Programación Aplicada y Lab.

PRÁCTICA No. 3

César Mauricio Arellano Velásquez

Raúl González Portillo

Allan Jair Escamilla Hernández

Este documento presenta la resolución de un programa que simula un cajero automático real, que a través del uso de máquinas de estados ejecuta determinados procesos o estados, para esto es necesario introducir señales, que son clave para la continuidad del programa.

1. INTRODUCCIÓN

En el desarrollo de esta práctica se simulará el funcionamiento de un cajero automatico, implementado el uso de máquinas de estado en el lenguaje de programación C. Además, en esta práctica pondremos en práctica muchos de los temas ya vistos en clase, ya que para resolver este problema haremos uso de listas dinamicas, manejo de cadenas, así como también haremos uso de manejo de archivos de texto plano.

1. ANÁLISIS

Para la resolución del problema, se aplicaron los conocimientos visto en clase como: máquinas de estados, listas dinámicas, manejo de cadenas, asignación dinámica de memoria y manejo de archivos de texto plano.

*Función Principal*

En la función principal se encuentra una condicional que evalúa los casos en los que el usuario pueda meter parámetros por terminal, y evalúa que exclusivamente al introducir como parámetro “-c” se ejecute el menú de administrador. Además, en el caso de que no se introduzcan parámetros por la terminal, se arrancará la máquina de estados que simula el funcionamiento de un cajero automático.

Menu Administrador:

1.- Dar de alta cuentahabientes.

2.- Salir.

Menu Cajero:

1.- Ingresar Dinero

2.- Retirar Dinero.

3.- Mostrar Saldo.

4.- Mostrar Movimientos.

5.- Cambiar Contraseña.

6.- Salir.

1. Créditos.En la función presenta los créditos del programa, es decir, los nombres de los desarrolladores, y el objetivo general del programa.
2. CargarBase. Lee de un archivo los usuarios que se encuentran registrados,
3. Login\_Cajero. Evalúa que el número de cuenta y el pin que introdujo el usuario se encuentren en la base de usuarios.
4. Registro. Se ejecuta en el modo administrador para registrar nuevos uuarios.
5. GuardarUsuarios. Guarda en un archivo la lista dinámica de usuarios creados.
6. SolicitarInfo\_BuscarCoincidencia\_SesionIniciada. Una vez que el usuario introdujo su pin y su número de cuenta, evalúa que los datos se encuentren en la lista de usuarios.
7. MsgIngresarDinero. Muestra un mensaje cuando el usuario va ingresar dinero.
8. MsgRetirarDinero. Muestras un mensaje cuando el usuario va a retirar dinero.
9. MostrarSaldo\_MsgMenu. Muestra un mensaje cuando el usuario va a mostrar su saldo.
10. MostrarHistorial\_MsgMenu. Muestra un mensaje cuando el usuario va a visualizar su historial de movimientos.
11. MsgCambiarPassword. Muestra un mensaje al cambiar el password.
12. MsgSalir\_LimpiarLista. Muestra un mensaje al salir y borra la lista dinámica.
13. Msg\_Espera. Muestra un mensaje de espera.
14. MsgMenu. Despliega el menú.
15. SumarDinero\_ActualizarHistorial\_MsgMenu. Suma el dinero del usuario para luego actualizar el historial del usuario.
16. PedirCantidad\_SaldoSuficiente. Pide la cantidad a retirar y evalúa si tiene suficiente saldo.
17. PedirPassword. Pide el pin del usuario.
18. VolverAPedirPassword\_Coinciden. Evalua que coincidan los pines que introdujo el usuario.
19. SesionFallida. Muestra mensaje de que falló el inicio de sesión.
20. MsgSaldoInsuficiente\_MsgMenu. Muestra un mensaje para avisarle al usuario que no cuenta con el saldo suficiente para realizar un retiro.
21. MsgRetiroExitoso\_RestarSaldo\_ActualizarHistorial\_MsgMenu. Muestra un mensaje de que se realizó un retiro exitoso y actualiza el saldo del usuario.
22. MsgErrorCambioPassword\_MsgMenu. Muestra un mensaje avisando al usuario que falló al intentar cambiar el pin.
23. MsgPasswordCambiada\_ActualizarPassword\_MsgMenu. Muestra un mensaje de que se cambió el pin correctamente.
24. Borrar\_Lista. Borra la lista dinámica.
25. ActualizarRegistro. Actualiza los registros de la lista dinámica a un archivo.
26. ActualizarCuentaHabiente. Actualiza los movimientos de un usuario logueado en la máquina de estados.
27. Menu
28. Validacion
29. Nul

III. DISEÑO

**Anexo. Tablas de Máquinas de Estados**

**Anexo. Diagrama Máquina de Estados**

**Anexo. Diagrama IPO Administrador**

**Pseudocódigo**

**Principal**

{

BorrarPantalla( | );

Si(argc<>1 AND argc<>2){

Imprimir("Insertó parámetros de más, intente de nuevo.");

Salir( | );

}Sino{

Si(argc=1){

Creditos( | );

BorrarPantalla( | );

CargarBase( | );

initialise( | );

machine( | );

}

Si(argc=2 AND argv[1] = "-c"){

CopiarCadena (argv[1] | Parametro);

CargarBase( | );

Repite

{

menu( | );

Seleccion(Opcion){

caso 1:

Registro( | );

Romper;

caso 2:

GuardarUsuarios( | );

Imprimir("Se han guardado los datos correctamente");

Imprimir("Saliendo del programa...");

Borrar\_Lista( | );

Salir;

Romper;

Por Defecto:

Imprimir("Opción Inválida");

Romper;

}

imprimir("Presione Enter para continuar...");

}mientras(Opcion<>2);

}sino{

imprimir("Opción inválida, intente de nuevo.");

imprimir("Modo Admin: $./practica3.c -c.");

Salir( | );

}

}

}

}

**CargarBase** ( | ) {

Archivo = AbrirArchivo("CuentaHabientes.txt","Modo Lectura");

Si(Archivo = NULL AND Parametro = "-c"){

Imprimir("Se ha detectado que es la primera ejecución del programa");

Imprimir("Presione Enter para empezar la configuración inicial...");

CopiarCadena("1234567890" | NumCuenta);

Login=1;

Registro( | );

Imprimir("Presione Enter para continuar...");

}Sino{

Si(Archivo=NULL){

Imprimir("Se ha detectado que es la primera ejecución del programa, debido a que no existen los archivos necesarios para funcionar.");

Imprimir("Para continuar es necesario acceder al módulo de administrador.");

Imprimir("Saliendo del programa...");

Salir( | );

}Sino{

Si(Leido=0){

mientras(LeerEnArchivo(User)=1){

new (Nuevo);

CopiarCadena(User | ↑Nuevo.NumCuenta);

LeerEnArchivo(↑Nuevo.Password | );

LeerEnArchivo(↑Nuevo.Usuario | );

LeerEnArchivo(↑Nuevo.Saldo | );

↑Nuevo.sig = NULL;

Si (Inicio <> NULL){

temp = Inicio;

mientras (↑temp.sig <> NULL)

temp = ↑temp.sig;

↑temp.sig = Nuevo;

}Sino

Inicio = Nuevo;

Cuenta = ConvertirCadenaAEntero(↑Nuevo.NumCuenta);

Cuenta += 1;

Leido=1;

}

CerrarArchivo(Archivo | );

}

}

}

}

**menu**( | ){

BorrarPantalla( | );

Imprimir("Bienvenido al modo administrador ");

Imprimir("Menú: ");

Imprimir("1.- Dar de alta cuentahabientes ");

Imprimir("2.- Salir y Guardar Cambios ");

Imprimir("Ingrese la opción que desea realizar-> ");

Leer( | Opcion);

BorrarPantalla( | );

}

**Registro** ( | ){

BorrarPantalla( | );

temp=Inicio;

new (Nuevo);

Imprimir("Ingrese nombre del cuentahabiente-> ");

Leer( | ↑Nuevo.Usuario);

Imprimir("Ingrese PIN del cuentahabiente-> ");

LeerCadena(↑Nuevo.Password | );

validacion(↑Nuevo.Password | );

Si(Validacion1 = 1 OR Validacion2 = 1){

Imprimir("El PIN debe incluir solo numeros y únicamente incluir 4 números. ");

Validacion1 = 0;

Validacion2 = 0;

}

↑Nuevo.Saldo=0;

Si(Login=1){ //Si es la primera ejecución...

CopiarCadena(NumCuenta | ↑Nuevo.NumCuenta);

Cuenta = ConvertirCadenaAEntero(↑Nuevo.NumCuenta | );

Login=0;

}Sino

ConvetirEnteroaCadena(↑Nuevo.NumCuenta, Cuenta | );

Imprimir(" El número de cuenta generado es: ", ↑Nuevo.NumCuenta);

Cuenta+=1;

↑Nuevo.sig=NULL;

Si (Inicio <> NULL){

temp = Inicio;

mientras (↑temp . sig <> NULL)

temp = ↑temp . sig;

↑temp . sig = Nuevo;

}Sino

Inicio = Nuevo;

}

**validacion**(Password | ){

BorrarPantalla( | );

Si ((LongitudCadena(Password)) <> 4)

Validacion1 = 1;

Desde i = 0; Hasta i < 4; i ++){

Si (Password [i] < '0' OR Password [i] > '9'){

Validacion2 = 1;

}

}

}

**GuardarUsuarios** ( | ) {

temp=Inicio;

Archivo = AbrirArchivo("CuentaHabientes.txt","Modo de Escritura");

mientras(temp<>NULL){

ImprimirEnArchivo(↑temp.NumCuenta | );

ImprimirEnArchivo(↑temp.Password | );

ImprimirEnArchivo(↑temp.Usuario | );

ImprimirEnArchivo(↑temp.Saldo | );

temp=↑temp.sig;

}

CerrarArchivo(Archivo | );

}

**Borrar\_Lista**( | ){

temp=Inicio;

mientras(temp<>NULL){

Inicio=↑Inicio.sig;

free(temp | );

temp=Inicio;

}

}

**Login\_Cajero**( | ){

temp=Inicio;

mientras(temp <> NULL){

si(CompararCadenas(NumCuenta,↑temp.NumCuenta | )=0 AND CompararCadenas(Password,↑temp.Password | )==0){

CopiarCadena(↑temp.Usuario | Usuario);

Saldo=↑temp.Saldo;

Login=1;

Cuenta-=1;

}

temp = ↑temp.sig;

}

}

**SolicitarInfo\_BuscarCoincidencia\_SesionIniciada** ( | ){

Sistema("limpiar");

Imprimir("Por favor ingrese sus datos para entrar a su cuenta:");

Imprimir("Número de cuenta-> ");

Leer ( | NumCuenta);

Imprimir ("PIN-> ");

Leer ( | Password);

Login\_Cajero( | );

Sistema("limpiar");

Si(Login = 1){

Imprimir("Ha iniciado sesión con éxito");

Regresar 1;

} Si no

Regresar 0;

}

**MsgIngresarDinero** ( | ){

imprimir("Accion: Ingresar Dinero");

imprimir("1.-Ingresa Dinero con (d)");

imprimir("2.-Cancelar acción con (!)");

imprimir("Seleccionar opcion: ");

}

**MsgRetirarDinero** ( | ){

imprimir("Accion: Retirar Dinero");

imprimir("1.-Retirar Dinero con D");

imprimir("2.-Cancelar acción con #");

imprimir("Seleccionar opcion: ");

}

**MostrarSaldo\_MsgMenu** ( | ){

imprimir("Accion: Mostrar Saldo");

imprimir("Su saldo actual es de: ",Saldo);

imprimir("Presione Enter para continuar...");

MsgMenu( | );

}

**MostrarHistorial\_MsgMenu** ( | ){

imprimir("Accion: Mostrar Movimientos. ");

imprimir("Historial de Movimientos: ");

abrirArchivo(temp, "rt");

si(Archivo<>NULL){

mientras(leerdeArchivo(Temp,99,Archivo)<>NULL){

Temp[longiud(Temp)-1]='\0';

imprimir(Temp);

}

Cerrar(Archivo);

}

sino

imprimir("No hay movimientos en su cuenta. ");

imprimir("Presione Enter para continuar... ");

MsgMenu( | );

}

**MsgSalir\_LimpiarLista** ( | ){

Imprimir("Saliendo del Menú...");

GuardarUsuarios( | );

Borrar\_Lista( | );

Leido=0;

Login=0;

Msg\_Espera( | );

}

**MsgCambiarPassword ( | )**{

imprimir("Accion: Cambiar PIN ");

imprimir("1.- Cambiar PIN con (Z) ");

imprimir("2.- Cancelar Acción con ($) ");

imprimir("Seleccionar opción: ");

}

**MsgMenu** ( | ){

Sistema("limpiar");

Imprimir("Bienvenido”, Usuario, “al menú de acciones:");

Imprimir("1.- Ingresar dinero, ingrese (i)");

Imprimir("2.- Retirar dinero, ingrese (R)");

Imprimir("3.- Mostrar Saldo, ingrese (C)");

Imprimir("4.- Mostrar historial, ingrese (M)");

Imprimir("5.- Cambiar PIN, ingrese (P)");

Imprimir("6.- Salir del menú, ingrese (Q)");

Imprimir("Seleccione una opción-> ");

}

**Msg\_Espera ( | )**{

imprimir("Bienvenido a: ");

imprimir (" \_\_\_\_ \_ \_ \_ \_ \_ ");

imprimir (" | \_\_ ) \_\_ \_ \_ \_\_ \_\_\_ \_\_\_ \_\_\_ \_\_ \_ \_ \_\_ (\_) |\_ \_\_ \_| (\_)\_\_\_| |\_ \_\_ \_ ");

imprimir (" | \_ | / \_` | '\_ | / \_\_/ \_ | / \_\_/ \_` | '\_ || | \_\_/ \_` | | / \_\_| \_\_/ \_` | ");

imprimir (" | |\_) | (\_| | | | | (\_| (\_) | | (\_| (\_| | |\_) | | || (\_| | | |\_\_ | || (\_| ");

imprimir (" |\_\_\_\_/ |\_\_,\_|\_| |\_||\_\_\_|\_\_\_/\_ |\_\_\_|\_\_,\_| .\_\_/|\_||\_\_|\_\_,\_|\_|\_|\_\_\_/|\_\_|\_\_,\_| ");

imprimir (" \_\_ \_ \_\_\_ \_ \_\_ \_\_\_ \_ \_\_(\_) \_\_\_ \_\_\_ |\_| ");

imprimir (" / \_` |/ \_ | '\_ | / \_ | '\_\_| |/ \_\_/ \_ | ");

imprimir (" | (\_| | \_\_/ | | | \_\_/ | | | (\_| (\_) | ");

imprimir (" |\_\_, ||\_\_\_|\_| |\_||\_\_\_|\_| |\_||\_\_\_|\_\_\_/ ");

imprimir (" |\_\_\_/ ");

imprimir("Por favor ingrese I para continuar: ");

}

**PedirCantidad\_SaldoSuficiente** ( | ){

temp=Inicio;

sistema("limpiar");

Imprimir("Ingresa el monto a retirar: ");

Leer( | Dinero);

mientras(temp<>NULL){

Si((CompararCadenas(Usuario,↑temp.Usuario))=0){

Si(↑temp.Saldo >=Dinero){

↑temp.Saldo-=Dinero;

Saldo=↑temp.Saldo;

LeerHora( | Bitacora);

Concatenar (", Mov: Retirar Dinero, Monto retirado: " Bitacora);

ImprimirEnCadena (Dinero | SaldoActual);

Concatenar(SaldoActual | Bitacora);

Concatenar(", Saldo Actual: " | Bitacora);

ImprimirEnCadena(temp->Saldo | SaldoActual);

Concatenar(SaldoActual | Bitacora);

regresar 3;

}

else

regresar 2;

}

temp=↑temp.sig;

}

}

SumarDinero\_ActualizarHistorial\_MsgMenu ( | ){

temp=Inicio;

imprimir("Ingresa el monto a ingresar: ");

leer(Dinero);

mientras(temp<>NULL){

si(Usuario == ↑temp.Usuario){

↑temp.Saldo+=Dinero;

Saldo=↑temp.Saldo;

imprimir(Bitacora,200,"%d/%m/%Y - %H:%M:%S",timeinfo);

concatenar(Bitacora,", Mov: Ingresar Dinero, Monto Ingresado: ");

imprimir(SaldoActual,"%d",Dinero);

concatenar(Bitacora,SaldoActual);

concatenar(Bitacora,", Saldo Actual: ");

imprimir(SaldoActual,"%lld",↑temp->Saldo);

concatenar(Bitacora,SaldoActual);

}

temp=↑temp->sig;

}

ActualizarRegistro( | );

MsgMenu( | );

}

**MsgPasswordCambiada\_ActualizarPassword\_MsgMenu ( | )**{

temp = Inicio;

mientras(temp<>NULL){

si(Usuario = ↑temp.Usuario)

copiarCadena(↑temp.Password,NewPassword);

temp=↑temp.sig;

}

GuardarUsuarios( | );

imprimir("La PIN se modificó éxitosamente. ");

imprimir("Presione Enter para continuar... ");

MsgMenu( | );

}

**PedirPassword ( | )**{

imprimir("Ingrese nuevo PIN (Recuerde ingresar 4 dígitos) -> ");

leer(NewPassword);

imprimir("Accion: Comprobando PIN ");

imprimir("Comprobar PIN con (p) ");

imprimir("Seleccionar opción: ");

}

**VolverAPedirPassword\_Coinciden** (void){

Sistema("limpiar");

Imprimir("Ingrese de nuevo la PIN");

Leer ( | Comprobar);

Validacion (Comprobar | );

Si (Validacion1 = 1 OR Validacion2 = 1){

Imprimir("El PIN debe incluir solo números y únicamente incluir 4 números.");

Validacion1 = 0;

Validacion2 = 0;

Regresar 1;

}

Si((CompararCadenas(NewPassword,Comprobar | ))==0){

Regresar 5;

}

else

Regresar 4;

}

**SesionFallida ( | )**{

imprimir("Usuario y/o PIN son incorrectos ");

imprimir("Intente de nuevo ");

imprimir("Presione Enter para continuar... ");

Msg\_Espera( | );

}

**MsgSaldoInsuficiente\_MsgMenu** ( | ){

Sistema("limpiar");

Imprimir("La operación falló, no hay saldo suficiente");

Imprimir("Intente de nuevo");

Imprimir("Presione Enter para continuar...");

Leer ( | );

MsgMenu( | );

}

**MsgRetiroExitoso\_RestarSaldo\_ActualizarHistorial\_MsgMenu ( | )**{

imprimir("Retiro Éxitoso ");

imprimir("Presione Enter para continuar... ");

ActualizarRegistro( | );

MsgMenu( | );

}

**MsgErrorCambioPassword\_MsgMenu** ( | ){

Sistema("limpiar");

Imprimir("Error, PINs no coinciden");

Imprimir("Intente nuevamente");

Imprimir("Presione Enter para continuar...");

Leer( | );

MsgMenu( | );

}

**nul**( | ){

Imprimir("Opción Inválida");

}

**validacion(Password | )**{

Si (longitud(Password) <> 4)

Validacion1 = 1;

Para i = 0 hasta i < 4{

Si (Password [i] < '0' OR Password [i] > '9'){

Validacion2 = 1;

}

}

}

IV CONCLUSIONES

A lo largo del desarrollo de esta práctica hemos podido reforzar nuestros conocimientos con respecto a máquinas de estado, así como también hemos podido combinar gran parte de los conocimientos adquiridos en parciales pasados y en el curso de Fundamentos de Programación. Esta práctica nos ha servido de ayuda para comprender de una mejor manera el funcionamiento que tiene este “paradigma” de programación (máquinas de estados), y cómo es que podemos realizar una implementación exitosa al hacer uso de los distintos de temas que hemos estudiado a lo largo del curso de Programación Aplicada.

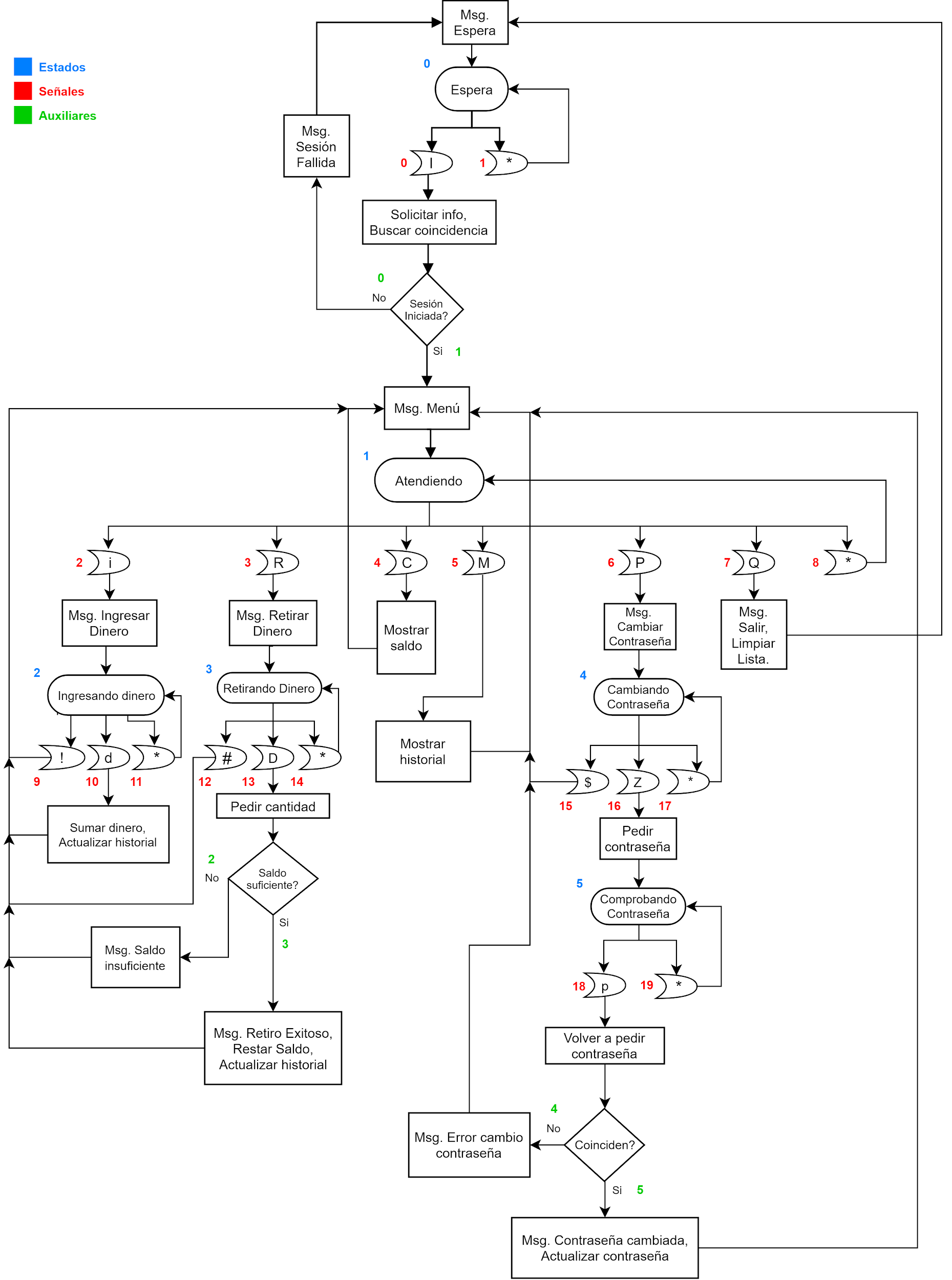
V REFERENCIAS

[1] P. sznajdleder Algoritmos a fondo con implementaciones en C y Java, 5, Argentina.

Alfaomega, 2017, 513-520

**VI ANEXOS**

**Diagrama Máquina de Estados**

****

**Tablas Máquina de Estados.**

Tabla de estados:

|  |  |  |
| --- | --- | --- |
| **Estado** | **Inicio** | **Fin** |
| 0 - Espera | 0 | 1 |
| 1 - Atendiendo | 2 | 8 |
| 2 - IngresandoDinero | 9 | 11 |
| 3 - RetirandoDinero | 12 | 14 |
| 4 - CambiandoPassword | 15 | 17 |
| 5 - ComprobandoPassword | 18 | 19 |

Tabla de acciones:

|  |  |  |  |
| --- | --- | --- | --- |
| **Señal / Evento** | **Acción** | **Más acciones** | **Siguiente estado** |
| 0 - I | SolicitarInfo\_BuscarCoincidencia\_ SesionIniciada? | 0 - Si hay | -1 - No se sabe |
| 1 - \* | NULL | -1 No hay | 0 - Espera |
| 2 - i | MsgIngresarDinero | -1 No hay | 2 - IngresandoDinero |
| 3 - R | MsgRetirarDinero | -1 No hay | 3 - RetirandoDinero |
| 4 - C | MostrarSaldo\_MsgMenu | -1 No hay | 1 - Atendiendo |
| 5 - M | MostrarHistorial\_MsgMenu | -1 No hay | 1 - Atendiendo |
| 6 - P | MsgCambiarPassword | -1 No hay | 4 - CambiandoPassword |
| 7 - Q | MsgSalir\_LimpiarLista | -1 No hay | 0 - Espera |
| 8 - \* | NULL | -1 No hay | 1 - Atendiendo |
| 9 - ! | MsgMenu | -1 No hay | 1 - Atendiendo |
| 10 - d | SumarDinero\_ActualizarHistorial\_MsgMenu | -1 No hay | 1 - Atendiendo |
| 11 - \* | NULL | -1 No hay | 2 - IngresandoDinero |
| 12 - # | MsgMenu | -1 No hay | 1 - Atendiendo |
| 13 - D | PedirCantidad\_SaldoSuficiente? | 0 - Si hay | -1 - No se sabe |
| 14 - \* | NULL | -1 No hay | 3 - RetirandoDinero |
| 15 - $ | MsgMenu | -1 No hay | 1 - Atendiendo |
| 16 - Z | PedirPassword | -1 No hay | 5 - ComprobandoPassword |
| 17 - \* | NULL | -1 No hay | 4 - CambiandoPassword |
| 18 - p | VolverAPedirPassword\_Conciden? | 0 - Si hay | -1 - No se sabe |
| 19 - \* | NULL | -1 No hay | 5 - ComprobandoPassword |

Tabla auxiliar:

|  |  |  |
| --- | --- | --- |
| **Acción** | **Más acciones** | **Siguiente estado** |
| 0 - SesionFallida | -1 No hay | 0 - Espera |
| 1 - MsgMenu | -1 No hay | 1 - Atendiendo |
| 2 - MsgSaldoInsuficiente\_MsgMenu | -1 No hay | 1 - Atendiendo |
| 3 - MsgRetiroExitoso\_RestarSaldo\_ ActualizarHistorial\_MsgMenu | -1 No hay | 1 - Atendiendo |
| 4 - MsgErrorCambioPassword\_MsgMenu | -1 No hay | 1 - Atendiendo |
| 5 - MsgPasswordCambiada\_ ActualizarPassword\_MsgMenu | -1 No hay | 1 - Atendiendo |

**Diagrama IPO Administrador.**

****