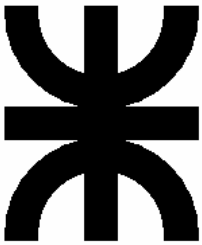


UNIVERSIDAD TECNOLOGICA NACIONAL
FACULTAD REGIONAL SAN NICOLAS

INGENIERIA EN ELECTRONICA

PROBLEMA DE INGENIERÍA

TECNICAS DIGITALES III

TEST DE REACCION

Integrantes**:**

- Bedini Juan Manuel
- Durán Julio Ricardo
- Robles Francisco Iván
- Fangio Juan Manuel

Docentes**:**

- Profesor: Poblete Felipe
- Auxiliar: González Mariano

AÑO 2007

INDICE

OBJETIVOS DEL TRABAJO

El proyecto desarrollado por este grupo de trabajo consiste en un sistema por medio del cual se llevan a cabo distintos tipos de tests para comprobar las aptitudes de los conductores, ya sean reflejo de los mismos, predicción de eventos, etc. Este sistema está basado en una computadora personal y pulsadores conectados al puerto paralelo. El programa se realizó en DEVC++, y utiliza una librería especial, denominada ALLEGRO, por medio de la cual se ha implementado toda la parte gráfica.

MATERIAS INTEGRADAS

• Informática I y II
• Técnicas Digitales III

POSIBLES APLICACIONES

• Test de reacción para obtener la licencia de conducir.

PROFESORES ENTREVISTADOS

• Vota Ramiro (Librerias graficas de C.)

BIBLIOGRAFÍA

• *Sitios de Internet.*
  • http://gda.utp.edu.co/instalacion_devcpp.html
  • http://alleg.sourceforge.net/index.es.html
  • http://www.gillius.org/allegtut/index.htm

DESARROLLO

INTRODUCCIÓN

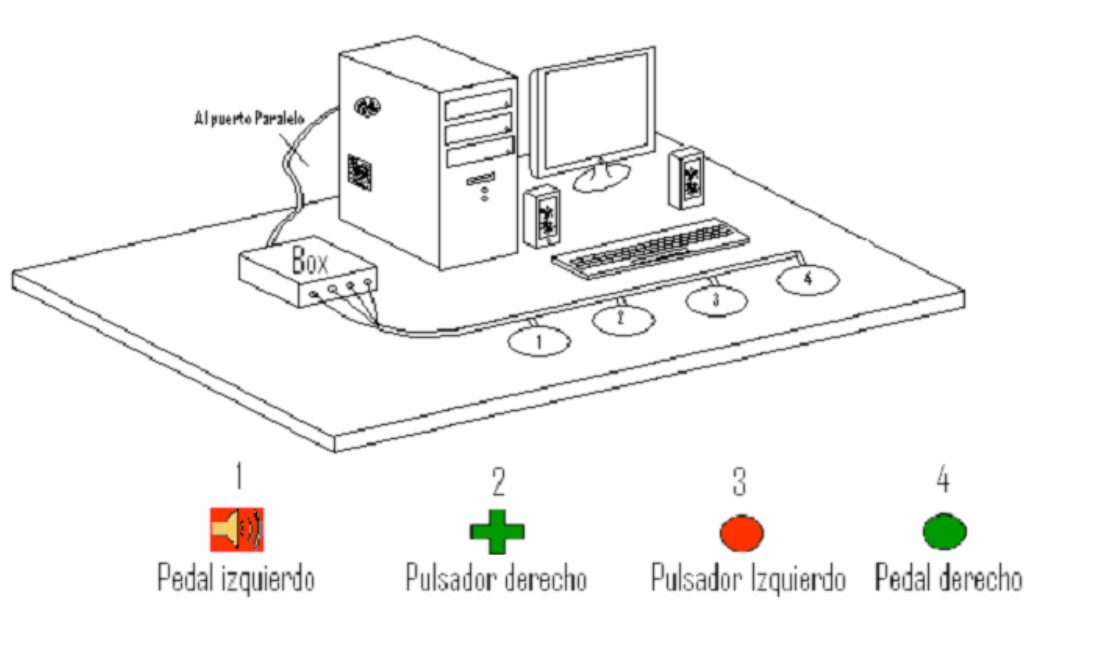
La velocidad de reacción es el tiempo que necesita el cuerpo para reaccionar ante un estímulo, sea visual, sonoro, táctil o de cualquier índole. En la vida diaria sería, por ejemplo, **lo que tardaria en soltar el vaso desde que se da cuenta que se esta quemando**. En la carretera, sería **lo que tardaria en pisar el freno desde que visualiza un peligro que lo cita a frenar**. **Tener una buena velocidad de reacción es muy positivo a la hora de conducir.**

Para que el conductor organice y coordine la información que le viene del exterior necesita prestar la debida atención pero, algunas circunstancias dificultan ese estado de alerta y le impiden reaccionar a tiempo.
Todo conductor debe ser consciente de que debe mantener un nivel de alerta adecuado, tratar de observar la circulación mucho más allá del coche que le precede porque en cualquier momento puede producirse un cambio en el entorno ante el que hay que reaccionar:

- **Un coche que frena**
- **un semáforo que cambia a rojo**
- **un peatón que cruza, etc.**

Cualquier conductor tendría grandes dificultades para atender adecuadamente, y al mismo tiempo, a distintas fuentes de información (semáforos, peatones, conversaciones por teléfono,sus propios pensamientos...) Lo que realmente importa es que sepa cambiar rápidamente de un estímulo a otro y seleccionar en cada momento lo fundamental.

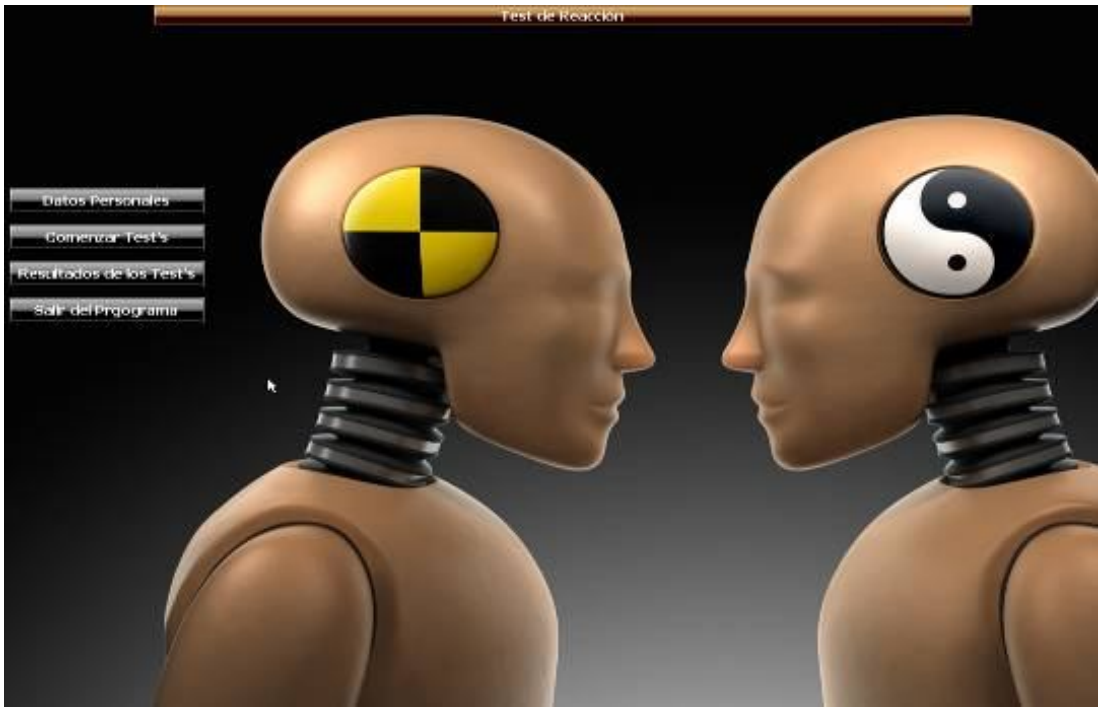
DIAGRAMA EN BLOQUES



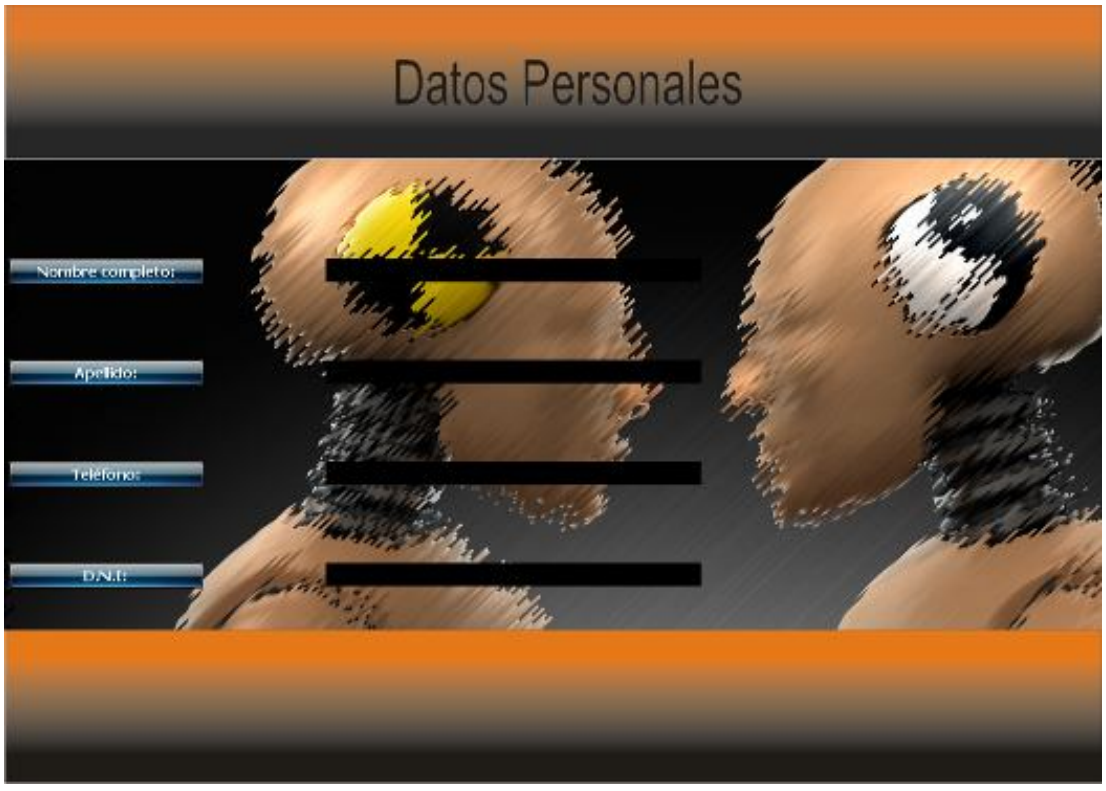
Nota: el circuito se encuentra en la sección “circuitos o diagramas” de este informe.

USANDO EL DISPOSITIVO

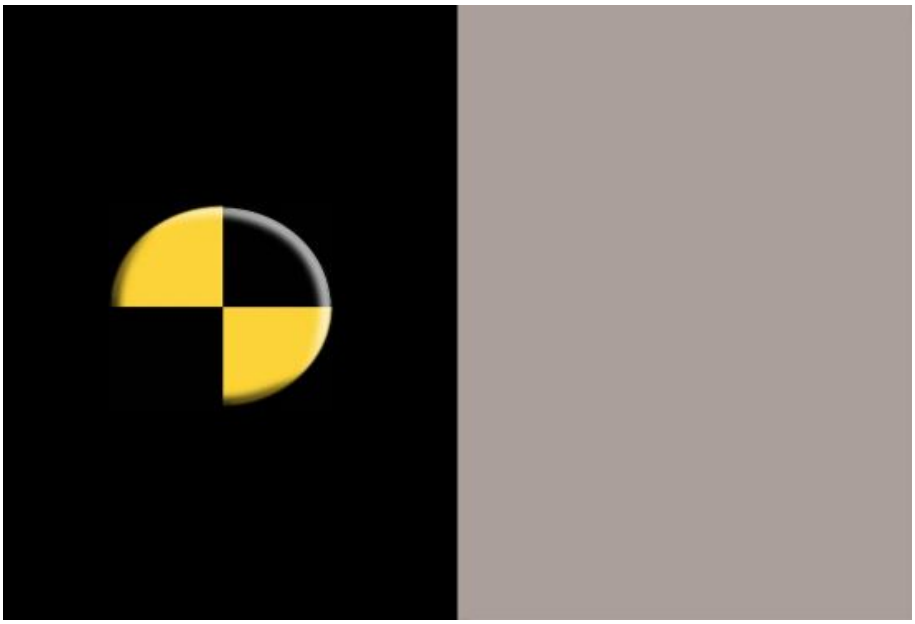
El sistema tiene dos pruebas en donde el usuario interactúa por medio de pulsadores
La pantalla de inicio es la siguiente:



El examinador selecciona “Datos Personales” para cargar los datos en los siguientes campos:

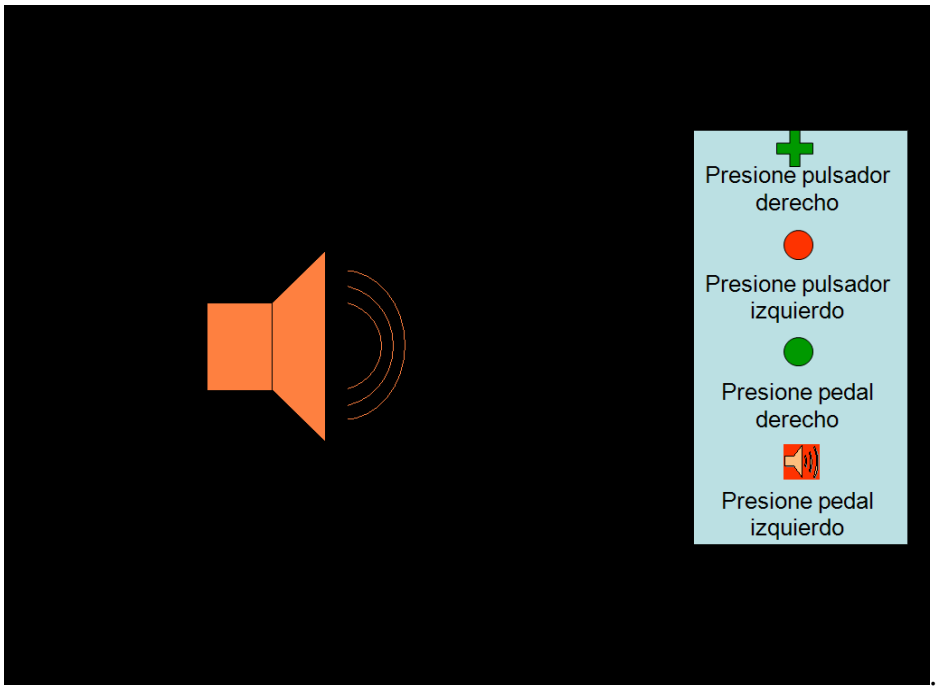


Una vez llenados los campos por medio del teclado de la PC, los mismos son guardados en archivos de texto, los cuales van a ser creados en el disco, uno por persona, en donde también se agregarán los resultados de los test. En este momento se está en condiciones de acceder a las pruebas.
La primera consiste en predicción: el usuario debe presionar el pulsador cuando crea que la imagen haya salido de la pantalla. En ese momento se guarda el tiempo y la imagen correspondiente a ese tiempo para luego ser mostrados en el menú dinámico de resultados:



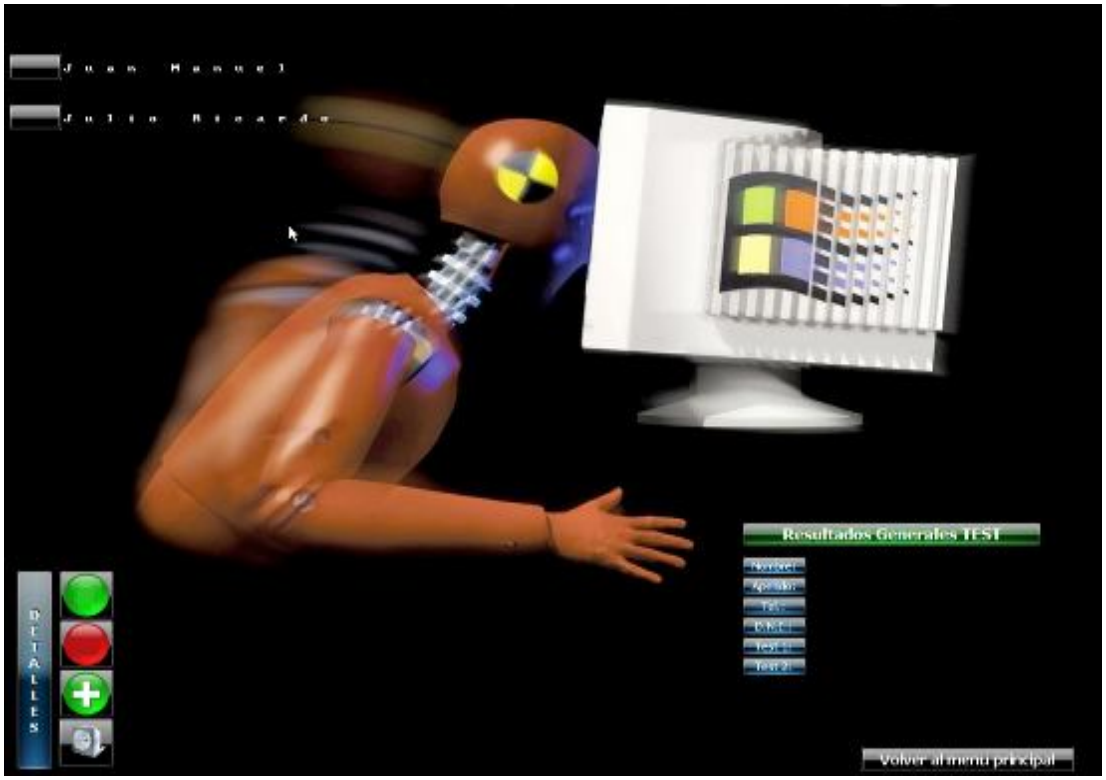
Luego de esta prueba se pasa al tutorial de la segunda prueba, que se muestra a continuación:

Esta segunda prueba consiste en la medición de la reacción de una persona, cada imagen corresponde a uno de los 4 pulsadores, estas irán apareciendo en forma aleatoria por un tiempo máx. de 1 seg, lo que implicaría que el usuario debería reaccionar (apretar el pulsador correspondiente con la imagen) dentro de ese tiempo. La siguiente imagen muestra la pantalla de este Test:

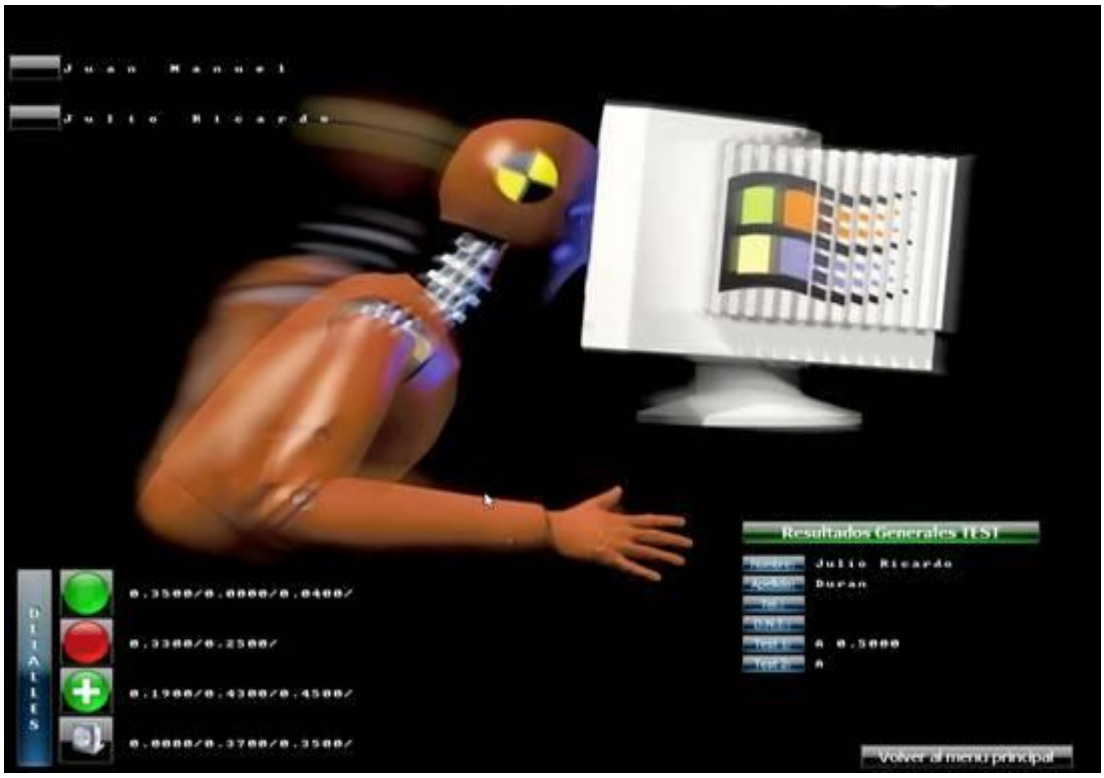


La imagen correspondiente al parlante esta acompañada de un beep.wav que lo levantamos con el software. Cada vez que se termina un test el sistema indica al usuario si ha aprobado aunque no muestra detalles de los resultados.

Una vez finalizadas las dos pruebas volvemos a la pantalla de inicio pudiendo ver los resultados de esa persona o realizar otra prueba con una diferente. Si elegimos ver los resultados aparece la siguiente pantalla:



Para ver los resultados en detalle se deben seleccionar los botones del lado superior izquierdo correspondientes a cada persona:



En este caso vemos los resultados de Julio Durán. Cada tiempo corresponde a la diferencia del tiempo en que se apretó el pulsador y el tiempo en que permanece la imagen en pantalla, cabe aclarar que cuando el tiempo mostrado es cero significa que no reaccionó ante la imagen. Cuanto más grande sea el tiempo mejor ha sido la respuesta de la persona.

En la parte inferior derecha se muestran los datos personales de la persona seleccionada. Éstos son levantados de los archivos guardados. Junto al resultado del primer Test se encuentra el tiempo de cuan cerca estuvo la persona de acertar al momento justo.

PROGRAMACIÓN

El código fuente ha sido desarrollado en devc++ utilizando la librería Allegro. Por medio de la utilización de esta librería, se pudo implementar todo el entorno gráfico que maneja el software.

Allegro:

A continuación se hará una descripción de las funciones de Allegro:

Instalar los periféricos

```
#include<allegro.h>
int main()
{
allegro_init();
install_keyboard();
install_timer();
install_mouse();
...
...
...
allegro_exit();
}
      END_OF_MAIN();
```

Le entrega al *allegro* el control del teclado, lo que nos permitirá utilizar sus funciones para capturar datos.

Instala los temporizadores del *allegro*, funciones para manejar el control del tiempo, necesario para el mouse.

Le da al *allegro* el control del mouse, esto nos permitirá utilizar las rutinas que posee para controlarlo.

Cualquier función de *allegro* que utilice el mouse, el teclado o los temporizadores, debe estar después de los install_* y antes del allegro_exit();

Abrir el modo gráfico

```
#include<allegro.h>
int main()
{
allegro_init();
install_keyboard();
install_timer();
install_mouse();
set_color_depth(8);
set_gfx_mode(GFX_AUTODETECT,800,600,0,0);
...
...
...
allegro_exit();
}
END_OF_MAIN();
```

Especifica la profundidad de color a la que deseamos trabajar (8,16,32 bits), la profundidad por defecto es 8 bits, mientras más alta es más lento y más bonito.

Inicia el modo gráfico

Receta de Cocina

Resolución de la pantalla que deseemos abrir (800x600 pixels) *allegro* permite abrir cualquier resolución que soporte la tarjeta de video.

El driver de la tarjeta de video, "GFX_AUTODETECT" para que *allegro* la detecte.

Bitmaps

Los BITMAP *, son variables que representan un rectángulo, sobre el que puedo dibujar cosas, para allegro la pantalla (screen) es un BITMAP*, y para toda función que dibuje algo hay que especificarle en que BITMAP, deseamos que lo haga.

Sin embargo, el hecho que creemos un BITMAP* y dibujemos sobre el no implica que esto último se muestre en la pantalla, sino que quedará almacenado en la memoria para poder usarlo después.

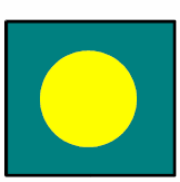
```
BITMAP * bmp;
```
Declara a bmp como una variable de tipo Bitmap.

```
BITMAP * create_bitmap(int x, int y);
```
Reserva memoria para un bitmap de tamaño x,y, para poder dibujar sobre un bitmap hay que utilizar esta función primero.

```
void destroy_bitmap(Btimap * bmp);
```
Libera la memoria que tenía reservada un bitmap.

Ej.
```
BITMAP *primero;
primero=create_bitmap(50,50);
circlefill(primero,25,25,15,12);
destroy_bitmap(primero);
```

Crea un bitmap “primero”, le reserva un tamaño de 50x50 pixels y le dibuja un círculo relleno, luego libera el bitmap, sin embargo, en la pantalla no aparece nada en ningún momento.



primero

screen

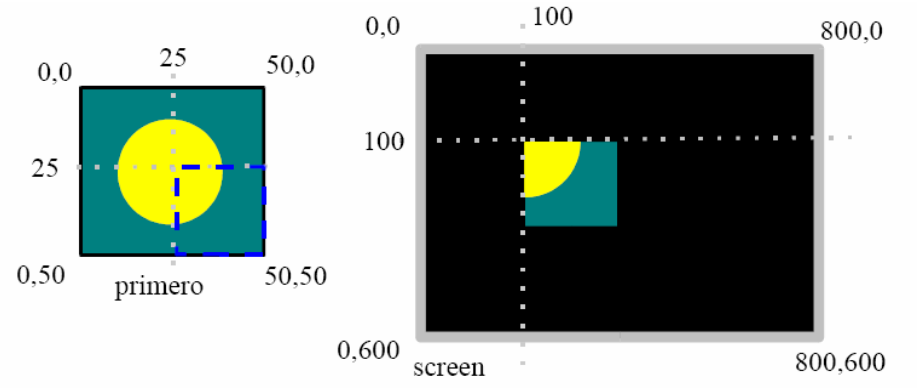
```
Void blit(BITMAP* origen, BITPMAP* destino, int xorigen, int yorigen,
int xdestino, int ydestino, int tamax, int tamy);
```

Dibuja del bitmap “origen” en la posición xorigen, yorigen, al bitmap destino en la posición xdestino, ydestino, un recuadro de tamaño tamax*tamay.

Retomando el ejemplo anterior
```
blit(primero,screen,25,25,100,100,25,25);
```

```
void clear(BITMAP* bmp);
```
Limpia el bitmap “bmp” al color 0, negro a menos que se cambie.

```
void clear_to_color(BITMAP* bmp,int c);
```
Limpia el bitmap bmp al color c.



Imprimir texto

```
void textprintf(Bitmap* bmp, font f, int x, int y, int c, char* formato, var1, var2, ...
, varn);
```

Detiene la ejecución del programa en espera de que presionen UNA sola tecla, devuelve el código ASCII, de la tecla presionada, NO imprime en pantalla la tecla presionada.

Primitivas de dibujo

```
int rectfill (Bitmap *bmp,int x1,int y1, int x2,int y2, int c);
```

Dibuja un rectángulo, desde la posición *(x1,y1)* hasta la posición *(x2,y2)* de *bmp* relleno con el color c, y sin borde.

Teclado y mouse

Ahora veremos las funciones que presenta *allegro* para acceder al teclado, recordemos que para utilizarlas debemos llamar primero a install_keyboard()

```
int readkey(void);
int clear_keybuf (void); Limpia el buffer del teclado
```

Para el mouse debemos ejecutar install mouse(), y podremos usar los siguientes comandos:

```
show_mouse (screen);      Muestra el mouse
unscare_mouse ();         Enmascara el mouse
```

Temporizadores

Son funciones que nos permiten llevar dentro del programa una cuenta del tiempo que ha pasado, esto nos permite colocarle al juego una velocidad fija en cualquier máquina

```
void install_int(proc funcion, int tiempo);
```

Llama a la función “función”, cada que pasen “tiempo” milisegundos, sin importar en donde se encuentre la ejecución del programa.

Toda variable que se utilice dentro de lafunción deberá ser preferiblemente global y volátil, así:

```
volatile int t;
```

La función que le enviemos al temporizador debe ser lo más sencilla posible y solo debe realizar cálculos de enteros, por ejemplo:

```
void tiempo()
{
t++;
}
END_OF_FUNCTION(tiempo);
```

Y en el código se deberá incluir antes de llamar a install_int:

```
LOCK_VARIABLE(t);
LOCK_FUNCTION(tiempo);
```

¿Cómo utilizarlos? Haciendo que el ciclo principal del juego no se llame sino cada medio segundo por ejemplo, garantizamos que nuestro juego corre a 0,5 segundos por ciclo.

```
volatile int t;

void tiempo()
{
t++;
}
END_OF_FUNCTION(tiempo);

BITMAP* buf;

void main()
{
allegro_init();
install_keyboard();
install_mouse();
install_timer();
set_gfx_mode(GFX_AUTODETECT,800,600,0,0);
buf=create_bitmap(800,600);
clear(buf);
//instalamos el temporizador
LOCK_VARIABLE(t);
LOCK_FUNCTION(tiempo);
t=0;
install_int(tiempo,500);                    //que cambie cada 0,5 seg
int auxt;                                   //tiempo auxiliar
auxt=t;
int x=0;
while(!key[KEY_ESC])                        //el ciclo principal
{
if(aux!=t) //Si ya avanzó el tiempo 0,5 seg.
{
clear(buf);                                 //limpiamos el buffer
rectfill(buf,x,10,x+100,110,12);            //dibujamos un rectangulo en el buff
blit(buf,screen,0,0,0,0,800,600);           //pasamos el buff a la pantalla
auxt=t;                                     //volvemos a guardar el tiempo actual
x=x+10;                                     //para que se desplace el cuadro
}
}
allegro_exit();
```

Creación de ARCHIVOS:

Por cada persona que entra al sistema se crea un archivo, en el cual se encontraran los siguientes datos:

** Todos los datos que se cargan en el menú "DATOS PERSONALES".
** Resultados (aprobó o no) y el tiempo de cada test realizado.

Para generar este archivo final de c/persona se pasa por distintas etapas:

1) Al llenar la ficha personal se crea el archivo de la persona (que es el final), pero aquí inicialmente solo se tienen los datos de la misma.
2) Cuando la persona termina el primer test el resultado del mismo (aprobó o no) así como el tiempo de reacción se guardan en el archivo que se creó anteriormente.
3) En el momento que se comienza la segunda prueba se crean 4 nuevos archivos, que corresponden a las cuatro figuras del test (círculo verde, círculo rojo, cruz y parlante), en cada uno de estos archivos se guardan los sucesivos tiempos de reacción de la persona.
4) Una vez que la persona completo el segundo test, el sistema pasa todos la información que se guardo en el paso anterior al archivo inicialmente creado, quedando en este todos los resultados y datos personales del usuario.

Código fuente encargado de crear los archivos:

```
stringstream ss;
string nom;
unsigned long int num_arch = per;

ss << "Persona_" << num_arch << ".txt";
ss >> nom;
fichero=fopen(nom.c_str(), "w");
```

Por medio de esta sección del programa, se pueden crear dinámicamente los archivos para cada persona que realiza las actividades.

Con la variable **"per"**, se le asigna a cada persona un archivo en donde se almacena la información personal así como los resultados de los test.

Gracias a esto, el sistema ira creando archivos con los siguientes nombres:

Persona_1.txt → para la primera persona que realiza el test;
Persona_2.txt → para la segunda persona que realiza el test;
; y así sucesivamente con cada persona que realiza el test.

Creación de números aleatorios:

Por medio de esta sección del programa se pueden crear números de forma aleatoria entre 1 y 4, los cuales fueron asignados a las 4 imágenes del test numero 2.

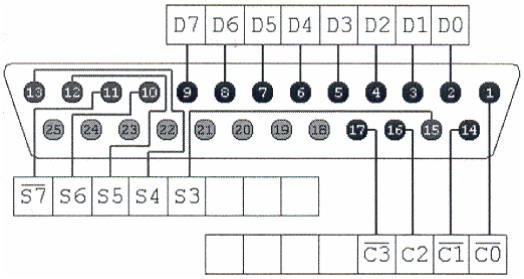
Con esto se logra que la secuencia de imágenes sea aleatoria por lo que sería imposible que el usuario anticipe la siguiente imagen.

Código fuente encargado de crear los archivos:

```
void GenerateSeed()
{
	srand((unsigned)time( NULL ));}

	int Random(int liminf,int limsup)
```

```
{
	int Numero = (rand () % (limsup-liminf+0x1)) + liminf;
	return Numero;
}

int Random(int liminf,int limsup);
GenerateSeed();
secret=Random(0x1,0x4);
```

CONFIGURACIÓN DEL PUERTO PARALELO



Datos	D0 ~ D7	Salida	2 ~ 9	Terminales de Datos de D0 a D7
ESTADO	BUSY	Entrada	11	Con un valor alto indica que la impresora está ocupada y no puede recibir datos nuevos. Puede ponerse a nivel alto en caso de error
	ACK	Entrada	10	Con un valor bajo indica la impresora que ha recibido un dato y está disponible para recibir otro.
	PE	Entrada	12	Un valor alto se maneja un error de falta de papel.
	SLCT IN	Entrada	13	Un valor alto se indica que la impresora está en línea.
	ERROR	Entrada	15	Con un valor bajo se indica que se ha generado un error en la impresor, falta de papel o algún otro.
CONTROL	SELECT	Salida	17	Activa a nivel bajo. Indica a la impresora que se ha seleccionado ésta.
	INIT	Salida	16	Con un nivel bajo se envía un reset a la impresora
	AUTFD	Salida	14	A nivel bajo, la impresora se encarga de hacer un salto de línea al recibir el carácter de retorno de carro
	STROBE	Salida	1	Validación de datos. Cuando la impresora detecta un nivel bajo, acepta un dato.
			18 ~ 25	Retornos de los bits de los datos 0 hasta 7.

El puerto paralelo utiliza señales de tipo TTL. Éste codifica las señales de la siguiente manera:

- "1" lógico = 5 Voltios.
- "0" lógico = 0 Voltios.

Aunque no es necesario tener exactamente ese voltaje en la entrada, ya que el puerto reconoce un "1" lógico aún cuando el voltaje de entrada alcanza un mínimo de 3.6 Voltios, ya que por debajo de ese nivel, el puerto toma la señal como un "0" lógico.

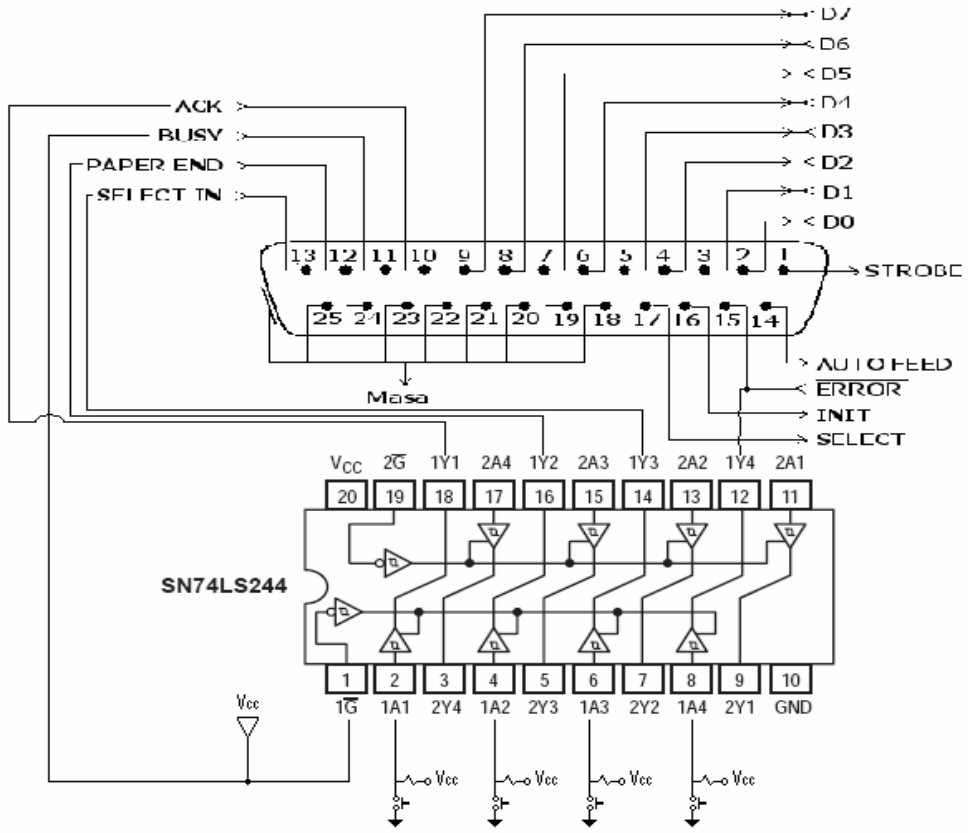
Como se puede observar en la tabla anterior los bits 10, 11, 12, 13 y 15 son entradas, estos son los que se han utilizado para los pulsadores. Cabe aclarar que estos bits en cuestión forman parte de un registro interno del puerto, denominado registro de ESTADO (DIR. 379h), al cual se debe acceder para poder así leer el estado de los mismos.

BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
BSY	ACK	PE	SLCT IN	ERR	X	X	X

Código fuente encargado de leer los pulsadores:

```
#include <stdio.h>
#include <conio.h>
#include <windows.h>

// prototype (function typedef) for DLL function Inp32:

     typedef short _stdcall (*inpfuncPtr)(short portaddr);
     typedef void _stdcall (*oupfuncPtr)(short portaddr, short datum);

short PUERTO(void)
{
     HINSTANCE hLib;
     inpfuncPtr inp32;
     oupfuncPtr oup32;

     short x;
     int i,j,h,r,p;

     //cargamos la libreria para poder usar los puertos

     hLib = LoadLibrary("inpout32.dll");
     if (hLib == NULL)
          {printf("LoadLibrary Failed.\n");
          return -1;}

     // comprobamos en funcionamiento de las funciones

     inp32 = (inpfuncPtr) GetProcAddress(hLib, "Inp32");
     if (inp32 == NULL)
          {printf("GetProcAddress for Inp32 Failed.\n");
          return -1;}

     oup32 = (oupfuncPtr) GetProcAddress(hLib, "Out32");
     if (oup32 == NULL)
          {printf("GetProcAddress for Oup32 Failed.\n");
          return -1;}

     // escribiendo el puerto paralelo

     j=0;
     i=0x379;
     x=0;

     x = (inp32)(i);

    return(x);
   }
```

CIRCUITOS O DIAGRAMAS



Aclaración:

- BSY: busy, en el registro este bit esta negado. El circuito de conexionado de los pulsadores es el siguiente:

P1 P2 P3 P4

ERROR en realidad no se encuentra negado como figura en la imagen anterior (tener en cuenta).

Como se puede observar el circuito de c/pulsador genera un “1” lógico en forma constante, y solo cuando se presiona el mismo, el puerto “lee” un “0” lógico. Esto genera como resultado los siguientes valores del registro (en hexadecimal):

- 38 = 00111xxx. Figura: Circulo Verde(P1)
- 58 = 01011xxx. Figura: Circulo Rojo (P2)
- 68 = 01101xxx. Figura: Cruz (P3)
- 70 = 01110xxx. Figura: Parlante (P4)

RESULTADOS DE LAS PRUEBAS

Las pruebas fueron exitosas en su totalidad.

CONCLUSIONES

En el transcurso del proyecto, el grupo fue sorteando diferentes tipos de dificultades, algunas de ellas las describiremos en esta sección.

Uno de los más importantes inconvenientes fue crear archivos con distintos nombres, en forma dinámica, es decir que el sistema para c/usuario cree un archivo, en donde se guardan todos los resultados junto con sus datos, pero sin repetir nombres de otros anteriores, de esta manera se corresponden unívocamente un solo archivo con cada usuario que entró.

Otro inconveniente que se presentó, fue al momento de realizar la comunicación por puerto paralelo, ya que para el manejo del puerto, como se observó en el desarrollo de este informe, se necesita la asistencia de la librería “windows.h”, la cual no corre en conjunto con ALLEGRO. Esto genero un problema ya que todo el software esta hecho en base a esta librería (ALLEGRO). Este problema se soluciono creando dos *.cpp que formen parte del mismo proyecto, uno el programa principal, y el otro la función que se encarga de leer el puerto (que funciona con “windows.h”). Compilando el proyecto, se logró hacer andar ambas librerías en forma conjunta (“windows.h” y “ALLEGRO”).

Gracias a los distintos tutoriales que se han descargado de internet, pudimos ir sorteando todos los desafíos que se fueron presentando.

ANEXOS:

LISTADOS DE PROGRAMAS

Dev C++ con inclusión de la galeria Allegro.

```
#include <allegro.h>
#include <winalleg.h>
#include <stdio.h>
#include <conio.h>
#include <sstream>
#include <string>
#include <fstream>
#include <time.h>

// prototype (function typedef) for DLL function Inp32:

     typedef short _stdcall (*inpfuncPtr)(short portaddr);
     typedef void _stdcall (*oupfuncPtr)(short portaddr, short datum);

void GenerateSeed();                    // numero al azar
int Random(int liminf,int limsup);      // esto tambien

using namespace std;


volatile int t;
void tiempo()
{
t++;
}
END_OF_FUNCTION(tiempo);



PALETTE paleta;
BITMAP* fondo;
BITMAP* titulo;
BITMAP* comenzar_test;              //Desde aca
BITMAP* comenzar_test1;
BITMAP* datos_personales;
BITMAP* datos_personales1;          //Se crean los bitmaps de los botones
BITMAP* resultado_test;
BITMAP* resultado_test1;
BITMAP* salir_programa;
BITMAP* salir_programa1;            //Hasta aca
int g=0,h=0;//variables para que no dibuje continuamente, g para los pintados y h para
los comunes.
int p=180;
int a=0,b=0,c=0,d=0;
int f=0;//me controla retorno al menu
int cond=0,cond1=0;//CAMBIA DESPUES EL COND1 PARA QUE ANDE BIEN, QUE SOLO ENTRE DESPUES
DE COMLETAR DATOS cambia a 0
int per=0;   //me genera los numeros de los archivos de las  personas
int loop=0;  //para que no siga jutando planillas sin completar las pruebas

int cir_mov();
int cadena();
int mostrar();
int PUERTO();

int main()
{
do
```

```
{
g=0;
h=0;
f=0;
allegro_init();
install_keyboard();
install_mouse();                      //Llamo al mouse
set_color_depth(32);
set_gfx_mode(GFX_AUTODETECT,1024,768,0,0);


fondo=load_bitmap("dummies.bmp",paleta);                                  //Fondo de la
pantalla princ
set_palette(paleta);
blit(fondo,screen,0,0,0,0,1024,768);
titulo=load_bitmap("titulo.bmp",paleta);
blit(titulo,screen,0,0,140,0,1024,768);

datos_personales=load_bitmap("datos_personales.bmp",paleta);
datos_personales1=load_bitmap("datos_personales1.bmp",paleta);
comenzar_test=load_bitmap("comenzar_test.bmp",paleta);
comenzar_test1=load_bitmap("comenzar_test1.bmp",paleta);              //Creo Bitmap's
resultado_test=load_bitmap("resultado_test.bmp",paleta);
resultado_test1=load_bitmap("resultado_test1.bmp",paleta);
salir_programa=load_bitmap("salir_programa.bmp",paleta);
salir_programa1=load_bitmap("salir_programa1.bmp",paleta);


show_mouse(screen);                                                       //Muestra mouse en
pantalla

while(g==0)                            //el ciclo principal y sus animaciones
{


if ((((mouse_y<190)||(mouse_y>190))&&((mouse_y>350)||(mouse_y<350)))||(mouse_x>p))
  { scare_mouse();
    blit(datos_personales,screen,0,0,5,200,1024,768);
    blit(comenzar_test,screen,0,0,5,240,1024,768);
    blit(resultado_test,screen,0,0,5,280,1024,768);                     //Coloco menus
originales
    blit(salir_programa,screen,0,0,5,320,1024,768);
    unscare_mouse();
    h=0;

    while( ( ((mouse_y>190)||(mouse_y<350)) &&(mouse_x>p)||((mouse_y<190)||(mouse_y>350))
&& ((mouse_x>p)||(mouse_x<p))  || ( ((mouse_y>220)&&(mouse_y<245)) ||
((mouse_y>260)&&(mouse_y<285)) || ((mouse_y>300)&&(mouse_y<325)) )&&(mouse_x<p) )
&&(h!=1))
     {
//para los intermedios
      } //Este me mantiene los menus originales
  }

if ((mouse_y>190)&&(mouse_y<230)&&(mouse_x<p))
  { scare_mouse();
    blit(comenzar_test,screen,0,0,5,240,1024,768);
    blit(resultado_test,screen,0,0,5,280,1024,768);     //Reorganizo los otros
    blit(salir_programa,screen,0,0,5,320,1024,768);
    blit(datos_personales1,screen,0,0,5,200,1024,768);  //Coloco el reslatado
    unscare_mouse();

    while(((mouse_y>190)&&(mouse_y<230)&&(mouse_x<p))&&(g==0))
     {
       if (mouse_b &1 && cond1==0)
       {g=1;
        a=1;
        }
```

```
       }//este me sirve para no pintar los pintados, continuamente, lo mismo para los
demas pos
   }

if ((mouse_y>240)&&(mouse_y<270)&&(mouse_x<p))
  {
    scare_mouse();
    blit(datos_personales,screen,0,0,5,200,1024,768);
    blit(resultado_test,screen,0,0,5,280,1024,768);
    blit(salir_programa,screen,0,0,5,320,1024,768);
    blit(comenzar_test1,screen,0,0,5,240,1024,768);
    unscare_mouse();

    while(((mouse_y>240)&&(mouse_y<270)&&(mouse_x<p))&&(g==0))
     {
       if (mouse_b &1 &&(cond==1))
       {g=1;
        b=1;}
        }
    }
                                                                          //Animacion
del menu princ
if ((mouse_y>280)&&(mouse_y<310)&&(mouse_x<p))
  {
    scare_mouse();
    blit(datos_personales,screen,0,0,5,200,1024,768);
    blit(comenzar_test,screen,0,0,5,240,1024,768);
    blit(salir_programa,screen,0,0,5,320,1024,768);
    blit(resultado_test1,screen,0,0,5,280,1024,768);
    unscare_mouse();
    while(((mouse_y>280)&&(mouse_y<310)&&(mouse_x<p))&&(g==0))
     {
       if (mouse_b &1 ) //mirar esta condicion creo que anda bien
       {g=1;
        c=1;}

       }
    }

if ((mouse_y>320)&&(mouse_y<350)&&(mouse_x<p))
  {
    scare_mouse();
    blit(datos_personales,screen,0,0,5,200,1024,768);
    blit(comenzar_test,screen,0,0,5,240,1024,768);
    blit(resultado_test,screen,0,0,5,280,1024,768);
    blit(salir_programa1,screen,0,0,5,320,1024,768);
    unscare_mouse();
    while(((mouse_y>320)&&(mouse_y<350)&&(mouse_x<p))&&(g==0))
     { if (mouse_b &1)
       {g=1;
        d=1;}
       }
    }

h=1;
}                       //aca termina ciclo principal



destroy_bitmap(fondo);
destroy_bitmap(datos_personales);
destroy_bitmap(datos_personales1);
destroy_bitmap(comenzar_test);
destroy_bitmap(comenzar_test1);
destroy_bitmap(resultado_test);
destroy_bitmap(resultado_test1);
destroy_bitmap(salir_programa);
destroy_bitmap(salir_programa1);
```

```
destroy_bitmap(titulo);



allegro_exit();

t=0;

if (a==1&&loop==0)
{per++;
 f=cadena();
 a=0;cond=1;cond1=1;loop=1;}
else
{a=0;f=1;}

if (b==1)
{f=cir_mov();
 b=0;cond=0;loop=0;cond1=0;}

if (c==1)
{f=mostrar();
 c=0;cond1=0;}

if (d==1)
{f=0;
 d=0;}

}while (f==1);
}
END_OF_MAIN();

//*******************************--------************************************



int cadena()
{
  FILE *fichero;
  int camp=1;
  int x=300;
  int y=255;
  int color=makecol(255,255,255);
  int max=25;
  int j=0,p=0;
  char cadena1[25];

  char lt=0;
  BITMAP* primero;
  PALETTE paleta;
  int i=0,auxt=0;
  float w=0;
  int r=0;

  BITMAP* datos_fondo;
  BITMAP* barra_inf;
  BITMAP* barra_sup;
  BITMAP* camp_1;
  BITMAP* camp_2;
  BITMAP* camp_3;            //Se crean los bitmaps de los botones
  BITMAP* camp_4;
  BITMAP* camp_blanco;
  BITMAP* salir;



  allegro_init();
  install_keyboard();
  install_mouse();
```

```
  set_color_depth(32);
  set_gfx_mode(GFX_AUTODETECT,1024,768,0,0);
  LOCK_VARIABLE(t);
  LOCK_FUNCTION(tiempo);
  t=0;

  datos_fondo=load_bitmap("dummies_datos.bmp",paleta);
//Fondo de la pantalla princ
  set_palette(paleta);
  blit(datos_fondo,screen,0,0,0,0,1024,768);
  barra_sup=load_bitmap("barra_sup.bmp",paleta);
  blit(barra_sup,screen,0,0,0,0,1024,768);
  barra_inf=load_bitmap("barra_inf.bmp",paleta);
  blit(barra_inf,screen,0,0,0,616,1024,768);

  camp_1=load_bitmap("camp_1.bmp",paleta);
  camp_2=load_bitmap("camp_2.bmp",paleta);
  camp_3=load_bitmap("camp_3.bmp",paleta);
  camp_4=load_bitmap("camp_4.bmp",paleta);
  camp_blanco=create_bitmap(350,23);

  blit(camp_1,screen,0,0,5,250,1024,768);
  blit(camp_2,screen,0,0,5,350,1024,768);
  blit(camp_3,screen,0,0,5,450,1024,768);                        //Coloco menus originales
  blit(camp_4,screen,0,0,5,550,1024,768);

  rectfill(camp_blanco,0,0,350,23,0);
  blit(camp_blanco,screen,0,0,300,250,1024,768);
  blit(camp_blanco,screen,0,0,300,350,1024,768);
  blit(camp_blanco,screen,0,0,300,450,1024,768);                     //Coloco campos
donde escribo
  blit(camp_blanco,screen,0,0,300,550,1024,768);


  install_int(tiempo,500); //que cambie cada 0,5 seg


  primero=create_bitmap(6,10);

  //show_mouse(screen);

 stringstream ss;
 string nom;
 unsigned long int num_arch = per;
 ss << "Persona_" << num_arch << ".txt";
 ss >> nom;
 fichero=fopen(nom.c_str(), "w");

  while (camp<=4)
  {

  do
   {

    if ((mouse_y>(y-3))&&(mouse_y<(y+3))&&(mouse_x>(x+12*i-9))&&(mouse_x<(x+12*i+9)))
    {scare_mouse();
     r=1;}

    auxt=t;
    rectfill(primero,0,0,6,10,makecol(255,255,255));
    blit(primero,screen,0,0,x+12*i,y,6,14);
    while (w<15000000)
    {w++;
          }
    w=0;
    rectfill(primero,0,0,6,10,0);
    blit(primero,screen,0,0,x+12*i,y,6,14);
    while (w<15000000)
```

```
    {w++;
          }
    if (r==1)
    {unscare_mouse();
     r==0;}
    w=0;
    lt=0;
   while (keypressed() && (t-auxt)*0.01<0.05)
     {
     lt=readkey();
     if( (lt>=0) && (lt<=255) )
         {if(lt!=13)
           {if(lt==8)
             {
              scare_mouse();
              if(i==0)
              {clear_keybuf();
              unscare_mouse();
                      }
              else
              {
              i--;
              textprintf(screen,font,x+12*i,y,color," ");
              cadena1[i+1]='#';
              clear_keybuf();
              unscare_mouse();}
              }
            else
             {
              scare_mouse();
              textprintf(screen,font,x+12*i,y,color,"%c",lt);
              cadena1[i]=lt;
              cadena1[i+1]='#';
              i++;
              clear_keybuf();
              unscare_mouse();
              }
           }
         else
          {cadena1[i]='#';
          clear_keybuf();}
         }
     }

   }while( lt!=13 && i<max );
y=y+100;
j=0;


    stringstream ss;
    string nom;
    unsigned long int num_arch = per;

    ss << "Persona_" << num_arch << ".txt";
    ss >> nom;



 fichero=fopen(nom.c_str(), "a");
 while (cadena1[j]!= '#' && j<25)
   { p=cadena1[j];
     fputc(p,fichero);
     j++;

   }

   if (camp!=4)
   {p=cadena1[j];
```

```
    fputc(p,fichero);}                  //completo los campos en el archivo
   else
   {cadena1[j]='/';
   p=cadena1[j];
   fputc(p,fichero);}
 fclose(fichero);

camp++;
i=0;
}


destroy_bitmap(primero);

destroy_bitmap(camp_1);
destroy_bitmap(camp_2);
destroy_bitmap(camp_3);
destroy_bitmap(camp_4);

allegro_exit();
return 1;
}

//*******************************--------************************************

int cir_mov()
{

     HINSTANCE hLib;
     inpfuncPtr inp32;
     oupfuncPtr oup32;

     short xx;
     int ii,jj,hh,rr;

     //cargamos la libreria para poder usar los puertos

     hLib = LoadLibrary("inpout32.dll");


FILE *fichero;
FILE *fichero2;
BITMAP* buf;
PALETTE paleta;
BITMAP* circulo;
BITMAP* mitad;
BITMAP* aurora;
BITMAP* tuto1;
BITMAP* tuto2;
BITMAP* fondo_test2;
BITMAP* c_verde;
BITMAP* c_rojo;
BITMAP* cruz;
BITMAP* parl_nar;
BITMAP* negro;
BITMAP* dummy_error;

float c;
int secret;
DATAFILE *MisDatos; //con esto comienzo a cargar la nueva fuente
FONT *MiFuente;     //con esto comienzo a cargar la nueva fuente
float ref;
float Treaccion,pers;
char s;
char cadena1[150];
int  k=0;
int  v;
short  entero;    //aca recibo el dato
```

```
int  contador_bucle=0;
int  cond=0;
float fVal = 65.3;
char cVal[5];
int cont_trafo=0;
int cont_erro;

allegro_init();
install_keyboard();
install_mouse();
install_timer();
set_color_depth(32);
set_gfx_mode(GFX_AUTODETECT,1024,768,0,0);
buf=create_bitmap(1024,768);
clear(buf);
//instalamos el temporizador

LOCK_VARIABLE(t); //instala timer
LOCK_FUNCTION(tiempo);
t=0;
install_int(tiempo,10); //que cambie cada 0.01 seg en ms
int auxt,auxt1=0; //tiempo auxiliar
auxt=t;
int x=0;
int z=0;
dummy_error=load_bitmap("dummy_error.bmp",paleta);
negro=load_bitmap("negro.bmp",paleta);
c_verde=load_bitmap("c_verde.bmp",paleta);
c_rojo=load_bitmap("c_rojo.bmp",paleta);
cruz=load_bitmap("cruz.bmp",paleta);
parl_nar=load_bitmap("parl_nar.bmp",paleta);
fondo_test2=load_bitmap("fondo_test2.bmp",paleta);
tuto2=load_bitmap("tuto2.bmp",paleta);
tuto1=load_bitmap("tuto1.bmp",paleta);
circulo=load_bitmap("circulo.bmp",paleta);
mitad=load_bitmap("mitad.bmp",paleta);
set_palette(paleta);

fichero=fopen("test2_1.txt", "w");
fclose(fichero);
fichero=fopen("test2_2.txt", "w");
fclose(fichero);
fichero=fopen("test2_3.txt", "w");
fclose(fichero);
fichero=fopen("test2_4.txt", "w");
fclose(fichero);

blit(tuto1,buf,0,0,0,0,1024,768);     //imprime el tutorial
blit(buf,screen,0,0,0,0,1024,768);
while(!key[KEY_SPACE])
{
                      }


while(x!=1302) //el ciclo principal
  {
     if(auxt!=t) //Si ya avanzó el tiempo 0,5 seg.
     {
     //limpiamos el buffer
     clear(buf);
     //dibujamos un rectangulo en el buff
     blit(circulo,buf,0,0,x-266,250,300,285);
     blit(mitad,buf,0,0,512,0,1024,768);
     //pasamos el buff a la pantalla
     blit(buf,screen,0,0,0,0,1024,768);
     //volvemos a guardar el tiempo actual
     auxt=t;
     entero=PUERTO();
```

```
     if ((entero==0x38)&&(cond==0))
     {
     auxt1=t;
     cond=1;}
     //para que se desplace el cuadro
     x=x+6;
     clear(buf);

     }
}
entero=0;
cond=0;
c=auxt1;
ref=auxt*0.01;
pers=c*0.01;
Treaccion=ref-pers;
clear(buf);
aurora=load_bitmap("aurora.bmp",paleta);
blit(aurora,buf,0,0,0,0,1024,768);
blit(buf,screen,0,0,0,0,1024,768);
MisDatos=load_datafile("tempus.dat");    //aca se carga
MiFuente=(FONT *)MisDatos[0].dat;        //aca se carga
text_mode(-1);                           //me hace transparente el fonde del texto

    stringstream ss;
    string nom;
    unsigned long int num_arch = per;             //_-_-_busco nombre IMPORTANTE

    ss << "Persona_" << num_arch << ".txt";
    ss >> nom;

    fichero=fopen(nom.c_str(), "a+");

   if (pers!=0)

               {textprintf(screen,MiFuente,0,150,320000,"Reacciono %f seg.
antes",Treaccion);

               if (Treaccion>1)
               {textprintf(screen,MiFuente,0,300,320000,"Ud no paso la prueba");
               //fichero=fopen("resul_1.txt", "a");
               s='D';
               v=s;
               fputc(v,fichero);
               fclose(fichero);
               }
               else
               {textprintf(screen,MiFuente,0,300,320000,"Ud paso la prueba");
               //fichero=fopen("resul_1.txt", "a");
               s='A';
               v=s;
               fputc(v,fichero);

               s=' ';                        //Guardo el tiempo de reaccion para mostralo
               v=s;
               fputc(v,fichero);

               sprintf(cVal,"%f",Treaccion); // string result '65' is stored in char
array cVal
               cont_trafo=0;
               do{
               v=cVal[cont_trafo];
               fputc(v,fichero);
               cont_trafo++;}
               while (cont_trafo<=5);


               fclose(fichero);}
```

```
               }
   else

   {textprintf(screen,MiFuente,0,300,320000,"Tiempo exedido: Ud no paso la prueba");
               //fichero=fopen("resul_1.txt", "a");
               s='D';
               v=s;
               fputc(v,fichero);
               fclose(fichero);
   }

//fichero=fopen("resul_1.txt", "a");
fichero=fopen(nom.c_str(), "a+");
s='/';
v=s;
fputc(v,fichero);
fclose(fichero);

destroy_bitmap(circulo);
destroy_bitmap(mitad);
destroy_bitmap(aurora);
destroy_bitmap(tuto1);
clear_keybuf();

readkey();

blit(tuto2,buf,0,0,0,0,1024,768);     //imprime el tutorial 2
blit(buf,screen,0,0,0,0,1024,768);
readkey();
while(!key[KEY_SPACE])
{
                       }
blit(fondo_test2,buf,0,0,0,0,1024,768);     //imprime fondo del segundo test
blit(buf,screen,0,0,0,0,1024,768);

clear_keybuf();


install_int(tiempo,10);  //Timer para las imagenes
t=0;
auxt=t;
auxt1=0;
cont_erro=0;

// Instalamos driver de sonido, si devuelve un 0 es porque funciona correctamente
    z = install_sound(DIGI_AUTODETECT, MIDI_AUTODETECT, NULL);
    // cargamos el "sample"  - fichero de sonido digital -
 SAMPLE *sample = load_wav("Beep.wav");



do
{
auxt1=0;
GenerateSeed();
secret=Random(0x1,0x4);
contador_bucle++;

if (secret==1)
{t=0;//Reinicia el timer
 //auxt1=0;//Tambien reinicia la variable contadora
 blit(c_verde,buf,0,0,90,192,512,384);
 blit(buf,screen,0,0,0,0,512,768);

 while(t<100)
 {   entero=PUERTO();
     if ((entero==0x38)&&(cond==0))
        {auxt1=t;
```

```
        cond=1;}
  }
  }

if (secret==2)
{t=0;//Reinicia el timer
 //auxt1=0;
 blit(c_rojo,buf,0,0,90,192,512,384);
 blit(buf,screen,0,0,0,0,512,768);
 while(t<100)
 {  entero=PUERTO();
     if ((entero==0x58)&&(cond==0))
     {
     auxt1=t;
     cond=1;}
  }
  }

if (secret==3)
{t=0;//Reinicia el timer
 //auxt1=0;
 blit(cruz,buf,0,0,90,192,512,384);
 blit(buf,screen,0,0,0,0,512,768);
  while(t<100)
 {  entero=PUERTO();
     if ((entero==0x68)&&(cond==0))
     {
     auxt1=t;
     cond=1;}
  }
  }

if (secret==4)
{t=0;//Reinicia el timer
 //auxt1=0;
 blit(parl_nar,buf,0,0,90,192,512,384);
 blit(buf,screen,0,0,0,0,512,768);
 // Hacemos sonar el sample// 100 = volumen   128 = sonido balanceado   0 = sin bucle
    play_sample(sample,100,128,1000,0);
  while(t<100)
 { entero=PUERTO();
     if ((entero==0x70)&&(cond==0))
     {
     auxt1=t;
     cond=1;}
  }
  }

auxt=100;


c=auxt1; //Lo q tarda el tipo
ref=auxt*0.01; //Es el segundo
pers=c*0.01;

if (pers!=0)
{Treaccion=ref-pers;
}
else
{Treaccion=0.0000;
cont_erro++;}

    stringstream ss;
    string nom;
    unsigned long int num_arch = secret;

    ss << "test2_" << num_arch << ".txt";
    ss >> nom;
```

```
    fichero=fopen(nom.c_str(), "a+");



                sprintf(cVal,"%f",Treaccion); // string result '65' is stored in char
array cVal
                cont_trafo=0;
                do{
                v=cVal[cont_trafo];
                fputc(v,fichero);
                cont_trafo++;}
                while (cont_trafo<=5);
                s='/';
                v=s;
                fputc(v,fichero);
                fclose(fichero);

blit(negro,buf,0,0,90,192,512,384);
 blit(buf,screen,0,0,0,0,512,768);
while(auxt==t)
 {
  }
auxt=t;
clear_keybuf();
cond=0;
entero=0;
}
while(contador_bucle<=10);



fichero=fopen("test2_1.txt", "a+");
s='#';
v=s;
fputc(v,fichero);
fclose(fichero);

fichero=fopen("test2_2.txt", "a+");
s='#';
v=s;
fputc(v,fichero);
fclose(fichero);

fichero=fopen("test2_3.txt", "a+");
s='#';
v=s;
fputc(v,fichero);
fclose(fichero);

fichero=fopen("test2_4.txt", "a+");
s='#';
v=s;
fputc(v,fichero);
fclose(fichero);

clear(buf);

    stringstream ss1;
    string nom1;
    unsigned long int num_arch1 = per;

    ss1 << "Persona_" << num_arch1 << ".txt";
    ss1 >> nom1;

    fichero=fopen(nom1.c_str(), "a");

if (cont_erro>3)
```

```
{
 blit(dummy_error,buf,0,0,300,225,423,317);
 blit(buf,screen,0,0,0,0,1024,768);
 s='D';
 v=s;
 fputc(v,fichero);
 }

else
{s='A';
 v=s;
 fputc(v,fichero);
  }


 s='/';
 v=s;
 fputc(v,fichero);

 ////////////////////////////////////////////////////
fichero2=fopen("test2_1.txt", "a+");
 rewind(fichero2);    //cursor al principio
 text_mode(-1);

while (s!= '#')
   {
     s=fgetc(fichero2);
     v=s;
     fputc(v,fichero);
     k++;
   }
 fclose(fichero2);
 s=' ';

fichero2=fopen("test2_2.txt", "a+");
 rewind(fichero2);    //cursor al principio
 text_mode(-1);
 /**********/
while (s!= '#')
   {
     s=fgetc(fichero2);
     v=s;
     fputc(v,fichero);
     k++;
   }
 fclose(fichero2);
 s=' ';

fichero2=fopen("test2_3.txt", "a+");
 rewind(fichero2);    //cursor al principio
 text_mode(-1);
 /**********/
while (s!= '#')
   {
     s=fgetc(fichero2);
     v=s;
     fputc(v,fichero);
     k++;
   }
 fclose(fichero2);
 s=' ';

fichero2=fopen("test2_4.txt", "a+");
 rewind(fichero2);                                    //cursor al principio
 text_mode(-1);
 /**********/
while (s!= '#')
   {
```

```
     s=fgetc(fichero2);
     v=s;
     fputc(v,fichero);
     k++;
   }
fclose(fichero2);

fclose(fichero);

readkey();
clear_keybuf();
clear(buf);

destroy_bitmap(dummy_error);
destroy_bitmap(c_verde);
destroy_bitmap(c_rojo);
destroy_bitmap(cruz);
destroy_bitmap(parl_nar);
destroy_bitmap(fondo_test2);
destroy_bitmap(tuto2);
destroy_bitmap(negro);
allegro_exit();
return 1;
}

//*******************************--------************************************


int mostrar()
{
  DATAFILE *MisDatos; //con esto comienzo a cargar la nueva fuente
  FONT *MiFuente;     //con esto comienzo a cargar la nueva fuente
  int pers=per;
  int cant=1,l=1,ju=0,detector=0;
  FILE *fichero;
  int x=300;
  int y=255;
  int color=makecol(255,255,255);
  int j=0,p=0,i=0,g1=0;
  char cadena1[150];
  int camp=1;
  char  s;
  PALETTE paleta;
  BITMAP* boton;
  BITMAP* boton1;
  BITMAP* fondo_resul;
  BITMAP* volver;
  BITMAP* volver_1;
  int hor=48,con=1,final=0;
  float posy;
  int mitad=0;


  allegro_init();
  install_keyboard();
  install_mouse();
  set_color_depth(32);
  set_gfx_mode(GFX_AUTODETECT,1024,768,0,0);
  LOCK_VARIABLE(t);
  LOCK_FUNCTION(tiempo);
  t=0;

  fondo_resul=load_bitmap("resul_back.bmp",paleta);  //Fondo de la pantalla princ de
resultados
  set_palette(paleta);
  blit(fondo_resul,screen,0,0,0,0,1024,768);

  while (final!=1) //while gigante que sale con una tecla
```

```
{ fondo_resul=load_bitmap("resul_back.bmp",paleta);  //Fondo de la pantalla princ de
resultados
  set_palette(paleta);
  volver=load_bitmap("volver.bmp",paleta);          //boton interactivo
  volver_1=load_bitmap("volver_1.bmp",paleta);      // botones originales
  set_palette(paleta);
  boton=load_bitmap("boton.bmp",paleta);
  boton1=load_bitmap("boton1.bmp",paleta);


  hor=48;
  con=1;
  camp=1;
  j=0;p=0;i=0;g1=0;cant=1;ju=0;detector=0;l=1;y=255;x=300;



  blit(volver,screen,0,0,800,740,1024,768);
  cant=1;

  MisDatos=load_datafile("tempus.dat");    //aca se carga
  MiFuente=(FONT *)MisDatos[0].dat;        //aca se carga
  text_mode(-1);

  while (cant<=pers)
  { i=0;
    j=-1;   //es porque si no no me toma el primer dato, esto es porque no puedo borrar
el array de char

    stringstream ss;
    string nom;
    unsigned long int num_arch = cant;
    ss << "Persona_" << num_arch << ".txt";
    ss >> nom;

    fichero=fopen(nom.c_str(), "a+");
    rewind(fichero);    //cursor al principio

    do
    {
      j++;
      cadena1[j]=fgetc(fichero);
     }while (cadena1[j]!= '#');
    p=0;
    while (cadena1[i]!= '#')
    {
      textprintf(screen,font,p*20+48,cant*50+10,makecol(255,255,255)," %c",cadena1[i]);
      i++;
      p++;
     }
    blit(boton,screen,0,0,5,cant*50,1024,768);
    cant++;

   fclose(fichero);
   }

show_mouse(screen);  //cualquier cosa volve a screen



while(g1==0)            //el ciclo principal y sus animaciones
{detector=0;
 con=1;
 g1=0;

  if (  ( ((mouse_y>50)||(mouse_y<cant*50+25)) &&(mouse_x>hor) ||
((mouse_y<50)||(mouse_y>cant*50+25)) && ((mouse_x>hor)||(mouse_x<hor))) && ju!=1  )
    {scare_mouse();
```

```
        while (con<=pers)
         {
          blit(boton,screen,0,0,5,con*50,1024,768);
          con++;
          }
     unscare_mouse();


    ju=1;
    }
  if ((mouse_y>50)&&(mouse_y<cant*50+25)&&(mouse_x<hor))
  { ju=0;
    con=pers;
    do
    {if (mouse_y>con*50 && mouse_y<(con*50 + 25))
      {detector=con;
       con=0;}
       con=con-1;
    }while(con>0);

    if (detector>0)
    {scare_mouse();
     blit(boton1,screen,0,0,5,detector*50,1024,768);
     unscare_mouse();}

    con=1;
    while(((mouse_y>detector*50)&&(mouse_y<detector*50+25)&&(mouse_x<hor))&&(g1==0))
     {
       while (con<=pers)
         {   if (con!=detector)
             {scare_mouse();
              blit(boton,screen,0,0,5,con*50,1024,768);
              unscare_mouse();}
          con++;}




       if (mouse_b &1)
       {g1=1;
        mitad=1;}
      }
   }



if (mouse_y>720 && mouse_x>800)

  { scare_mouse();
    blit(volver_1,screen,0,0,800,740,1024,768);
    unscare_mouse();
while ((mouse_y>720 && mouse_x<1015 && mouse_x>800 && mouse_y<768) && g1==0)
  {

  if (mouse_b&1)
 { final=1;
   g1=1;
   }
   }

 scare_mouse();
  blit(volver,screen,0,0,800,740,1024,768);
unscare_mouse();
 }

 }
```

```
  if (mitad==0)
  {
   //blit(fondo_resul,screen,0,0,0,0,800,740);  //para engañar texto
  }
  else
  {blit(fondo_resul,screen,0,580,0,580,650,768);
  mitad=0;}

//hasta aca tengo menu

j=0;
i=0;
 if (detector!=0)
  {  stringstream ss;
    string nom;
    unsigned long int num_arch = detector;             //_-_-_tomo el archivo

    ss << "Persona_" << num_arch << ".txt";
    ss >> nom;
    camp=0;

 fichero=fopen(nom.c_str(), "a+");
 rewind(fichero);    //cursor al principio
 text_mode(-1);
 /**********/
 while (j<150)
    {cadena1[j]='\0';    // con esto logro hacer andar el programa
    j++;}
 j=0;
 /**********/
 while (cadena1[j]!= '/'&& j<150)
   {
     cadena1[j]=fgetc(fichero);
     j++;
   }

 blit(fondo_resul,screen,500,0,500,0,1024,700);//es 700 para que no me sobreescriba el
boton de salida

 do
 {p=0;
 while (cadena1[i]!= '#' && cadena1[i]!= '/')
 {
  textprintf(screen,font,p*10+750,y+300,makecol(255,255,255)," %c",cadena1[i]);
  i++;
  p++;
 }
 y=y+20;
 i++;
 }while (y<358);

 y=255;
do
{p=0;

 while (cadena1[i]!= '#')
 {s=fgetc(fichero);
  textprintf(screen,font,p*10+110,y+330,makecol(255,255,255)," %c",cadena1[i]);
  p++;
  i++;
 }
 y=y+50;
 i++;
}
```

```
while (y<497);
fclose(fichero);
}

 destroy_bitmap(fondo_resul);
 destroy_bitmap(boton);
 destroy_bitmap(boton1);
 destroy_bitmap(volver);
 destroy_bitmap(volver_1);
}                                //Termina el while que mantiene el  submenu activo

 allegro_exit();
 return 1;
}


void GenerateSeed()
{
srand((unsigned)time( NULL ));}

int Random(int liminf,int limsup)
{
int Numero = (rand () % (limsup-liminf+0x1)) + liminf;
return Numero;
}
```