UNIVERSIDAD MARIANO GALVEZ DE GUATEMALA

INGENIERIA EN SISTEMAS

SEDE MAZATENANGO

ALGORITMOS

SEGUNDO SEMESTRE SECCION “C”

MANUAL TECNICO

ALPACA

INTEGRANTES:

Benigno Ricardo Coj Santay

Bryan Ernesto Ventura Castaneda

David Alejandro Rojas Morales

German Francisco Armas Argueta

Sergio Paolo López Morales

Mazatenango, noviembre de 2024

MANUAL TECNICO

ALPACA

INDICE

1. Objetivos

Se ha creado dicho documento con el propósito de mostrar como fue diseñado el sistema, y al mismo tiempo dar referencias de como interactuar con el programa para que sea actualizado o al mismo tiempo se le de un mantenimiento adecuado en caso de un fallo

A grandes rasgos se diseño con el mero propósito de guiar al programador que este al frente de dicho sistema como se hizo, su proceso de instalación, código fuente, etc.

1.1. Objetivos específicos

- Guía de instalación del sistema
- Mostrar código fuente para una posible actualización del sistema en el futuro
- Requisitos para la ejecución de dicho programa
- Manifestar evidencias del diseño del sistema antes de implementarlo

2. Alcance

Este documento está dirigido a: Programador

Conocimientos básicos en: Programación en C++

3. Requerimientos técnicos

3.1. Software

Dev-C++: Dev-C++ es un entorno desarrollo integrado (IDE) para los lenguajes de programación C/C++. Usa Mingw basado en GCC (GNU Compiler Collection) como compilador. Puede crear ejecutables para Windows, ya sea en modo consola o GUI gráfico, así como también DLLs y bibliotecas estáticas.

3.2. Hardware

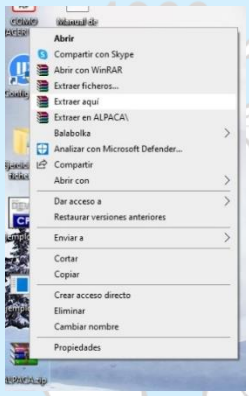
- Una computadora completa (bocinas no necesarias): esto incluye ratón, teclado, cpu, monitor.
- Como complemento se podría utilizar equipo touch (computadoras con monitores táctiles para evitar el uso del ratón).

3.3. Requerimientos mínimos de hardware

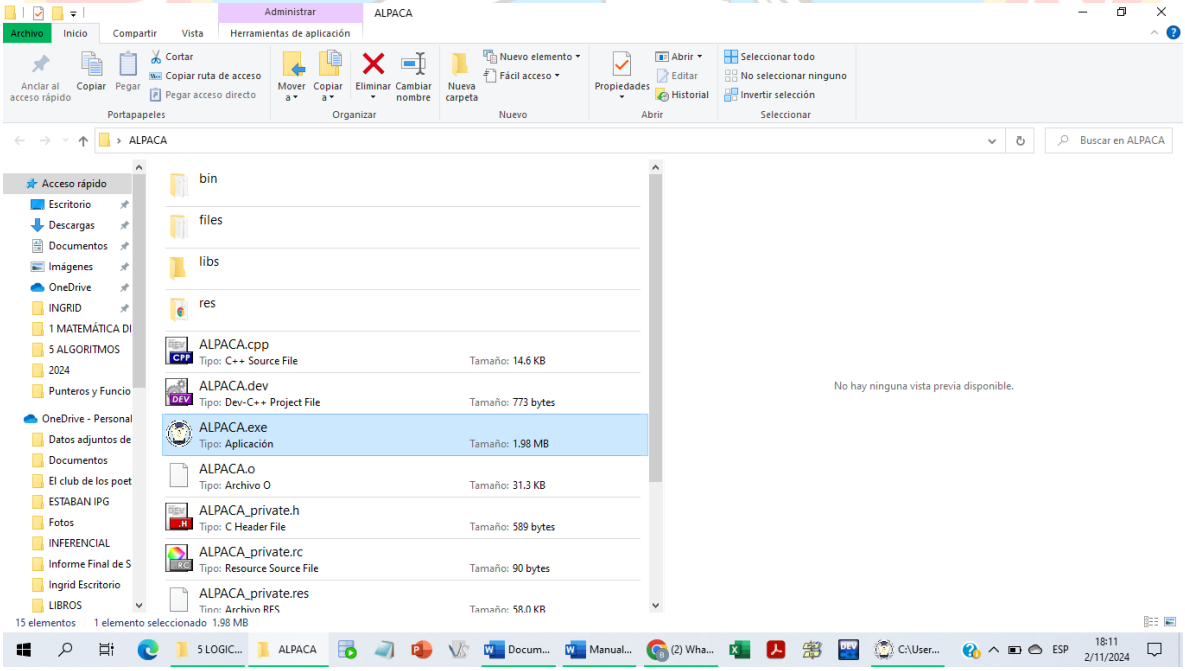
- Procesador: Intel inside 1.5 ghz
- Memoria RAM (mínimo): 512 mb
- Disco Duro: 64 gb

4. Instalación

Luego de adquirir el programa se le proporcionara un archivo zip, el cual debe descomprimir utilizando la opción de extraer aquí.



Después dentro de la carpeta que tendrán luego de descomprimir buscar el archivo ALPACA.EXE el cual es el ejecutable para iniciar el programa.



5. Desarrollo

```
#include<cstdlib>
#include<iostream>
#include<iomanip>
#include<fstream>
#include<string.h>
#include<windows.h>
#include<sstream>
#include <ctime>
#define colorines SetConsoleTextAttribute//Esto es para acortar y no tener que escribir SetConsoleTextAttribute
using namespace std;


//Ajustado de Pantalla
void ajustarTamanioVentana(int ancho, int alto) {//Esta funcion adapta el tamaño de la consola
    HANDLE hConsole = GetStdHandle(STD_OUTPUT_HANDLE);
    COORD bufferSize = { (SHORT)ancho, (SHORT)alto };
    SetConsoleScreenBufferSize(hConsole, bufferSize);
    SMALL_RECT ventana = { 0, 0, (SHORT)(ancho - 1), (SHORT)(alto - 1) };
    SetConsoleWindowInfo(hConsole, TRUE, &ventana);
}
/*Codigo por gpt modificado por mi, vi videos acerca del tema pero, o no servian o no me convencieron eran  extensos y los resultados no eran muy buenos, o daban error*/
// Función para desencriptar y mostrar el contenido del archivo binario, para ver el contenido original sin encriptar vea la carpeta bin en los .txt
void desencriptarArchivoBinario(const string& nombreArchivoEntrada, char clave) {
    ifstream archivoEntrada(nombreArchivoEntrada.c_str(), ios::binary); // Abrir archivo en binario
```

```
    if (!archivoEntrada) {
        cerr << "[Error "<<nombreArchivoEntrada<< "binario no cargado]" << endl;
//esta funcion no la conocia, es cout pero de tipo errores
        return;
    }

    char caracter;
    //cout <<nombreArchivoEntrada<< endl;

    while (archivoEntrada.read(&caracter, sizeof(caracter))) { // Leer cada carácter
encriptado
        char desencriptado = caracter ^ clave; // Desencriptar usando XOR
        cout << desencriptado; // Mostrar el carácter desencriptado en la consola
    }

    archivoEntrada.close();
    cout << endl; // Nueva línea al final de la salida de cada archivo
}

//====|Productos


const int MAX_PROD = 100;
int NOproducto, totalProductos, noventa, codigo_producto, existencia, cantidad,
codigoBuscar;
int numProductos = 0, contador = 1;

char producto[256];
```

```
float precio_unitario, precio_compra, totalc, preciot, pagot;

string linea;

//Funcion para quitar un producto
void quitarprod() {
    ifstream productos("files/Productos.txt");
    ofstream temp("files/temp.txt");
    int numBorrar;
    string linea;

    if (!productos.is_open() || !temp.is_open()) {
        cout << "Error al abrir el archivo." << endl;
        return;
    }

    cout << "Ingrese el numero de producto que desea quitar: ";
    cin >> numBorrar;

    bool encontrado = false;
    while (getline(productos, linea)) {
        stringstream ss(linea);
        ss >> codigo_producto >> producto >> existencia;

        if (codigo_producto == numBorrar) {//Si el No coincide con el que buscamos
eliminar se elimina
```

```
            encontrado = true;
        } else {//si no no guarda ningun tipo de cambio
            temp << codigo_producto << ' ' << producto << ' ' << existencia << endl;
        }
    }

    productos.close();
    temp.close();

    remove("files/Productos.txt");
    rename("files/temp.txt", "files/Productos.txt");

    if (encontrado) {
        cout << "Producto eliminado exitosamente." << endl;
    } else {
        cout << "Producto no encontrado." << endl;
    }

    system("pause");
}

//Funcion de buscar un producto
void buscarprod() {
    ifstream productos("files/Productos.txt");

    bool encontrado = false;

    if (!productos.is_open()) {
```

```
        cout << "Error al abrir el archivo." << endl;
        return;
    }

    cout << "Ingrese el código (No.) del producto a buscar: ";
    cin >> codigoBuscar;

    // Recorremos cada línea del archivo
    while (getline(productos, linea)) {
        stringstream ss(linea);
        ss >> codigo_producto >> producto >> cantidad;

        // Si el código coincide, mostramos la información del producto
        if (codigo_producto == codigoBuscar) {//si el no del producto coincide con el que buscamos lo muestra
            cout << contador << ") " << "No. " << codigo_producto << " - Producto: "
                 << producto << " - Cantidad: " << cantidad << endl;
            encontrado = true;
            contador++;
        }
    }

    if (!encontrado) {
        cout << "Producto no encontrado." << endl;
    }

    productos.close();
    system("pause");
```

```
}

//Funcion para mostrar todo el inventario
void inventario() {
    ifstream productos("files/Productos.txt");



    if (!productos.is_open()) {
        cout << "Error al abrir el archivo." << endl;
        return;
    }

    cout << left << setw(10) << "No." << setw(30) << "Producto" << setw(10) <<
"Cantidad" << endl;
    cout << "-------------------------------------------------" << endl;

    while (getline(productos, linea)) {// funcion para leer todo el txt hasta que
encuentre lineas vacias
        stringstream ss(linea);
        ss >> codigo_producto >> producto >> existencia;
        cout << left << setw(10) << codigo_producto << setw(30) << producto <<
setw(10) << existencia << endl;
    }
          cout<<endl;
    productos.close();
    system("pause");
}
//========|Aciones
//Codigos por parte de German, editados por Bryan
```

```
//Funcion de Comprar Productos
void compras(){
        ofstream escritura;
        ofstream agregar;
   escritura.open("files/compras.txt", ios::out | ios::app); //Se crea registro de compra
   agregar.open("files/Productos.txt", ios::out | ios::app); //se crea la de productos tambien
   if (escritura.is_open()) {
      cout << "Ingrese el Codigo del Nuevo Producto a Comprar: "; cin >> codigo_producto;
      cout << "Ingrese el Nombre del Nuevo Producto: "; cin.ignore(); cin.getline(producto, sizeof(producto));
      cout << "Ingrese la Cantidad del Nuevo Producto: "; cin >> existencia;
      cout << "Ingrese el Precio Unitario de Compra: "; cin >> precio_unitario;
                totalc = existencia * precio_unitario;
                cout << "Total a Pagar: "<<totalc<<endl;
      while (precio_unitario){
      cout << "Ingrese el Precio de Pago: "; cin >> precio_compra;

                if(precio_compra<totalc){//si el precio de pago es menor al de pagar hara:
                        cout<<"El precio de Pago es menor al de pagar"<<totalc<<endl;
                        }else{
                        precio_compra = precio_compra;
                        break;
                        }}
      escritura << codigo_producto << " " << producto << " " << existencia << " " << precio_unitario << " " << precio_compra << endl;
```

```
        agregar << codigo_producto << " " << producto << " " << existencia << " " <<
endl;
                cout << "Producto agregado exitosamente." << endl;
        system("pause");
        ofstream registrocompras("files/comprastotal.txt", ios::out | ios::app);
//creamos un registro de compras totales
                                if (registrocompras.is_open()) {
                                float totalc;

                        registrocompras << codigo_producto << producto << " " <<
existencia <<" "<< precio_unitario <<" "<< totalc<< endl;
                        registrocompras.close();
    }     else {
                        cout << "Error al abrir el archivo 'comprastotal.txt'" << endl;
                                } }else {
        cout << "Error, el Archivo No se Pudo Abrir o No ha sido Creado" << endl;
    }
    escritura.close();
}
//Funcion para mostrar todas las compras
void totalcompras(){
          ifstream registrocompras;
    registrocompras.open("files/comprastotal.txt", ios::in);
    if (registrocompras.is_open()) {
          float totalc;
        cout << "No." << setw(10) << "Producto" << setw(15) << "Cantidad"
<<setw(20)<<"Valor unitario"<<setw(20)<<"Valor Total" <<endl;
                cout << "-------------------------------------------------------------------------" <<
endl;
```

```
        while (registrocompras >> codigo_producto >> producto >> existencia >>
precio_unitario >> totalc) {

            cout << codigo_producto << setw(10) << producto << setw(15) <<
existencia << setw(16) << precio_unitario << setw(23) << totalc << endl;

        }

    } else {

        cout << "Error al abrir el archivo 'comprastotal.txt'" << endl;

    }

    registrocompras.close();

    system("pause");

}

//FUncion para realizar las ventas

void ventas(){


        ofstream escritura;

    escritura.open("files/ventas.txt", ios::out | ios::app);//se crea un registro de venta

    if (escritura.is_open()) {

        cout << "Ingrese el no de Venta: "; cin>>noventa;

        cout << "Ingrese el Nombre del Producto a Vender: "; cin.ignore();
cin.getline(producto, sizeof(producto));

        cout << "Ingrese Cantidad del producto a vender: "; cin >> cantidad;

                cout << "Ingrese el Valor Unitario de Venta del Producto: "; cin >>
precio_compra;

        preciot= cantidad*precio_compra;

        cout << "Valor a recibir: "<<preciot<<endl; cin.ignore();

        while (pagot){

        cout << "Ingrese el Pago de la Venta: "; cin>>pagot;

                if(pagot<preciot){//igual que en el de comprar, solo que envez de
pagar, lo que se tiene que recibir

                        cout<<"El Pago es menor al que se debe recibir"<<endl;
```

```
                    }else{
                    ;
                    break;
                    }}
        escritura << noventa<< " "  << " " << producto << " " << cantidad << " " <<
precio_compra << " " << preciot<< " "<<pagot << endl;
                cout << "Venta exitosa." << endl;
        system("pause");
        ofstream registroventas("files/toventas.txt", ios::out | ios::app);
                        if (registroventas.is_open()) {
                    registroventas << noventa << " " << producto << " " <<
cantidad <<" "<< precio_compra <<" "<< preciot<< endl;
                    registroventas.close();
    }     else {
                    cout << "Error al abrir el archivo 'tventas.txt'" << endl;
                        } }else {
        cout << "Error, el Archivo No se Pudo Abrir o No ha sido Creado" << endl;
    }
    escritura.close();
}
//Funcion que muestra todas las ventas realizadas
void totalventas() {
    ifstream registroVentas("files/toventas.txt");
    cout << left << setw(10) << "No." << setw(30) << "Producto" << setw(10) <<
"Cantidad" << setw(15) << "Precio Unitario" << setw(15) << "Precio Total" << endl;
    cout << "----------------------------------------------------------------" << endl;

    while (registroVentas >> noventa>> producto >> cantidad >> precio_compra >>
preciot) {
```

```
        cout <<noventa << setw(10) << producto << setw(30) << cantidad <<
setw(10) << precio_compra << setw(15) << preciot << endl;
    }
    registroVentas.close();
    system("pause");
}
//====|Otros
void limpiarregistros(const string& nombreArchivo) {
    ofstream archivo(nombreArchivo.c_str(), ios::trunc);

    if (!archivo.is_open()) {
        cout << "Error al abrir el archivo: " << nombreArchivo << endl;
        return;
    }

    archivo.close();
    cout << "El archivo " << nombreArchivo << " ha sido limpiado exitosamente." <<
endl;
}


int main(){
        ajustarTamanioVentana(85, 32);
        int opcion, codigobinario=1, espera=1;
        char sino;
        const char* rutaarchivo ="MegaPaca.html";
        char clave=5;
        HANDLE hConsole= GetStdHandle(STD_OUTPUT_HANDLE);//Esto es
para activar en evento del cambio del color de texto
```

```
        PlaySound(TEXT("res/snds/ON.wav"), NULL, SND_FILENAME |
SND_ASYNC | SND_LOOP);//Un sonido que edite
        for(int conteo=1; conteo<=2; conteo++){
                cout<<"Iniciando Sistema";
                for(int puntos=1;puntos<=3;puntos++){
                        cout<<'.';
                        Sleep(100);
                }
                Sleep(250);
                cout<<endl; system("cls");
        }

        do{
                system("cls");
                colorines(hConsole, 12);//Esto es para darle color solo al texto en
valor numerico, del 1 hasta el infinito, apartir del 30 creo ya empiesan con
subrayado
                //Este es el nombre encriptado
                desencriptarArchivoBinario("bin/ALPACA.bin", clave);//la funcion para
texto encriptario en formato binario
                Sleep(1000);
                if(espera==1){
                        Sleep(800);
                        //PlaySound(TEXT("res/snds/Chase the truth.wav"), NULL,
SND_FILENAME | SND_ASYNC | SND_LOOP);//Una cancion que hice
                        PlaySound(TEXT("res/snds/DooM - Ats Dooms
gate(Synthwave).wav"), NULL, SND_FILENAME | SND_ASYNC | SND_LOOP);
                        espera=0;
                }
```

```
colorines(hConsole, 10);
//Este es el menu encriptado(ver txt en la carpeta bin para ver opciones sin encriptar)
desencriptarArchivoBinario("bin/MENU.bin", clave);
cin>>opcion;
switch(opcion){
	case 1://Parte Paolo
		system("cls");
	break;

	case 2://Parte Paolo
		system("cls");
	break;

	case 3://Parte Paolo
		system("cls");
		ShellExecute(0, "open", "res/ALPACA.html", 0, 0, SW_SHOWNORMAL);	//Abre enlaces .html
	break;

	case 4://Parte German
		system("cls");
		compras();
	break;

	case 5://Parte German
		system("cls");
		ventas();
	break;
```

```
case 6://Parte Benigno
        system("cls");
        totalventas();
break;

case 7://Parte Benigno
        system("cls");
        totalcompras();
break;

case 8://Parte David
        system("cls");
        quitarprod();
break;

case 9://Parte Benigno
        system("cls");
        buscarprod();
break;

case 10://Parte David
        system("cls");
        inventario();
break;

case 11:
        system("cls");
```

```
                cout<<"Esta seguro de que desea Eliminar los
registros?"<<endl<<setw(18)<<"[Si/No]"<<endl;
                cin>>sino;

                if (sino=='s'||sino=='S')
                {
                cin.ignore();
                break;
                }
                else
                { system("cls");
                cin.ignore();
                limpiarregistros("files/Productos.txt");
                limpiarregistros("files/compras.txt");
                limpiarregistros("files/ventas.txt");
                limpiarregistros("files/comprastotal.txt");
                limpiarregistros("files/toventas.txt");
                break;
                }
        break;

        case 12:
                cout<<"Esta seguro de que desea
Salir?"<<endl<<setw(18)<<"[Si/No]"<<endl;
                cin>>sino;

                if (sino=='s'||sino=='S')
                { cin.ignore(); codigobinario=0; }
                else
```

```
                    { system("cls"); cin.ignore(); break; }
                    break;
                default:
                    //system("cls");
                    cout<<"[ERROR]:Opcion Inexistente, Intente otra vez."<<endl;
                    break;
            }
        }while(codigobinario!=0);
        system("cls");
        PlaySound(NULL, NULL, 0);

        colorines(hConsole, 12);
        desencriptarArchivoBinario("bin/ALPACA.bin", clave);//la funcion
        PlaySound(TEXT("res/snds/OFF.wav"), NULL, SND_FILENAME);//Un sonido que edite
        colorines(hConsole, 15);//color normal blanco
        system("pause"); return 0;
}
```