
as);
                al_draw_text(lexend_regular[18], texto_white, 550, 320,
ALLEGRO_ALIGN_CENTER, "1      2      3      4       5       6      7");
                al_draw_text(lexend_regular[20], texto_white, 550, 340,
ALLEGRO_ALIGN_CENTER, "Do  Lu  Ma  Mi   Ju   Vi   Sa");
                int CalX=422,CalY=365;
                for (int i = 0; i < 4; ++i) {
                    for(int j=0;i<7;i++){
                        int valor=semana[i]-48;
                        colorearDia(CalX,CalY,valor);
                        CalX+=40;
                    }
                }
            }else if(estado==4) {

                //interfaz de dificultad
                al_draw_filled_rectangle(100, 0, 1000, 1200, al_map_rgba(47,
50, 58,200)); //rectangulo que tapa lo de Arias
```

```
                al_draw_filled_rounded_rectangle(325, 150, 800, 720, 100, 100,
al_map_rgb(227, 218, 201));
                al_draw_filled_circle(562, 400, 238, al_map_rgb(227, 218,
201));//255, 134, 0, 1
                //al_draw_filled_pieslice(0,0,40, al_map_rgb(74, 63, 75));
                al_draw_pieslice(475, 400, 120, 2.9, 3.6, al_map_rgb(255, 255,
255), 4);

                //texto de los bitmaps nuevo, editar, borrar

                al_draw_arc(475, 400, 120, 2.95, 0.55, al_map_rgba(22, 82, 1,
244), 20);//22,82,1
                al_draw_arc(475, 400, 120, 3.6, 0.6, al_map_rgba(69, 183, 30,
244), 20);//255, 134, 0, 1
                al_draw_arc(475, 400, 120, 4.3, 0.8, al_map_rgba(255, 255, 0,
244), 20);
                al_draw_arc(475, 400, 120, 5.2, 0.7, al_map_rgba(255, 134, 0,
244), 20);//69,183,30
                al_draw_arc(475, 400, 120, 5.95, 0.5, al_map_rgba(255, 0, 0,
244), 20);//255,0,0
                al_draw_filled_circle(475, 400, 15, al_map_rgb(0, 0,
0));//255, 134, 0, 1
                al_draw_filled_triangle(470, y, 480, y2, x3, y3, al_map_rgb(0,
0, 0));
                //muy
facil:yvar:405,x3:400,y3:385,facil:x3:400,y3:315,normal:x3:475,y3:300,dificil
:x3:550,y:315,muy dificil:yvar:405,x3:550,y3:385
                al_draw_filled_rounded_rectangle(450, 175, 675, 250, 25, 25,
                                                 al_map_rgb(222, 186,
201));//222, 186, 192, 1
                al_draw_scaled_bitmap(LOGO, 0, 0, 516, 484, 360, 175, 80, 80,
0);
                al_draw_filled_rounded_rectangle(625, 270, 780, 635, 25, 25,
                                                 al_map_rgb(222, 186,
201));//222, 186, 192, 1
                al_draw_filled_rounded_rectangle(650, 290, 755, 330, 10, 10,
al_map_rgb(146, 98, 107));//146, 98, 107, 1
                al_draw_filled_rounded_rectangle(650, 335, 755, 403, 10, 10,
al_map_rgb(146, 98, 107));//146, 98, 107, 1
                al_draw_filled_rounded_rectangle(650, 408, 755, 476, 10, 10,
al_map_rgb(146, 98, 107));//146, 98, 107, 1
                al_draw_filled_rounded_rectangle(650, 481, 755, 549, 10, 10,
al_map_rgb(146, 98, 107));//146, 98, 107, 1
                al_draw_filled_rounded_rectangle(650, 554, 755, 622, 10, 10,
al_map_rgb(146, 98, 107));//146, 98, 107, 1
                // texto de los rectángulos de dificultad
                al_draw_text(lexend_regular[15], texto_white, 700, 300,
ALLEGRO_ALIGN_CENTER, "1. Muy facil");
                al_draw_text(lexend_regular[20], texto_white, 700, 350,
ALLEGRO_ALIGN_CENTER, "2. Facil");
                al_draw_text(lexend_regular[16], texto_white, 703, 423,
ALLEGRO_ALIGN_CENTER, "3. Intermedio");
```

```
                al_draw_text(lexend_regular[20], texto_white, 700, 500,
ALLEGRO_ALIGN_CENTER, "4. Difícil");
                al_draw_text(lexend_regular[16], texto_white, 700, 574,
ALLEGRO_ALIGN_CENTER, "5. Muy difícil");
                al_draw_text(lexend_regular[45], texto_white, 565, 185,
ALLEGRO_ALIGN_CENTER, "Dificultad");
            }

al_draw_text(lexend_regular[25],texto_black,157,551,ALLEGRO_ALIGN_CENTER,"A"
);

al_draw_text(lexend_regular[25],texto_black,157,640,ALLEGRO_ALIGN_CENTER,"B"
);
            al_draw_scaled_bitmap(NUEVOHABITO, 0, 0, 738, 740, 100, 544,50,
50, 0);
            al_draw_scaled_bitmap(BORRARHABITO, 0, 0, 744, 740, 100, 632,50,
50, 0);
            al_draw_scaled_bitmap(FLECHAS, 0, 0, 360, 360, 895, 540,150, 150,
0);
            break;
        case 2:
            al_draw_scaled_bitmap(HABITOS, 0, 0, 100, 300, 0, 0,100, 300, 0);
            al_draw_scaled_bitmap(CALENDARIOROSA, 0, 0, 100, 300, 0, 175,100,
300, 0);
            //al_draw_scaled_bitmap(RECORDS, 0, 0, 100, 300, 0, 350,100, 300,
0);
            al_draw_scaled_bitmap(AJUSTES, 0, 0, 100, 300, 0, 525,100, 300,
0);
            al_draw_text(lexend_regular[40],al_map_rgba(0, 0, 0,
100),12,0,ALLEGRO_ALIGN_CENTER,"1");
            //al_draw_text(lexend_regular[40],al_map_rgba(0, 0, 0,
100),12,175,ALLEGRO_ALIGN_CENTER,"2");
            //al_draw_text(lexend_regular[40],al_map_rgba(0, 0, 0,
100),12,345,ALLEGRO_ALIGN_CENTER,"3");
            al_draw_text(lexend_regular[39],al_map_rgba(0, 0, 0,
100),12,513,ALLEGRO_ALIGN_CENTER,"4");

            break;
        case 3:
            al_draw_scaled_bitmap(HABITOS, 0, 0, 100, 300, 0, 0,100, 300, 0);
            //al_draw_scaled_bitmap(CALENDARIOBLANCO, 0, 0, 100, 300, 0,
175,100, 300, 0);
            al_draw_scaled_bitmap(RECORDSROSA, 0, 0, 100, 300, 0, 350,100,
300, 0);
            al_draw_scaled_bitmap(AJUSTES, 0, 0, 100, 300, 0, 525,100, 300,
0);
            al_draw_text(lexend_regular[40],al_map_rgba(0, 0, 0,
100),12,0,ALLEGRO_ALIGN_CENTER,"1");
            //al_draw_text(lexend_regular[40],al_map_rgba(0, 0, 0,
100),12,175,ALLEGRO_ALIGN_CENTER,"2");
            //al_draw_text(lexend_regular[40],al_map_rgba(0, 0, 0,
100),12,335,ALLEGRO_ALIGN_CENTER,"3");--CONFLICTO
```

```
            //al_draw_text(lexend_regular[40],al_map_rgba(0, 0, 0,
100),12,503,ALLEGRO_ALIGN_CENTER,"4"); --CONFLICTO
            //al_draw_text(lexend_regular[40],al_map_rgba(0, 0, 0,
100),12,345,ALLEGRO_ALIGN_CENTER,"3");
            al_draw_text(lexend_regular[40],al_map_rgba(0, 0, 0,
100),12,513,ALLEGRO_ALIGN_CENTER,"4");
            al_draw_filled_rectangle(100,0,1000,700, fondo_principal_oscuro);
            al_draw_text(lexend_regular[40],al_map_rgba(0, 0, 0,
100),12,0,ALLEGRO_ALIGN_CENTER,"1");
            //al_draw_text(lexend_regular[40],al_map_rgba(0, 0, 0,
100),12,175,ALLEGRO_ALIGN_CENTER,"2");
            //al_draw_text(lexend_regular[40],al_map_rgba(0, 0, 0,
100),12,345,ALLEGRO_ALIGN_CENTER,"3");
            al_draw_text(lexend_regular[39],al_map_rgba(0, 0, 0,
100),12,513,ALLEGRO_ALIGN_CENTER,"4");

            al_draw_filled_rectangle(1000, 0, 1200, 700, al_map_rgb(255, 255,
255));
            al_draw_filled_rectangle(100, 0, 1000, 700,
fondo_principal_oscuro);
            al_draw_filled_rectangle(100, 0, 1000, 150,
fondo_principal_oscuro);
            al_draw_scaled_bitmap(LOGO, 0, 0, 516, 484, 488, 0,125, 125, 0);
            //l_draw_filled_rectangle(150, 150, 950, 600, al_map_rgb(119, 86,
122));

            al_draw_filled_rectangle(100, 525, 165, 700, al_map_rgb(255, 255,
255));//222, 186, 192, 1
            al_draw_filled_rectangle(935, 525, 1110, 700,al_map_rgb(255, 255,
255));//222, 186, 192, 1
            //texto de los bitmaps nuevo, editar, borrar

al_draw_text(lexend_regular[25],texto_black,157,551,ALLEGRO_ALIGN_CENTER,"A"
);

al_draw_text(lexend_regular[25],texto_black,157,640,ALLEGRO_ALIGN_CENTER,"B"
);
            al_draw_scaled_bitmap(NUEVOHABITO, 0, 0, 738, 740, 100, 544,50,
50, 0);
            al_draw_scaled_bitmap(BORRARHABITO, 0, 0, 744, 740, 100, 632,50,
50, 0);
            al_draw_scaled_bitmap(FLECHAS, 0, 0, 360, 360, 895, 540,150, 150,
0);
            //al_draw_text(lexend_regular[59],al_map_rgb(255, 255, 255),
580,250,ALLEGRO_ALIGN_CENTER,"Próximamente");
            break;
        case 4:
            al_draw_scaled_bitmap(HABITOS, 0, 0, 100, 300, 0, 0,100, 300, 0);
            //al_draw_scaled_bitmap(CALENDARIOBLANCO, 0, 0, 100, 300, 0,
175,100, 300, 0);
            //al_draw_filled_rectangle(0, 175, 100, 350, texto_white);
            //al_draw_scaled_bitmap(RECORDS, 0, 0, 100, 300, 0, 350,100, 300,
0);
```

```
            al_draw_filled_rectangle(0, 175, 100, 350, texto_white);
            al_draw_filled_rectangle(0, 350, 100, 525, texto_white);
            al_draw_scaled_bitmap(AJUSTESROSA, 0, 0, 100, 300, 0, 525,100,
300, 0);
            al_draw_text(lexend_regular[40],al_map_rgba(0, 0, 0,
100),12,0,ALLEGRO_ALIGN_CENTER,"1");
            //al_draw_text(lexend_regular[40],al_map_rgba(0, 0, 0,
100),12,175,ALLEGRO_ALIGN_CENTER,"2");

            //al_draw_text(lexend_regular[40],al_map_rgba(0, 0, 0,
100),12,345,ALLEGRO_ALIGN_CENTER,"3");
            al_draw_text(lexend_regular[39],al_map_rgba(0, 0, 0,
100),12,513,ALLEGRO_ALIGN_CENTER,"4");

            al_draw_filled_rectangle(100,0,1000,700, fondo_principal_oscuro);
            al_draw_text(lexend_regular[15], texto_white, 550, 310,
ALLEGRO_ALIGN_CENTER, "Ingresa tu nombre:");
            al_draw_rectangle(300,340,800,365,texto_white,5);
            al_draw_text(lexend_regular[20], texto_white, 550, 340,
ALLEGRO_ALIGN_CENTER, nombre);
            break;
        default:
            break;
    }
    if(momento!=0){
        al_draw_filled_rectangle(1000, 0, 1200, 700, al_map_rgb(255, 255,
255));
        ObtenerHora();
        al_draw_line(100,25,100,675,fondo_gris1,2);
    }
}
void actualizar_display(){
    //FIGURAS PRIMITAVAS
    //al_draw_text(lexend_regular[10],al_map_rgba(255, 255, 255,
10),199,500,ALLEGRO_ALIGN_CENTER,"1");
    //al_draw_filled_rectangle(1000, 0, 1200, 700, al_map_rgb(255, 255, 255));
--CONFLICTO
    //al_draw_filled_rectangle(101, 0, 1000, 720, al_map_rgb(47, 50, 58));
--CONFLICTO
    //al_draw_filled_rectangle(300, 90, 500, 120, al_map_rgb(214, 164, 226));
--CONFLICTO
    //al_draw_filled_rectangle(600, 90, 800, 120, al_map_rgb(214, 164, 226));
--CONFLICTO
    //al_draw_filled_rectangle(150, 150, 950, 600, al_map_rgb(119, 86, 122));
    al_draw_filled_rectangle(175, 175, 925, 400, al_map_rgb(74, 63, 75));
    al_draw_filled_rectangle(190, 190, 910, 235, al_map_rgb(227, 158,
193));//227, 158, 193, 1
    al_draw_filled_rectangle(190, 250, 540, 370, al_map_rgb(227, 158, 193));
//(225, 0, 129));
    al_draw_filled_rectangle(550, 250, 910, 370, al_map_rgb(227, 158, 193));
    //al_draw_filled_rectangle(550, 250, 910, 290, al_map_rgb(225, 0, 129));
    //interfaz de dificultad
```

```
  al_draw_filled_rectangle(0, 0, 1000 , 1200, al_map_rgb(47, 50, 58));
//rectangulo que tapa lo de Arias
  al_draw_filled_rounded_rectangle(325, 150, 800, 720, 100, 100,
al_map_rgb(227, 218, 201));
  al_draw_filled_circle(562, 400, 238, al_map_rgb(227, 218, 201));//255, 134,
0, 1
  //al_draw_filled_pieslice(0,0,40, al_map_rgb(74, 63, 75));
  al_draw_pieslice(475, 400, 120,2.9, 3.6, al_map_rgb(255, 255, 255), 4);

   al_draw_filled_rectangle(100, 525, 165, 700, al_map_rgb(255, 255,
255));//222, 186, 192, 1
  //texto de los bitmaps nuevo y borrar

al_draw_text(lexend_regular[25],texto_black,157,551,ALLEGRO_ALIGN_CENTER,"A"
);

al_draw_text(lexend_regular[25],texto_black,157,640,ALLEGRO_ALIGN_CENTER,"B"
);
  al_draw_scaled_bitmap(NUEVOHABITO, 0, 0, 738, 740, 100, 544,50, 50, 0);
  al_draw_scaled_bitmap(BORRARHABITO, 0, 0, 744, 740, 100, 632,50, 50, 0);
  //al_draw_scaled_bitmap(BORRARHABITO, 0, 0, 744, 740, 100, 632,75, 68, 0);

  al_draw_arc(475, 400, 120, 2.95, 0.55, al_map_rgba(255, 0, 0, 244), 20);
  al_draw_arc(475, 400, 120, 3.6, 0.6, al_map_rgba(255, 134, 0, 244),
20);//255, 134, 0, 1
  al_draw_arc(475, 400, 120, 4.3, 0.8, al_map_rgba(255, 255, 0, 244), 20);
  al_draw_arc(475, 400, 120, 5.2, 0.7, al_map_rgba(69, 183, 30, 244), 20);
  al_draw_arc(475, 400, 120, 5.95, 0.5, al_map_rgba(22, 82, 1, 244),
20);//22, 82, 1, 1
  al_draw_filled_circle(475, 400, 15, al_map_rgb(0, 0, 0));//255, 134, 0, 1
  al_draw_filled_triangle(470, 400, 480, 400, 475, 300, al_map_rgb(0, 0, 0));
  al_draw_filled_rounded_rectangle(450, 175, 675, 250, 25, 25,
al_map_rgb(222, 186, 201));//222, 186, 192, 1
  al_draw_scaled_bitmap(LOGO, 0, 0, 516, 484, 360, 175,80, 80, 0);
  al_draw_filled_rounded_rectangle(625, 270, 780, 635, 25, 25,
al_map_rgb(222, 186, 201));
  al_draw_filled_rounded_rectangle(650, 290, 755, 330, 10, 10,
al_map_rgb(146, 98, 107));
  al_draw_filled_rounded_rectangle(650, 335, 755, 403, 10, 10,
al_map_rgb(146, 98, 107));
  al_draw_filled_rounded_rectangle(650, 408, 755, 476, 10, 10,
al_map_rgb(146, 98, 107));
  al_draw_filled_rounded_rectangle(650, 481, 755, 549, 10, 10,
al_map_rgb(146, 98, 107));
  al_draw_filled_rounded_rectangle(650, 554, 755, 622, 10, 10,
al_map_rgb(146, 98, 107));
   al_draw_text(lexend_regular[39],al_map_rgb(255, 255,
255),12,513,ALLEGRO_ALIGN_CENTER,"Próximamente");
  // texto de los rectángulos de dificultad
   al_draw_scaled_bitmap(LOGO, 0, 0, 516, 484, 488, 0,125, 125, 0);

al_draw_text(lexend_regular[15],texto_white,700,300,ALLEGRO_ALIGN_CENTER,"1.
Muy facil");
```

```
al_draw_text(lexend_regular[20],texto_white,700,350,ALLEGRO_ALIGN_CENTER,"2.
Facil");

al_draw_text(lexend_regular[16],texto_white,703,423,ALLEGRO_ALIGN_CENTER,"3.
Intermedio");

al_draw_text(lexend_regular[20],texto_white,700,500,ALLEGRO_ALIGN_CENTER,"4.
Difícil");

al_draw_text(lexend_regular[16],texto_white,700,574,ALLEGRO_ALIGN_CENTER,"5.
Muy difícil");

al_draw_text(lexend_regular[45],texto_white,565,185,ALLEGRO_ALIGN_CENTER,"Di
ficultad");

   al_draw_filled_rectangle(935, 525, 1110, 700,al_map_rgb(255, 255,
255));//222, 186, 192, 1

   al_draw_scaled_bitmap(FLECHAS, 0, 0, 360, 360, 895, 540,150, 150, 0);
           /*
           //botones de recordatorios y texto
           al_draw_filled_rectangle(300, 90, 500, 150, al_map_rgb(214, 164,
226));//Cuadro realizado

al_draw_text(lexend_regular[28],texto_black,400,85,ALLEGRO_ALIGN_CENTER,"Real
izado");

al_draw_text(lexend_regular[28],texto_black,400,115,ALLEGRO_ALIGN_CENTER,"(Y)
");
           al_draw_filled_rectangle(600, 90, 800, 150, al_map_rgb(214, 164,
226));//Cuadro realizado

al_draw_text(lexend_regular[28],texto_black,705,85,ALLEGRO_ALIGN_CENTER,"No
realizado");

al_draw_text(lexend_regular[28],texto_black,705,115,ALLEGRO_ALIGN_CENTER,"(N)
");*/

   //al_draw_scaled_bitmap(FLECHAS, 0, 0, 360, 360, 850, 496,210, 210, 0);

//confirma la acción
   /*al_draw_filled_rounded_rectangle(225, 100, 890, 675, 25, 25,
al_map_rgba(0, 0,0, 160));//222, 186, 192, 1
   al_draw_text(lexend_regular[40],al_map_rgb(255, 255, 255),560, 300,
ALLEGRO_ALIGN_CENTER,"¿Quieres confirmar la acción?");//deseas eliminar este
elemento?

   al_draw_filled_rounded_rectangle(425, 425, 525, 525, 25, 25,
al_map_rgba(0, 0,0, 160));//222, 186, 192, 1
   al_draw_filled_rounded_rectangle(625, 425, 725, 525, 25, 25,
al_map_rgba(0, 0,0, 160));//222, 186, 192, 1
```

```
   al_draw_text(lexend_regular[20],al_map_rgb(255, 255, 255),475, 450,
ALLEGRO_ALIGN_CENTER,"Sí");
   al_draw_text(lexend_regular[20],al_map_rgb(255, 255, 255),475, 475,
ALLEGRO_ALIGN_CENTER,"(Enter)");

   al_draw_text(lexend_regular[20],al_map_rgb(255, 255, 255),675, 450,
ALLEGRO_ALIGN_CENTER,"No");
   al_draw_text(lexend_regular[20],al_map_rgb(255, 255, 255),675, 475,
ALLEGRO_ALIGN_CENTER,"(Esc)");*/

   //mensaje "Parece que no tienes hábitos activos"
   ObtenerHora();
   ventanaActual();
   //IMPRIME NOMBRE USUARIO:

al_draw_text(lexend_regular[10],fondo_gris1,1100,645,ALLEGRO_ALIGN_CENTER,"U
SUARIO");

al_draw_text(lexend_regular[14],texto_black,1100,655,ALLEGRO_ALIGN_CENTER,us
uario.nombre);

   al_flip_display();
}
void inicializar_rutas_usuario(char * id_string_usuario){
   strcat(rutaDIFICULTAD, id_string_usuario);
   strcat(rutaDIFICULTAD, frag_2rutaDIFICULTAD);

   strcat(rutaTIPO, id_string_usuario);
   strcat(rutaTIPO, frag_2rutaTIPO);

   strcat(rutaHABITO, id_string_usuario);
   strcat(rutaHABITO, frag_2rutaHABITO);

   strcat(rutaREGISTROHABITO, id_string_usuario);
   strcat(rutaREGISTROHABITO, frag_2rutaREGISTROHABITO);

   strcat(rutaHORARIO, id_string_usuario);
   strcat(rutaHORARIO, frag_2rutaHORARIO);

   strcat(rutaHORA_HORARIO, id_string_usuario);
   strcat(rutaHORA_HORARIO, frag_2rutaHORA_HORARIO);

   strcat(rutaRECORDATORIO, id_string_usuario);
   strcat(rutaRECORDATORIO, frag_2rutaRECORDATORIO);

   strcat(rutaPRODUCTIVIDAD, id_string_usuario);
   strcat(rutaPRODUCTIVIDAD, frag_2rutaPRODUCTIVIDAD);
}
int obtenerNumeroRegistros(char * ruta, size_t tamanio){
   long int entero = sizeof(int);
   int bytesSoloRegistro = contadorBytesArch(ruta) - entero, registros=0;
   return registros = bytesSoloRegistro/tamanio;
}
```

```
void *crearArreglo(size_t tamanioElemento, int cantidadElementos) {
   int retorno=0;
   void *arreglo = NULL;
   if(cantidadElementos!=0){
       arreglo = malloc(tamanioElemento * cantidadElementos);
       //printf("Direccion asignada en malloc: %p\n", malloc(tamanioElemento
* cantidadElementos));
       if (arreglo == NULL) {
           printf("Error: No se pudo asignar memoria para el arreglo.\n");
           retorno = 1;
       }
   }
   return arreglo;
}
void *aumentarArreglo(void *arreglo, size_t tamanioElemento, int nuevoTamano)
{
   int retorno = 0;
   void *temp = realloc(arreglo, tamanioElemento * nuevoTamano);
   if (temp == NULL) {
       //printf("Error: No se pudo ajustar el tamaño del arreglo.\n");
       //free(arreglo); // Se libera el arreglo original si realloc falló
       retorno = 1;
   }
   return temp;
}
void cargar_registros_no_vacios(){//tipo
   n_cantidad_registros_disponibles=0;
   if(verificarExistenciaDeArchivo(rutaHABITO)){
       for(int i=0; i<n_reg_habitos; i++){
           if(Habitos[i].ID_habito != 0){
               n_cantidad_registros_disponibles++;
           }
           /*printf("------------------>Habitos: %i, %s, %s, %s, %i, %p, %s,
%p, %s, %i, %lli, %d/%d/%d\n",
                   Habitos[i].ID_habito, Habitos[i].nombre, Habitos[i].nota,
Habitos[i].repeticion_semanal, Habitos[i].repeticion, Habitos[i].ptr_fk_tipo,
                   Habitos[i].fk_tipo.tipo, Habitos[i].ptr_fk_difi,
Habitos[i].fk_difi.dificultad, Habitos[i].racha, Habitos[i].tiempo,
Habitos[i].fecha_ini.tm_mday, Habitos[i].fecha_ini.tm_mon,
Habitos[i].fecha_ini.tm_year);*/
       }
       //printf("----_---_- %i\n", n_cantidad_registros_disponibles);
       if(registrosInicializados == 0){
           arrPos = (int *) crearArreglo(sizeof(int),
n_cantidad_registros_disponibles);
           //printf("Tamaño del arreglo de posicion 1: %lli\n",
sizeof(arrPos)); //--Esto devuelve no el tamaño del arreglo que se creó, sino
al apuntador, por lo que es incorrecto qu e se intente saber el tamaño con
sizeof, NO HACER
           //printf("Tamaño del arreglo de posicion 2: %lli\n",
sizeof(*nuevoArreglo));// dEVUELVE el valor del primer indice del arreglo,
pues el nombre es [0] y se esta desreferenciando ese valor
```

```
            arrHab = (int *) crearArreglo(sizeof(int),
n_cantidad_registros_disponibles);
            registrosInicializados = 1;
        } else {
            int *nuevoArreglo = (int *)aumentarArreglo(arrPos, sizeof(int),
n_cantidad_registros_disponibles);
            arrPos = nuevoArreglo;
            int *nuevoArreglo2 = (int *)aumentarArreglo(arrHab, sizeof(int),
n_cantidad_registros_disponibles);
            arrHab = nuevoArreglo2;
        }
        for(int i=0, indice2=0; i<n_reg_habitos; i++){
            if(Habitos[i].ID_habito != 0){
                arrHab[indice2] = i;
                indice2++;
            }
        }
    }
}
void CARGAR_TODOS_LOS_REGISTROS(){
    int retorno = 0, i=0;
    if(registrosInicializados==0){
        //inicializa arreglos necesarios
        n_reg_dificultades = obtenerNumeroRegistros(rutaDIFICULTAD,
sizeof(DIFICULTAD));
        //printf("Registros: %i\n", n_reg_dificultades);
        Dificultades = (DIFICULTAD *) crearArreglo(sizeof(DIFICULTAD),
n_reg_dificultades);
        for(i = 0; i<n_reg_dificultades; i++){
            SELECT(rutaDIFICULTAD, &Dificultades[i], sizeof(DIFICULTAD), 1,
i+1);
            //printf("IDDDD: %i, dificultad: %s\n",
Dificultades[i].ID_dificultad, Dificultades[i].dificultad);
        }
        n_reg_tipo = obtenerNumeroRegistros(rutaTIPO, sizeof(TIPO));
        //printf("Registros: %i\n", n_reg_tipo);
        Tipos = (TIPO *) crearArreglo(sizeof(TIPO), n_reg_tipo);
        for(i = 0; i<n_reg_tipo; i++){
            SELECT(rutaTIPO, &Tipos[i], sizeof(TIPO), 1, i+1);
            //printf("IDDDD: %i, tipo: %s\n", Tipos[i].ID_tipo,
Tipos[i].tipo);
        }
        //printf("LOLIN.COM: %lli\n", sizeof(Tipos));
        //printf("IDDDD malo: %i, tipo: %s\n", Tipos[2].ID_tipo,
Tipos[2].tipo);
        //printf("IDDDD malo: %i, tipo: %s\n", Tipos[6].ID_tipo,
Tipos[6].tipo);
        if(verificarExistenciaDeArchivo(rutaHABITO)){
            n_reg_habitos = obtenerNumeroRegistros(rutaHABITO,
sizeof(HABITO));
            //printf("Registros: %i\n", n_reg_habitos);
            Habitos = (HABITO *) crearArreglo(sizeof(HABITO), n_reg_habitos);
            for(i = 0; i<n_reg_habitos; i++){
```

```
                SELECT(rutaHABITO, &Habitos[i], sizeof(HABITO), 1, i+1);
                /*printf("HAB: %i, %s, %s, %s, %i, %p, %s, %p, %s, %i, %lli,
%d/%d/%d\n",
                        Habitos[i].ID_habito, Habitos[i].nombre,
Habitos[i].nota, Habitos[i].repeticion_semanal, Habitos[i].repeticion,
Habitos[i].ptr_fk_tipo,
                        Habitos[i].fk_tipo.tipo, Habitos[i].ptr_fk_difi,
Habitos[i].fk_difi.dificultad, Habitos[i].racha, Habitos[i].tiempo,
Habitos[i].fecha_ini.tm_mday, Habitos[i].fecha_ini.tm_mon,
Habitos[i].fecha_ini.tm_year);*/
            }
        }
        if(verificarExistenciaDeArchivo(rutaREGISTROHABITO)){
            n_reg_reg_hab = obtenerNumeroRegistros(rutaREGISTROHABITO,
sizeof(REGISTRO_HABITOS));
            //printf("Registros: %i\n", n_reg_reg_hab);
            Reg_habitos = (REGISTRO_HABITOS *)
crearArreglo(sizeof(REGISTRO_HABITOS), n_reg_reg_hab);
            for(i = 0; i<n_reg_reg_hab; i++){
                SELECT(rutaREGISTROHABITO, &Reg_habitos[i],
sizeof(REGISTRO_HABITOS), 1, i+1);
                /*printf("RH: %i, %p, %s, %d, %d, %d, %i, %i, %i\n",
Reg_habitos[i].ID_RH, Reg_habitos[i].ptr_fk_habito,
Reg_habitos[i].fk_habito.nombre, Reg_habitos[i].fecha.tm_mday,
                        Reg_habitos[i].fecha.tm_mon,
Reg_habitos[i].fecha.tm_year, Reg_habitos[i].completado,
Reg_habitos[i].no_completado, Reg_habitos[i].fk_habito.ID_habito);*/
            }
        }
        /*
        n_reg_horario = obtenerNumeroRegistros(rutaHORARIO, sizeof(HORARIO));
        printf("Registros: %i\n", n_reg_horario);
        Horarios = (HORARIO *) crearArreglo(sizeof(HORARIO), n_reg_horario);
        for(i = 0; i<n_reg_horario; i++){
            SELECT(rutaHORARIO, &Horarios[i], sizeof(HORARIO), 1, i+1);
            printf("HORARIO: %i, %s, %s, %p, %i, %s, %d, %d, %d, %d, %d, %d,
%d, %d, %d\n",
                    Horarios[i].ID_horario, Horarios[i].nombre,
Horarios[i].repeticion_semanal, Horarios[i].ptr_fk_tipo,
                    Horarios[i].fk_tipo.ID_tipo, Horarios[i].fk_tipo.tipo,
Horarios[i].fecha_ini.tm_mday, Horarios[i].fecha_ini.tm_mon,
Horarios[i].fecha_ini.tm_year,
                    Horarios[i].fecha_final.tm_mday,
Horarios[i].fecha_final.tm_mon, Horarios[i].fecha_final.tm_year,
                    Horarios[i].alerta.tm_mday, Horarios[i].alerta.tm_mon,
Horarios[i].alerta.tm_year);
        }
        n_reg_hora_horario = obtenerNumeroRegistros(rutaHORA_HORARIO,
sizeof(HORA_HORARIO));
        printf("Registros: %i\n", n_reg_hora_horario);
        Hora_horarios = (HORA_HORARIO *) crearArreglo(sizeof(HORA_HORARIO),
n_reg_hora_horario);
        for(i = 0; i<n_reg_hora_horario; i++){
```

```
            SELECT(rutaHORA_HORARIO, &Hora_horarios[i], sizeof(HORA_HORARIO),
1, i+1);
            printf("HH: %i, %p, %i, %s, %lli, %d/%d/%d, %d/%d/%d\n",
Hora_horarios[i].ID_HH, Hora_horarios[i].ptr_fk_horario,
Hora_horarios[i].fk_horario.ID_horario,
                    Hora_horarios[i].fk_horario.nombre,
Hora_horarios[i].tiempo,
                    Hora_horarios[i].dia_h_ini.tm_mday,
Hora_horarios[i].dia_h_ini.tm_mon, Hora_horarios[i].dia_h_ini.tm_year,
                    Hora_horarios[i].h_final.tm_mday,
Hora_horarios[i].h_final.tm_mon, Hora_horarios[i].h_final.tm_year);
        }
        */
        /*
        n_reg_recordatorios = obtenerNumeroRegistros(rutaRECORDATORIO,
sizeof(RECORDATORIOS));
        printf("Registros: %i\n", n_reg_recordatorios);
        Recordatorios = (RECORDATORIOS *) crearArreglo(sizeof(RECORDATORIOS),
n_reg_recordatorios);
        for(i = 0; i<n_reg_recordatorios; i++){
            SELECT(rutaRECORDATORIO, &Recordatorios[i], sizeof(RECORDATORIOS),
1, i+1);
            printf("Record: %i, %s, %p, %i, %s, %lli, %d/%d/%d, %i\n",
Recordatorios[i].ID_recordatorio, Recordatorios[i].recordatorio,
Recordatorios[i].ptr_fk_tipo,
                    Recordatorios[i].fk_tipo.ID_tipo,
Recordatorios[i].fk_tipo.tipo, Recordatorios[i].tiempo,
                    Recordatorios[i].fecha.tm_mday,
Recordatorios[i].fecha.tm_mon, Recordatorios[i].fecha.tm_year,
Recordatorios[i].estado_comp);
        }
        n_reg_productividad = obtenerNumeroRegistros(rutaPRODUCTIVIDAD,
sizeof(PRODUCTIVIDAD));
        printf("Registros: %i\n", n_reg_productividad);
        Productividad = (PRODUCTIVIDAD *) crearArreglo(sizeof(PRODUCTIVIDAD),
n_reg_productividad);
        for(i = 0; i<n_reg_productividad; i++){
            SELECT(rutaPRODUCTIVIDAD, &Productividad[i],
sizeof(PRODUCTIVIDAD), 1, i+1);
            printf("PRODUCT: %i, %lli, %d/%d/%d, %i, %i\n",
Productividad[i].ID_product, Productividad[i].tiempo,
                    Productividad[i].fecha.tm_mday,
Productividad[i].fecha.tm_mon, Productividad[i].fecha.tm_year,
Productividad[i].habit, Productividad[i].racord);
        }*/
    }else{
        //Reinicializar arreglos necesarios
        if(verificarExistenciaDeArchivo(rutaHABITO)){
            n_reg_habitos = obtenerNumeroRegistros(rutaHABITO,
sizeof(HABITO));
            //printf("Nuevos Registros: %i\n", n_reg_habitos);
            HABITO * HabitosTemp = (HABITO *) aumentarArreglo(Habitos,
sizeof(HABITO), n_reg_habitos);
```

```
            Habitos = HabitosTemp;
            for(i = 0; i<n_reg_habitos; i++){
                SELECT(rutaHABITO, &Habitos[i], sizeof(HABITO), 1, i+1);
                /*
                printf(">>>>>%i, %s, %s, %s, %i, %p, %s, %p, %s, %i, %lli,
%d/%d/%d\n",
                        Habitos[i].ID_habito, Habitos[i].nombre,
Habitos[i].nota, Habitos[i].repeticion_semanal, Habitos[i].repeticion,
Habitos[i].ptr_fk_tipo,
                        Habitos[i].fk_tipo.tipo, Habitos[i].ptr_fk_difi,
Habitos[i].fk_difi.dificultad, Habitos[i].racha, Habitos[i].tiempo,
Habitos[i].fecha_ini.tm_mday, Habitos[i].fecha_ini.tm_mon,
Habitos[i].fecha_ini.tm_year);*/
            }
        }
        /*
        n_reg_reg_hab = obtenerNumeroRegistros(rutaREGISTROHABITO,
sizeof(REGISTRO_HABITOS));
        printf("Registros: %i\n", n_reg_reg_hab);
        Reg_habitos = (REGISTRO_HABITOS *)
crearArreglo(sizeof(REGISTRO_HABITOS), n_reg_reg_hab);
        for(i = 0; i<n_reg_reg_hab; i++){
            SELECT(rutaREGISTROHABITO, &Reg_habitos[i],
sizeof(REGISTRO_HABITOS), 1, i+1);
            printf("RH: %i, %p, %s, %d, %d, %d, %i, %i\n",
Reg_habitos[i].ID_RH, Reg_habitos[i].ptr_fk_habito,
Reg_habitos[i].fk_habito.nombre, Reg_habitos[i].fecha.tm_mday,
                    Reg_habitos[i].fecha.tm_mon, Reg_habitos[i].fecha.tm_year,
Reg_habitos[i].completado, Reg_habitos[i].no_completado);
        }
        n_reg_recordatorios = obtenerNumeroRegistros(rutaRECORDATORIO,
sizeof(RECORDATORIOS));
        printf("Registros: %i\n", n_reg_recordatorios);
        Recordatorios = (RECORDATORIOS *) crearArreglo(sizeof(RECORDATORIOS),
n_reg_recordatorios);
        for(i = 0; i<n_reg_recordatorios; i++){
            SELECT(rutaRECORDATORIO, &Recordatorios[i], sizeof(RECORDATORIOS),
1, i+1);
            printf("Record: %i, %s, %p, %i, %s, %lli, %d/%d/%d, %i\n",
Recordatorios[i].ID_recordatorio, Recordatorios[i].recordatorio,
Recordatorios[i].ptr_fk_tipo,
                    Recordatorios[i].fk_tipo.ID_tipo,
Recordatorios[i].fk_tipo.tipo, Recordatorios[i].tiempo,
                    Recordatorios[i].fecha.tm_mday,
Recordatorios[i].fecha.tm_mon, Recordatorios[i].fecha.tm_year,
Recordatorios[i].estado_comp);
        }
        n_reg_productividad = obtenerNumeroRegistros(rutaPRODUCTIVIDAD,
sizeof(PRODUCTIVIDAD));
        printf("Registros: %i\n", n_reg_productividad);
        Productividad = (PRODUCTIVIDAD *) crearArreglo(sizeof(PRODUCTIVIDAD),
n_reg_productividad);
        for(i = 0; i<n_reg_productividad; i++){
```

```
           SELECT(rutaPRODUCTIVIDAD, &Productividad[i],
sizeof(PRODUCTIVIDAD), 1, i+1);
           printf("PRODUCT: %i, %lli, %d/%d/%d, %i, %i\n",
Productividad[i].ID_product, Productividad[i].tiempo,
                  Productividad[i].fecha.tm_mday,
Productividad[i].fecha.tm_mon, Productividad[i].fecha.tm_year,
Productividad[i].habit, Productividad[i].racord);
       }*/
   }
   cargar_registros_no_vacios();
   obtenerHabitosHoy();
}
void verificarREGISTROS(){
   int contadorHabito=3, i=3, difDias, habInd=3, diaCONREGISTRO=0;
   double difSegundos;
   // Obtener la fecha y hora actuales
   time_t tiempoActual;
   struct tm *infoTiempo;
   time(&tiempoActual);
   infoTiempo = localtime(&tiempoActual);
   FECHA fechaActual = *infoTiempo;

   //printf("\n********ID HABITO 3: %i\n",
Habitos[contadorHabito].ID_habito);
   //printf("\n**********ID REGISTRO_ HABITO 3: %i\n",
Reg_habitos[contadorHabito].ID_RH);
   while(Reg_habitos[i].fk_habito.ID_habito!=1) {
       //printf("\ndebug i=%i\n", i);
       i++;
   }
   /*printf("RH: %i, %p, %s, %d, %d, %d, %i, %i, %i, %d\n",
Reg_habitos[i].ID_RH, Reg_habitos[i].ptr_fk_habito,
Reg_habitos[i].fk_habito.nombre, Reg_habitos[i].fecha.tm_mday,
         Reg_habitos[i].fecha.tm_mon, Reg_habitos[i].fecha.tm_year,
Reg_habitos[i].completado, Reg_habitos[i].no_completado,
Reg_habitos[i].fk_habito.ID_habito,
Reg_habitos[i].fk_habito.fecha_ini.tm_mday);*/
   /*Comparar si los nombres coinciden de habito y reg habito*/

   for(int indHabito=0;indHabito<cantidadREGISTROS_HABITOS;indHabito++){
//        difSegundos = difftime(mktime(&fechaActual),
mktime(&Habitos[3].fecha_ini));
//        int diferenciaDias = (int)(difSegundos / (60 * 60 * 24));
//        printf("\n\nDIFERENCIAS DIAS: %i\n\n", diferenciaDias);

//        printf("\nhabitos: %s, registro: %s\n", Habitos[habInd].nombre,
Reg_habitos[indHabito].fk_habito.nombre);
       if(strcmp(Habitos[habInd].nombre,
Reg_habitos[indHabito].fk_habito.nombre)==0){
           int diferenciaDias = fechaActual.tm_mday -
Habitos[habInd].fecha_ini.tm_mday;
           int diasParaRegistro[diferenciaDias];
```

```
            for(int j=0;j<diferenciaDias;j++){/*for para llenar los espacios
con numeros consecutivos*/
                diasParaRegistro[j]=(j+1);
//                  printf("VALOR DE J:%i\n\n", j);
            }
//              printf("Regdia. %i", Reg_habitos[indHabito].fecha.tm_mday);
//              difSegundos = difftime(mktime(&fechaActual),
mktime(&Habitos[3].fecha_ini.tm_mday));
//              int diferenciaDias = (int)(difSegundos / (60 * 60 * 24));
            //printf("\n\nDIFERENCIAS DIAS: %i\n\n", diferenciaDias);
            //printf("\n\nCOINCIDE EL NOMBRE\n\n");
            //rellenarRegistrosHabitos(indHabito);
        }
    }
}
void resetearSoloEstadoMomento(int momentoACambiar){
        momento = momentoACambiar;
        estado = 0;
}
void reseteatEstadoMomento(int momentoACambiar){
    momento = momentoACambiar;
    estado = 0;
    altura_Y_registros = 180;
    loc = 0;
}
time_t convertirAtime_t(const char *cadenaFecha) {
    struct tm tiempo = {0}; // Crear una estructura tm para almacenar la fecha
    char *token;
    char copiaFecha[strlen(cadenaFecha) + 1];
    strcpy(copiaFecha, cadenaFecha);

    // Strtok divide la cadena de texto en tokens utilizando '/'
    token = strtok(copiaFecha, "/");
    tiempo.tm_mday = atoi(token); // Obtener y almacenar el día

    token = strtok(NULL, "/");
    tiempo.tm_mon = atoi(token) - 1; // Obtener y almacenar el mes (restar 1
porque en tm es de 0 a 11)

    token = strtok(NULL, "/");
    tiempo.tm_year = atoi(token) - 1900; // Obtener y almacenar el año (restar
1900 porque tm_year cuenta desde 1900)

    // Configurar otros valores en la estructura tm
    tiempo.tm_hour = 0;
    tiempo.tm_min = 0;
    tiempo.tm_sec = 0;
    tiempo.tm_isdst = -1; // Indicar que la información sobre horario de
verano es desconocida

    // Convertir la estructura tm a time_t
    time_t tiempoUnix = mktime(&tiempo);
```

```
   return tiempoUnix;
}
void analizar_fecha_habitos(){
   int n=1;
   //printf("\nanalizar_fechahabitos\n");
   HABITO habEj = {0};
   cargar_registros_no_vacios();
   //printf("\nANALIZAR #registros:
%i-------------------------------------------------------\n",
n_reg_habitos);
   //printf("\nRegistro:");
   /*printf("RH: %i, %p, %s, %d, %d, %d, %i, %i, %i\n", Reg_habitos[n].ID_RH,
Reg_habitos[n].ptr_fk_habito, Reg_habitos[n].fk_habito.nombre,
Reg_habitos[n].fecha.tm_mday,
         Reg_habitos[n].fecha.tm_mon, Reg_habitos[n].fecha.tm_year,
Reg_habitos[n].completado, Reg_habitos[n].no_completado,
Reg_habitos[n].fk_habito.ID_habito);*/
//    SELECT(rutaHABITO, );
   verificarREGISTROS();
}
void main_habitus(int verif_iniciador_primera_vez, int ultimo_usuario){
   int pantalla_requiere_actualizacion=1;
   char usuarioString[100], frag_1RutaUsuario;
   momento=verif_iniciador_primera_vez;//Si es 0, es que no se ha iniciado la
aplicacion ni una vez
   if(verif_iniciador_primera_vez == 1){
       SELECT(rutaUSUARIO, &usuario, sizeof(USUARIO), 1, ultimo_usuario);
       //printf("USERR: %i, %s", usuario.ID_usuario, usuario.nombre);
   }
   //printf("Init: %i, Usuario: %i\n", momento, ultimo_usuario);
   itoa(ultimo_usuario, usuarioString, 10);
   inicializar_rutas_usuario(usuarioString);
   //printf("%s\n%s\n%s\n%s\n%s\n%s\n%s\n%s\n", rutaDIFICULTAD, rutaTIPO,
rutaHABITO, rutaREGISTROHABITO, rutaHORARIO, rutaHORA_HORARIO,
rutaRECORDATORIO, rutaPRODUCTIVIDAD);
   //momento=-1;//DEP
   int resetearCadena=0;
   cargarDiaDeLaSemana();
   CARGAR_TODOS_LOS_REGISTROS();
   analizar_fecha_habitos();

   while(fin!=1){
       if(al_event_queue_is_empty(cola_eventos) &&
pantalla_requiere_actualizacion){
           //pantalla_requiere_actualizacion=0;
           actualizar_display();
       }
       else if(!al_event_queue_is_empty(cola_eventos)){
           //pantalla_requiere_actualizacion = 1;
       }
       //EVENTOS SUCEDIENDO:
       al_wait_for_event(cola_eventos, &evento);
```

```
        if(evento.type == ALLEGRO_EVENT_DISPLAY_CLOSE){
            //funcion de confirmacion() --TODO
            fin = 1;
        }else if(evento.type == ALLEGRO_EVENT_TIMER){
            actualizar_display();
        }
        else if(evento.type == ALLEGRO_EVENT_DISPLAY_SWITCH_OUT){//Evento de
que perdiste el foco de la ventana
            //printf("PERDISTE EL FOCO\n");
        }
        else if(evento.type == ALLEGRO_EVENT_DISPLAY_SWITCH_IN){//Evento de
que retomaste el foco de la ventana
            //printf("RECUPERASTE EL FOCO\n");
        }
        else{//Si no fueron eventos generales de la ventana:
            switch (momento) {
                case -1: //Depuracion
                    switch (evento.type) {
                        case ALLEGRO_EVENT_KEY_DOWN:
                            switch (evento.keyboard.keycode) {
                                case ALLEGRO_KEY_BACKSPACE:
                                    //INSERT(rutaAPP, &reseteoAPP,
sizeof(struct APP), 1);
                                    break;
                                case ALLEGRO_KEY_F:

                                    break;
                                default:
                                    break;
                            }
                            break;
                        default:
                            break;
                    }
                    break;
                case 0:
                    if (evento.type == ALLEGRO_EVENT_KEY_CHAR) {
                        if(estado==0)
                            estado=1;
                        if(estado==1){
                            if (evento.keyboard.unichar >= 32 &&
evento.keyboard.unichar <= 126) {
                                // Añadir el carácter a la cadena de entrada
                                int len = strlen(nombre);
                                if (len < sizeof(nombre) - 1) {
                                    nombre[len] = evento.keyboard.unichar;
                                    nombre[len + 1] = '\0';
                                }
                            } else if (evento.keyboard.keycode ==
ALLEGRO_KEY_BACKSPACE) {
                                // Borrar el último carácter de la `cadena` de
entrada
                                int len = strlen(nombre);
```

```
                                if (len > 0) {
                                    nombre[len - 1] = '\0';
                                }
                            }else if(evento.keyboard.keycode==
ALLEGRO_KEY_ENTER){
                                //Aqui se ingresa el nombre del usuario
                                //UPDATE
                                strcpy(usuario.nombre,nombre);//Ponerle el
nuevo nombre "" -> "nuevo"

if(verificarExistenciaDeArchivo(rutaUSUARIO)==0){
                                    SUPER_INSERT(&usuario.ID_usuario,
rutaUSUARIO, &usuario, sizeof(USUARIO), 1);
                                } else {
                                    UPDATE(rutaUSUARIO,&usuario,sizeof
(USUARIO),1,1);//Por el programa que estamos desarrollando de solo 1 usuario;
                                }
                                //printf("%s",usuario.nombre);
                                //Actualiza archivo de ingresado por primera
vez:
                                struct APP appActualizado = {1,
usuario.ID_usuario};
                                INSERT(rutaAPP, &appActualizado, sizeof(struct
APP), 1);
                                resetearCadena=0;
                                reseteatEstadoMomento(1);
                            }
                        }
                        //al_draw_text(lexend_regular[30], texto_white, 600,
340, ALLEGRO_ALIGN_CENTER, nombre);
                        //al_flip_display();
                    }
                    break;
                case 1: /*Habitos*/
//<<<<<<< Updated upstream --Fersa estuvo áqui 28/11/2023
/*
                    creacionEstructuras();
                    CONTAR_REGISTROS();
//=======
//>>>>>>> Stashed changes
                    //al_flip_display();
                    /*Flechitas arriba y abajo para cambiar de habito*/
                    if(estado==0){
                        //printf("===LOCCCOCK:%i, %i, %s \n", arrHab[loc],
loc, Habitos[arrHab[loc]].nombre);
                        /*
                        printf("------------------>Habitos: %i, %s, %s, %s,
%i, %p, %s, %p, %s, %i, %lli, %d/%d/%d\n",
                               Habitos[indice2].ID_habito, Habitos[i].nombre,
Habitos[i].nota, Habitos[i].repeticion_semanal, Habitos[i].repeticion,
Habitos[i].ptr_fk_tipo,
                               Habitos[i].fk_tipo.tipo,
Habitos[i].ptr_fk_difi, Habitos[i].fk_difi.dificultad, Habitos[i].racha,
```

```
Habitos[i].tiempo, Habitos[i].fecha_ini.tm_mday, Habitos[i].fecha_ini.tm_mon,
Habitos[i].fecha_ini.tm_year);

                                */
                        switch(evento.type){
                            case ALLEGRO_EVENT_KEY_DOWN:
                                switch(evento.keyboard.keycode){
                                    //ESTADO 0 -> lectura
                                    case ALLEGRO_KEY_DOWN:

if(loc<(n_cantidad_registros_disponibles-1) && loc>=0){
                                            altura_Y_registros-=300;
                                            loc++;
                                        }
                                        break;
                                    case ALLEGRO_KEY_UP:
                                        if(loc>0 &&
loc<=n_cantidad_registros_disponibles){
                                            loc--;
                                            altura_Y_registros+=300;
                                        }
                                        break;
                                    case ALLEGRO_KEY_A:
                                        estado = 1;
                                        break;
                                    case ALLEGRO_KEY_C:
                                        //printf("\nArrHab: %d, %i\n",
Habitos[arrHab[loc]].racha, arrHab[loc]);

if(Habitos[arrHab[loc]].repeticion_semanal[diaDeLaSemana] == '1'){
                                            Habitos[arrHab[loc]].racha = 1;
                                            //printf("\nArrHab: %d\n",
Habitos[arrHab[loc]].racha);
                                            UPDATE(rutaHABITO,
&Habitos[arrHab[loc]], sizeof(HABITO), 1, arrHab[loc]+1);
                                            resetearSoloEstadoMomento(1);
                                            //printf("\nArrHab: %d\n",
Habitos[arrHab[loc]].racha);
                                            CARGAR_TODOS_LOS_REGISTROS();
                                            //printf("\nArrHab: %d\n",
Habitos[arrHab[loc]].racha);
                                        } else {
                                            mensajeAdvertencia = 2;
                                        }
                                        break;
                                    case ALLEGRO_KEY_N:

if(Habitos[arrHab[loc]].repeticion_semanal[diaDeLaSemana] == '1'){
                                            Habitos[arrHab[loc]].racha = 0;
                                            UPDATE(rutaHABITO,
&Habitos[arrHab[loc]], sizeof(HABITO), 1, arrHab[loc]+1);
                                            resetearSoloEstadoMomento(1);
                                            CARGAR_TODOS_LOS_REGISTROS();
```

```
                                        //printf("Racha actual: %d",
Habitos[arrHab[loc]].racha);
                                    } else {
                                        mensajeAdvertencia = 2;
                                    }

                                    break;
                                case ALLEGRO_KEY_B:
                                    estado = 5;
                                    break;
                                case ALLEGRO_KEY_2:
                                    reseteatEstadoMomento(2);
                                    break;
                                case ALLEGRO_KEY_3:
                                    reseteatEstadoMomento(3);
                                    break;
                                case ALLEGRO_KEY_4:
                                    reseteatEstadoMomento(4);
                                    break;
                                case ALLEGRO_KEY_ESCAPE:
                                    mensajeAdvertencia = 0;
                                    break;
                            }
                            break;
                        default:
                            break;
                    }
                }
                else if (estado == 1){
                    //printf("\nESTADO 1 habitos \n");
                    if(resetearCadena==0){
                        strcpy(Titulo,"");
                        resetearCadena=1;
                    }
                    if (evento.type == ALLEGRO_EVENT_KEY_CHAR) {
                        if (evento.keyboard.unichar >= 32 &&
evento.keyboard.unichar <= 126) {
                            int len = strlen(Titulo);
                            if (len < sizeof(Titulo) - 1) {
                                Titulo[len] = evento.keyboard.unichar;
                                Titulo[len + 1] = '\0';
                            }
                        } else if (evento.keyboard.keycode ==
ALLEGRO_KEY_BACKSPACE) {
                            int len = strlen(Titulo);
                            if (len > 0) {
                                Titulo[len - 1] = '\0';
                            }
                        }else if(evento.keyboard.keycode==
ALLEGRO_KEY_ENTER){
                            resetearCadena=0;
                            estado=2;
```

```
                            }else if(evento.keyboard.keycode ==
ALLEGRO_KEY_ESCAPE){
                                resetearCadena=0;
                                estado = 0;
                            }
                        }
                    }else if (estado == 2){
                        //printf("\nESTADO 2 habitos\n");
                        if(resetearCadena==0){
                            strcpy(notas,"");
                            resetearCadena=1;
                        }
                        if (evento.type == ALLEGRO_EVENT_KEY_CHAR) {
                            if (evento.keyboard.unichar >= 32 &&
evento.keyboard.unichar <= 126) {
                                int len = strlen(notas);
                                if (len < sizeof(notas) - 1) {
                                    notas[len] = evento.keyboard.unichar;
                                    notas[len + 1] = '\0';
                                }
                            } else if (evento.keyboard.keycode ==
ALLEGRO_KEY_BACKSPACE) {
                                int len = strlen(notas);
                                if (len > 0) {
                                    notas[len - 1] = '\0';
                                }
                            }else if(evento.keyboard.keycode==
ALLEGRO_KEY_ENTER){
                                resetearCadena=0;
                                estado=3;
                            }else if(evento.keyboard.keycode ==
ALLEGRO_KEY_ESCAPE){
                                estado = 0;
                                resetearCadena=0;
                            }
                        }
                    }else if(estado==3){
                        if(evento.type==ALLEGRO_EVENT_KEY_CHAR){
                            if(resetearCadena==0){
                                strcpy(semana,"0000000");
                                resetearCadena=1;
                            }
                            if(evento.keyboard.keycode==ALLEGRO_KEY_1){
                                if(semana[0]=='0') semana[0]='1';
                                else semana[0]='0';
                            }else if(evento.keyboard.keycode==ALLEGRO_KEY_2){
                                if(semana[1]=='0')semana[1]='1';
                                else semana[1]='0';
                            }else if(evento.keyboard.keycode==ALLEGRO_KEY_3){
                                if(semana[2]=='0')semana[2]='1';
                                else semana[2]='0';
                            }else if(evento.keyboard.keycode==ALLEGRO_KEY_4){
                                if(semana[3]=='0')semana[3]='1';
```

```
                                else semana[3]='0';
                            }else if(evento.keyboard.keycode==ALLEGRO_KEY_5){
                                if(semana[4]=='0')semana[4]='1';
                                else semana[4]='0';
                            }else if(evento.keyboard.keycode==ALLEGRO_KEY_6){
                                if(semana[5]=='0')semana[5]='1';
                                else semana[5]='0';
                            }else if(evento.keyboard.keycode==ALLEGRO_KEY_7){
                                if(semana[6]=='0')semana[6]='1';
                                else semana[6]='0';
                            }else if(evento.keyboard.keycode==
ALLEGRO_KEY_ENTER){
                                resetearCadena=0;
                                estado=4;
                            }else if(evento.keyboard.keycode ==
ALLEGRO_KEY_ESCAPE) {
                                estado = 0;
                                resetearCadena = 0;
                                strcpy(semana,"0000000");
                            }
                        }
                    }else if(estado==4){
                        if(evento.type==ALLEGRO_EVENT_KEY_CHAR){
                            if(resetearCadena==0){
                                y=400,y2=400,x3=475,y3=305,resetearCadena=1;
                            }
                            if(evento.keyboard.keycode==ALLEGRO_KEY_1){
                                y=405,y2=400,x3=400,y3=385;
                                dificultadHabito = 1;
                            }else if(evento.keyboard.keycode==ALLEGRO_KEY_2){
                                y=400,y2=400,x3=410,y3=335;
                                dificultadHabito = 2;
                            }else if(evento.keyboard.keycode==ALLEGRO_KEY_3){
                                y=400,y2=400,x3=475,y3=305;
                                dificultadHabito = 3;
                            }else if(evento.keyboard.keycode==ALLEGRO_KEY_4){
                                y=400,y2=400,x3=540,y3=335;
                                dificultadHabito = 4;
                            }else if(evento.keyboard.keycode==ALLEGRO_KEY_5){
                                y=400,y2=405,x3=550,y3=385;
                                dificultadHabito = 5;
                            }else if(evento.keyboard.keycode==
ALLEGRO_KEY_ENTER) {
                                DIFICULTAD difRecibe = {0};
                                SELECT(rutaDIFICULTAD, &difRecibe,
sizeof(DIFICULTAD), 1, dificultadHabito);
                                int repeticion = 4;
                                TIPO tipoRecibe = {0};
                                SELECT(rutaTIPO, &tipoRecibe, sizeof(TIPO), 1,
1);

                                time_t t;
                                struct tm info;
```

```
                                char fecha_actual[80];
                                time(&t);
                                localtime_s(&info, &t);

                                HABITO habNuevo = {0, "", "", "", repeticion,
NULL, tipoRecibe, NULL, difRecibe, 0,
                                                   time(NULL), info};
                                strcpy(habNuevo.nombre, Titulo);
                                strcpy(habNuevo.nota, notas);
                                strcpy(habNuevo.repeticion_semanal, semana);

                                SUPER_INSERT(&habNuevo.ID_habito, rutaHABITO,
&habNuevo, sizeof(HABITO), 1);
                                reseteatEstadoMomento(1);

                                CARGAR_TODOS_LOS_REGISTROS();

                                resetearCadena = 0;
                                y = 400;
                                y2 = 400;
                                x3 = 400;
                                y3 = 300;
                                dificultadHabito = 1;
                                estado=0;
                                //printf("Reg HABHAB: %i \n",
obtenerNumeroRegistros(rutaREGISTROHABITO, sizeof(REGISTRO_HABITOS)));
                            }else if(evento.keyboard.keycode ==
ALLEGRO_KEY_ESCAPE) {
                                estado = 0;
                                y=400;
                                y2=400;
                                x3=400;
                                y3=300;
                                resetearCadena = 0;
                                strcpy(semana,"0000000");
                                dificultadHabito = 1;
                            }
                        }
                    } else if(estado==5){//BORRAR HÁBITO
                        if (evento.keyboard.keycode == ALLEGRO_KEY_ENTER) {
                            DELETE(rutaHABITO, &habNULL, sizeof(HABITO), 1,
arrHab[loc]+1);
                            reseteatEstadoMomento(1);
                            CARGAR_TODOS_LOS_REGISTROS();
                        } else if (evento.keyboard.keycode ==
ALLEGRO_KEY_ESCAPE){
                            reseteatEstadoMomento(1);
                            //CARGAR_TODOS_LOS_REGISTROS();
                        }
                        //printf("Estas borrando: \t");
                        //printf("%i, %i, %s %i\n", arrHab[loc], loc,
Habitos[arrHab[loc]].nombre, loc);
                    }
```

```
                    //printf("Tamaño:%i\tLocalización:%i\n", tamArrPos, loc);
                    break;
                case 2:  /*Horario*/
                    if(estado == 0){
                        switch(evento.type){
                            case ALLEGRO_EVENT_KEY_DOWN:
                                switch(evento.keyboard.keycode){
                                    //ESTADO 0 -> lectura
                                    case ALLEGRO_KEY_DOWN:
                                        if(loc<tamArrPos && loc>=0){
                                            loc++;
                                        }
                                        break;
                                    case ALLEGRO_KEY_UP:
                                        if(loc>0 && loc<=tamArrPos){
                                            loc--;
                                        }
                                        break;
                                    case ALLEGRO_KEY_1:
                                        reseteatEstadoMomento(1);
                                        break;
                                    case ALLEGRO_KEY_3:
                                        reseteatEstadoMomento(3);
                                        break;

                                    case ALLEGRO_KEY_4:
                                        reseteatEstadoMomento(4);
                                        break;
                                }
                                break;
                            default:
                                break;
                        }
                    }  else if (estado == 1){
                        if (evento.type == ALLEGRO_EVENT_KEY_CHAR) {
                            if (evento.keyboard.unichar >= 32 &&
evento.keyboard.unichar <= 126) {

                            } else if (evento.keyboard.keycode ==
ALLEGRO_KEY_BACKSPACE) {

                            }
                            //al_draw_text(lexend_regular[20], texto_white,
550, 340, ALLEGRO_ALIGN_CENTER, nombre);
                            if(evento.keyboard.keycode== ALLEGRO_KEY_ENTER){

                            }
                            if(evento.keyboard.keycode ==
ALLEGRO_KEY_ESCAPE){
                                estado = 0;
                            }
                        }
                    } else if (estado == 2){
```

```
                        if (evento.type == ALLEGRO_EVENT_KEY_CHAR) {
                            if (evento.keyboard.unichar >= 32 &&
evento.keyboard.unichar <= 126) {

                            } else if (evento.keyboard.keycode ==
ALLEGRO_KEY_BACKSPACE) {

                            }
                            //al_draw_text(lexend_regular[20], texto_white,
550, 340, ALLEGRO_ALIGN_CENTER, nombre);
                            if(evento.keyboard.keycode== ALLEGRO_KEY_ENTER){

                            }
                            if(evento.keyboard.keycode ==
ALLEGRO_KEY_ESCAPE){
                                estado = 0;
                            }
                        }
                    } else if (estado == 3){
                        if (evento.type == ALLEGRO_EVENT_KEY_DOWN) {
                            if(evento.keyboard.keycode== ALLEGRO_KEY_ENTER){

                            }
                            if(evento.keyboard.keycode ==
ALLEGRO_KEY_ESCAPE){
                                estado = 0;
                            }
                        }
                    }
                    break;
                case 3:
                    if(estado == 0){
                        switch(evento.type){
                            case ALLEGRO_EVENT_KEY_DOWN:
                                switch(evento.keyboard.keycode){
                                    //ESTADO 0 -> lectura
                                    case ALLEGRO_KEY_DOWN:
                                        if(loc<tamArrPos && loc>=0){
                                            loc++;
                                        }
                                        break;
                                    case ALLEGRO_KEY_UP:
                                        if(loc>0 && loc<=tamArrPos){
                                            loc--;
                                        }
                                        break;
                                    case ALLEGRO_KEY_1:
                                        reseteatEstadoMomento(1);
                                        break;
                                    case ALLEGRO_KEY_2:
                                        reseteatEstadoMomento(2);
                                        break;
```

```
                                case ALLEGRO_KEY_4:
                                    reseteatEstadoMomento(4);
                                    break;
                                }
                                break;
                            default:
                                break;
                        }
                        break;
                    }  else if (estado == 1){
                        if (evento.type == ALLEGRO_EVENT_KEY_CHAR) {
                            if (evento.keyboard.unichar >= 32 &&
evento.keyboard.unichar <= 126) {

                            } else if (evento.keyboard.keycode ==
ALLEGRO_KEY_BACKSPACE) {

                            }
                            //al_draw_text(lexend_regular[20], texto_white,
550, 340, ALLEGRO_ALIGN_CENTER, nombre);
                            if(evento.keyboard.keycode== ALLEGRO_KEY_ENTER){

                            }
                            if(evento.keyboard.keycode ==
ALLEGRO_KEY_ESCAPE){
                                estado = 0;
                            }
                        }
                    } else if (estado == 2){
                        if (evento.type == ALLEGRO_EVENT_KEY_CHAR) {
                            if (evento.keyboard.unichar >= 32 &&
evento.keyboard.unichar <= 126) {

                            } else if (evento.keyboard.keycode ==
ALLEGRO_KEY_BACKSPACE) {

                            }
                            //al_draw_text(lexend_regular[20], texto_white,
550, 340, ALLEGRO_ALIGN_CENTER, nombre);
                            if(evento.keyboard.keycode== ALLEGRO_KEY_ENTER){

                            }
                            if(evento.keyboard.keycode ==
ALLEGRO_KEY_ESCAPE){
                                estado = 0;
                            }
                        }
                    } else if (estado == 3){
                        if (evento.type == ALLEGRO_EVENT_KEY_DOWN) {
                            if(evento.keyboard.keycode== ALLEGRO_KEY_ENTER){

                            }
```

```
                            if(evento.keyboard.keycode ==
ALLEGRO_KEY_ESCAPE){
                                estado = 0;
                            }
                        }
                    }
                case 4:
                    if(resetearCadena==0){
                        strcpy(nombre,usuario.nombre);
                        resetearCadena=1;
                    }
                    if (evento.type == ALLEGRO_EVENT_KEY_CHAR) {
                        if(evento.keyboard.keycode==ALLEGRO_KEY_4){
                            reseteatEstadoMomento(4);
                            resetearCadena=0;
                        }else if(evento.keyboard.keycode==ALLEGRO_KEY_1){
                            reseteatEstadoMomento(1);
                        }
                        else if(evento.keyboard.keycode==ALLEGRO_KEY_2){
                            reseteatEstadoMomento(2);
                        }
                        else if(evento.keyboard.keycode==ALLEGRO_KEY_3){
                            reseteatEstadoMomento(3);
                        }else if (evento.keyboard.unichar >= 32 &&
evento.keyboard.unichar <= 126) {
                            // Añadir el carácter a la cadena de entrada
                            int len = strlen(nombre);
                            if (len < sizeof(nombre) - 1) {
                                nombre[len] = evento.keyboard.unichar;
                                nombre[len + 1] = '\0';
                            }
                        } else if (evento.keyboard.keycode ==
ALLEGRO_KEY_BACKSPACE) {
                            // Borrar el último carácter de la cadena de
entrada
                            int len = strlen(nombre);
                            if (len > 0) {
                                nombre[len - 1] = '\0';
                            }
                        }
                        //al_draw_text(lexend_regular[20], texto_white, 550,
340, ALLEGRO_ALIGN_CENTER, nombre);
                        if(evento.keyboard.keycode== ALLEGRO_KEY_ENTER){
                            //Aqui se ingresa el nombre del usuario
                            strcpy(usuario.nombre,nombre);
                            UPDATE(rutaUSUARIO,&usuario,sizeof (USUARIO),1,1);
                            //printf("%s",usuario.nombre);
                            SELECT(rutaUSUARIO,&usuario,sizeof (USUARIO),1,1);
                            //printf("Seleccionando %s",usuario.nombre);
                            al_draw_filled_rectangle(100,0,1000,700,
fondo_principal_oscuro);
```

```
al_draw_text(lexend_regular[20],texto_white,550,340,ALLEGRO_ALIGN_CENTER,"Se
han guardado los cambios");
                            strcpy(nombre,"");
                            resetearCadena=0;
                            reseteatEstadoMomento(1);
                        }

                    }
                    break;
                default:
                    break;
            }
        }
    }
}
void IUSD(){
    int opcion;
    scanf("%i", &opcion);
    switch (opcion) {
        case 1:
            llamarINSERT();
            break;
        case 2:
            llamarUPDATE();
            break;
        case 3:
            llamarSELECT();
            break;
        case 4:
            llamarDELETE();
            break;
        default:
            break;
    }
}

void llamarINSERT(){}

void pedirDatosUPDATE(int opcion, int id){
    /*HABITO habit1 ={1, "HABITO 4 NUEVO OWO", "NOTA PARA 4", "2", 4, '\0',
'\0', 44, '\0', '\0'};
    habit1.ID_habito = manejarAUTOINCREMENT("./data/usuarios/1/habito.dat");*/
    HABITO newHab={0};
    //newHab.ID_habito = manejarAUTOINCREMENT("./data/usuarios/1/habito.dat");
    switch (opcion) {
        case 1:
            getchar();
            printf("NOMBRE DEL HABITO:");
            fgets(newHab.nombre, sizeof(newHab.nombre), stdin);
            printf("Nota del habito:");
            fgets(newHab.nota, sizeof(newHab.nota), stdin);
            printf("Repeticion semanal:");
```

```
            fgets(newHab.repeticion_semanal,
sizeof(newHab.repeticion_semanal), stdin);

            printf("Racha:");
            char racha[2];
            sprintf(racha,"%c", newHab.racha);
            fgets(racha, sizeof(newHab.racha), stdin);
            newHab.racha = atoi(racha);

            UPDATE("./data/usuarios/1/habito.dat", &newHab, sizeof(HABITO), 1,
id);
    }

}
void llamarUPDATE(){
    HABITO habNULL={0};
    USUARIO usuNULL={0};
    HORARIO  horNULL={0};
    RECORDATORIOS recNULL={0};
    int opcion, id;
    char *ruta[] ={};

    printf("Estructura:\n1.Habito\n2.Usuario\n3.Horario\n4.Recordatorio\n");
    scanf("%i", &opcion);

    printf("ID: ");
    scanf("%i", &id);
    switch (opcion) {
        case 1:
//            *ruta = "./data/usuarios/1/habito.dat";
//          DELETE("./data/usuarios/1/habito.dat", &habNULL, sizeof(HABITO),
1, id);
//            SELECT("./data/usuarios/1/habito.dat", &habNULL,
sizeof(HABITO), 1, id);
            pedirDatosUPDATE(opcion, id);
            break;
        case 2:
            *ruta = "./data/usuarios/1/usuario.dat";
            break;
        case 3:
            *ruta = "./data/usuarios/1/horario.dat";
            break;
        case 4:
            *ruta = "./data/usuarios/1/recordatorio.dat";
            break;
        default:
            break;
    }
}
void llamarSELECT(){
    HABITO habNULL={0};
    USUARIO usuNULL={0};
    HORARIO  horNULL={0};
```

```
   RECORDATORIOS recNULL={0};
   int opcion, id;
   char *ruta[] ={};

   printf("Estructura:\n1.Habito\n2.Usuario\n3.Horario\n4.Recordatorio\n");
   scanf("%i", &opcion);

   printf("ID: ");
   scanf("%i", &id);
   switch (opcion) {
       case 1:
//             *ruta = "./data/usuarios/1/habito.dat";
//             DELETE("./data/usuarios/1/habito.dat", &habNULL,
sizeof(HABITO), 1, id);
           //SELECT("./data/usuarios/1/habito.dat", &habNULL, sizeof(HABITO),
1, id);
           //printf("Nombre: %s\nNota:%s\nID:%i\n", habNULL.nombre,
habNULL.nota, habNULL.ID_habito);
           break;
       case 2:
           *ruta = "./data/usuarios/1/usuario.dat";
           break;
       case 3:
           *ruta = "./data/usuarios/1/horario.dat";
           break;
       case 4:
           *ruta = "./data/usuarios/1/recordatorio.dat";
           break;
       default:
           break;
   }
}
void llamarDELETE(){
   int opcion, id;
   char *ruta[] ={};

   printf("Estructura:\n1.Habito\n2.Usuario\n3.Horario\n4.Recordatorio\n");
   scanf("%i", &opcion);

   printf("ID: ");
   scanf("%i", &id);
   switch (opcion) {
       case 1:
//             *ruta = "./data/usuarios/1/habito.dat";
           DELETE("./data/usuarios/1/habito.dat", &habNULL, sizeof(HABITO),
1, id);
//             SELECT("./data/usuarios/1/habito.dat", &habNULL,
sizeof(HABITO), 1, id);
           break;
       case 2:
           *ruta = "./data/usuarios/1/usuario.dat";
           break;
       case 3:
```

```
            *ruta = "./data/usuarios/1/horario.dat";
            break;
        case 4:
            *ruta = "./data/usuarios/1/recordatorio.dat";
            break;
        default:
            break;
    }
}

void creacionEstructuras(){ //VISUALES
    int x=100;
    int separacionEntreRegistro=0;
    if(n_cantidad_registros_disponibles == 0){
        al_draw_text(lexend_regular[30],al_map_rgb(255, 255, 255),575, 200,
ALLEGRO_ALIGN_CENTER,"Parece que no tienes hábitos activos");
        //mensaje "Parece que no tienes recordatorios activos"
        al_draw_text(lexend_regular[30],al_map_rgb(255, 255, 255),575, 300,
ALLEGRO_ALIGN_CENTER,"Parece que no tienes recordatorios activos");
    }
    for(int n_habito = 0; n_habito < n_cantidad_registros_disponibles;
n_habito++ ){
        ALLEGRO_COLOR colorContenedorPrincipal =
(Habitos[arrHab[n_habito]].racha != 0)? al_map_rgb(119, 221, 119):
al_map_rgb(255, 105, 97);
        al_draw_filled_rectangle(x+75,
(altura_Y_registros+separacionEntreRegistro)-5, x+825,
(altura_Y_registros+separacionEntreRegistro)+220, al_map_rgb(74, 63,
75));//Cuadro principal habito

        al_draw_filled_rectangle(x+90,
(altura_Y_registros+separacionEntreRegistro)+10, x+810,
(altura_Y_registros+separacionEntreRegistro)+55, al_map_rgb(227, 158,
193));//Cuadro de titulo habito
        al_draw_filled_rectangle(x+90,
(altura_Y_registros+separacionEntreRegistro)+70, x+440,
(altura_Y_registros+separacionEntreRegistro)+190, al_map_rgb(227, 158,
193));//Cuadro contenedor botonera
        al_draw_filled_rectangle(x+450,
(altura_Y_registros+separacionEntreRegistro)+70, x+810,
(altura_Y_registros+separacionEntreRegistro)+190, al_map_rgb(227, 158,
193));//Cuadro contenedor principal notas y semana

        al_draw_filled_rectangle(x+450,
(altura_Y_registros+separacionEntreRegistro)+150, x+810,
(altura_Y_registros+separacionEntreRegistro)+190, al_map_rgb(225, 0,
129));//Cuadro contenedor semana

        //al_draw_filled_rectangle(x+120,
(altura_Y_registros+separacionEntreRegistro)+90, x+190,
(altura_Y_registros+separacionEntreRegistro)+160, al_map_rgb(119, 221,
119));//Boton completado
```

```
	//al_draw_filled_rectangle(x+340,
(altura_Y_registros+separacionEntreRegistro)+90, x+410,
(altura_Y_registros+separacionEntreRegistro)+160, al_map_rgb(255, 105,
97));//Boton No completadoal_draw_filled_rectangle(x+120,
(altura_Y_registros+separacionEntreRegistro)+90, x+190,
(altura_Y_registros+separacionEntreRegistro)+160, al_map_rgb(119, 221,
119));//Boton completado
	al_draw_filled_rectangle(x+120,
(altura_Y_registros+separacionEntreRegistro)+90, x+190,
(altura_Y_registros+separacionEntreRegistro)+160,
neutro3_french_lilac);//Boton completado
	al_draw_filled_rectangle(x+340,
(altura_Y_registros+separacionEntreRegistro)+90, x+410,
(altura_Y_registros+separacionEntreRegistro)+160,
neutro1_tinta_de_pulpo);//Boton No completado
	//Completado / No completado
	al_draw_filled_rectangle(x + 240, (altura_Y_registros +
separacionEntreRegistro) + 105, x + 290,
				(altura_Y_registros +
separacionEntreRegistro) + 150,
				colorContenedorPrincipal);//Cuadro pendientes
	al_draw_text(lexend_regular[10],texto_black,x+260,(altura_Y_registros
+ separacionEntreRegistro) +
90,ALLEGRO_ALIGN_CENTER,(Habitos[arrHab[n_habito]].racha == 0)?
"Completado":"No completado");


al_draw_text(lexend_regular[10],texto_black,x+155,(altura_Y_registros+separac
ionEntreRegistro)+115,ALLEGRO_ALIGN_CENTER,"Completado");

al_draw_text(lexend_regular[10],texto_black,x+155,(altura_Y_registros+separac
ionEntreRegistro)+125,ALLEGRO_ALIGN_CENTER,"(C)");

al_draw_text(lexend_regular[9],texto_white,x+375,(altura_Y_registros+separaci
onEntreRegistro)+115,ALLEGRO_ALIGN_CENTER,"No completado");

al_draw_text(lexend_regular[10],texto_white,x+375,(altura_Y_registros+separac
ionEntreRegistro)+125,ALLEGRO_ALIGN_CENTER,"(N)");
	//printf("N habitoooOOoo%i\n\n", n_habito);
	al_draw_textf(lexend_bold[40], texto_black, x+450,
(altura_Y_registros+separacionEntreRegistro)+10, ALLEGRO_ALIGN_CENTER, "%s",
Habitos[arrHab[n_habito]].nombre);//titulo habito
	al_draw_textf(lexend_regular[15], texto_black, x+630,
(altura_Y_registros+separacionEntreRegistro)+80, ALLEGRO_ALIGN_CENTER,
"Notas:");
	al_draw_textf(lexend_regular[15], texto_black, x+630,
(altura_Y_registros+separacionEntreRegistro)+100, ALLEGRO_ALIGN_CENTER,
"%s",Habitos[arrHab[n_habito]].nota);
	int CalX = x+540, CalY =
(altura_Y_registros+separacionEntreRegistro)+165;
	al_draw_text(lexend_regular[15], texto_black, CalX, CalY-16,
ALLEGRO_ALIGN_LEFT, "Do  Lu  Ma  Mi  Ju  Vi  Sa");
	for (int i = 0; i < 4; ++i) {
```

```
            char cadena[7]={0};
            strcpy(cadena,Habitos[arrHab[n_habito]].repeticion_semanal);
            for(int j=0;i<7;i++){
                int valor=cadena[i]-48;
                //printf("%d\n",valor);
                colorearDia(CalX,CalY,valor);
                CalX+=30;
            }
        }
        separacionEntreRegistro+=300;
    }
}
int main() {
    //int acceso;
    if(inicializar_allegro()){
        disp = al_create_display(1200, 700);
        AFK = al_create_timer(30);
        al_set_window_title(disp, "Hábitus");
        al_set_display_icon(disp, LOGO);
        cola_eventos = al_create_event_queue();

        al_register_event_source(cola_eventos,al_get_timer_event_source(AFK));
// FUENTE: eventos de tipo temporizador
        al_register_event_source(cola_eventos,
al_get_display_event_source(disp)); // FUENTE: eventos de la ventana
        al_register_event_source(cola_eventos,
al_get_keyboard_event_source());// FUENTE: eventos del teclado
        al_start_timer(AFK);

        /*CREACIÓN DE ESTRUCTURAS*/
        //creacionEstructuras();
        if(!verificarExistenciaDeArchivo(rutaAPP)){
            INSERT(rutaAPP, &reseteoAPP, sizeof(struct APP), 1);
        }
        SELECT(rutaAPP, &app_recibe, sizeof(struct APP), 1, 1);
        main_habitus(app_recibe.init, app_recibe.ID_last_user);
        al_destroy_event_queue(cola_eventos);
        al_destroy_display(disp);
        al_destroy_timer(AFK);
    }
    else{
        printf("\nOcurrio un error");// --TODO hacer un archivo txt que
registre todos los errores
    }
    return 0;
}
int inicializar_allegro() {
    int verif_todo_ok = 1;
    if (!al_init()) {
        printf("No se pudo iniciar Allegro");
    }
    if (!al_init_primitives_addon()) {
        printf("No se iniciaron las primitivas");
```

```
        verif_todo_ok = 0;
    }
    if (!al_install_keyboard()) {
        printf("No se instalo el teclado");
        verif_todo_ok = 0;
    }
    if (!al_init_image_addon()) {
        printf("No se inicio image addon");
        verif_todo_ok = 0;
    }
    if (!al_install_audio()) {//SONIDO
        printf("No se cargo el complemento de audio");
        verif_todo_ok = 0;
    }
    if (!al_init_acodec_addon()) {//SONIDO
        printf("No se pudo cargar el complemento de codex");
        verif_todo_ok = 0;
    }
    if (!al_init_font_addon() || !al_init_ttf_addon()) {
        printf("No se pudo cargar el complemento de fuentes");
        verif_todo_ok = 0;
    }
    if (!init_resources()) {
        printf("Error al iniciar los recursos fuentes");
        verif_todo_ok = 0;
    }
    return verif_todo_ok;

    }
```

PRESENTACIÓN

Para la presentación del proyecto realizada la última semana de clases:
https://drive.google.com/file/d/1Nu3O0pyyYZ2L6BwguuGm7WnpZXmj7GaY/view?usp=sharing

VIDEO DE FUNCIONAMIENTO

Para el ver el funcionamiento general de la aplicación, ingrese al siguiente link para ver un video demostración:
https://drive.google.com/file/d/1yHqwdv-ZVzpqdXDeL3iUtDgFTiNnaqX1/view?usp=drive_link