```cpp
/*
 * Pablo_Villa 874773
 * 01/12/23
 */

#include <iostream>
using namespace std;

const int MAX_PRODUCTOS = 100;
const int PRODUCTO_NO_ENCONTRADO = -1;
const int ALTA = 1;
const int BAJA = 2;
const int LISTADO = 3;
const int ENTRADA = 4;
const int SALIDA = 5;
const int BUSCAR = 6;
const int FIN = 7;

struct producto_almacen {
    int codigo = 0;
    string descripcion = "";
    int existencias = 0;
};

void anadir_producto(producto_almacen productos[], int &numero_productos);
void leer_producto(producto_almacen &producto);
void listar_productos(producto_almacen productos[], int numero_productos);
void anadir(producto_almacen productos[], int &numero_productos);
void eliminar(producto_almacen productos[], int &numero_productos);
void borrar_producto(producto_almacen productos[], int &numero_productos);
void buscar(producto_almacen productos[], int &numero_productos);
int busqueda(const producto_almacen productos[], int codigo, int numero_productos);
void cargar_datos_desde_archivo(componente componentes[], int &numero_componentes);
void guardar_datos_en_archivo(componente componentes[], int numero_componentes);

const string NOMBRE_ARCHIVO = "datos_componentes.txt";

int main(){
    producto_almacen productos[MAX_PRODUCTOS];
    int opcion = 0;
    int numero_productos = 0;

    cargar_datos_desde_archivo(componentes, numero_componentes);

    while(opcion != FIN){
        cout << ALTA << "->Alta " << BAJA << "->Baja " << LISTADO << "->Listado " << ENTRADA
             << "->Entrada " << SALIDA << "->Salida " << BUSCAR << "-> Buscar " << FIN << "->Acabar: ";

        cin >> opcion;

        switch(opcion){
            case ALTA:
                anadir_producto(productos, numero_productos);
                break;

            case BAJA:
                borrar_producto(productos, numero_productos);
                break;

            case LISTADO:
                listar_productos(productos, numero_productos);
                break;

            case ENTRADA:
                anadir(productos, numero_productos);
                break;
```

```
67
68              case SALIDA:
69                  eliminar(productos, numero_productos);
70                  break;
71
72              case BUSCAR:
73                  buscar(productos, numero_productos);
74                  break;
75
76              case FIN:
77                  guardar_datos_en_archivo(componentes, numero_componentes);
78                  cout << "Fin" << endl;
79                  break;
80          }
81      }
82  }
83
84  void leer_producto(producto_almacen &producto){
85      cout << "Código : ";
86      cin >> producto.codigo;
87      cout << "Descripción : ";
88      cin.ignore();
89      getline(cin, producto.descripcion);
90      cout << "Existencias : ";
91      cin >> producto.existencias;
92  }
93
94  void anadir_producto(producto_almacen productos[], int &numero_productos){
95      if (numero_productos > MAX_PRODUCTOS){
96          cout << "Número máximo de productos" << endl;
97      } else {
98          leer_producto(productos[numero_productos]);
99          numero_productos++;
100     }
101 }
102
103 void borrar_producto(producto_almacen productos[], int &numero_productos){
104     int codigo = 0;
105     int posicion = 0;
106     cout << "Introduce un código de un producto que desees eliminar ";
107     cin >> codigo;
108
109
110     posicion = busqueda(productos, codigo, numero_productos);
111     if (posicion != PRODUCTO_NO_ENCONTRADO) {
112         productos[posicion] = productos[numero_productos - 1];
113         numero_productos--;
114         cout << "Producto eliminado" << endl;
115     }
116     else {
117         cout << "Producto no encontrado" << endl;
118     }
119 }
120
121 void mostrar_producto(const producto_almacen &producto){
122     cout << "Codigo: ";
123     cout << producto.codigo << endl;
124     cout << "Descripcion: ";
125     cout << producto.descripcion << endl;
126     cout << "Existencias: ";
127     cout << producto.existencias << endl;
128 }
129
130 void listar_productos(producto_almacen productos[], int numero_productos){
131     for (int i = 0; i < numero_productos; i++){
132         mostrar_producto(productos[i]);
```

```
    }
}

void anadir(producto_almacen productos[], int &numero_productos){
    int codigo = 0;
    int existencias = 0;
    bool encontrado = false;
    int posicion = 0;

    cout << "codigo: ";
    cin >> codigo;
    cout << "Añadir existencias: ";
    cin >> existencias;

    posicion = busqueda(productos, codigo, numero_productos);
    if (posicion == PRODUCTO_NO_ENCONTRADO) {
        cout << "El código introducido no corresponde con ningún producto." << endl;
    } else {
        productos[posicion].existencias =  productos[posicion].existencias + existencias;
        cout << "Se han almacenado (" << existencias << ") de "
             << productos[posicion].descripcion << " con código " << productos[posicion].codigo << endl;
    }
}

void eliminar(producto_almacen productos[], int &numero_productos){
    int codigo = 0;
    int existencias = 0;
    bool encontrado = false;
    int posicion = 0;

    cout << "Codigo: ";
    cin >> codigo;
    cout << "Eliminar existencias: ";
    cin >> existencias;

    posicion = busqueda(productos, codigo, numero_productos);
    if (posicion == PRODUCTO_NO_ENCONTRADO) {
        cout << "El código introducido no corresponde con ningún producto." << endl;
    } else {
        productos[posicion].existencias -= existencias;
        cout << "Se han retirado (" << existencias << ") de "
             << productos[posicion].descripcion << " con código " << productos[posicion].codigo << endl;
    }
}

void buscar(producto_almacen productos[], int &numero_productos){
    int codigo = 0;
    int posicion = 0;
    cout << "Introduce un código que desees buscar ";
    cin >> codigo;

    posicion = busqueda(productos, codigo, numero_productos);
    if(posicion == PRODUCTO_NO_ENCONTRADO){
        cout << "El código introducido no corresponde a ningún producto" << endl;
    } else {
        mostrar_producto(productos[posicion]);
    }
}

int busqueda(const producto_almacen productos[], int codigo, int numero_productos){
    for (int i = 0; i < numero_productos; i++){
        if(codigo == productos[i].codigo){
            return i;
        }
    }
    return PRODUCTO_NO_ENCONTRADO;
```

```
}

void cargar_datos_desde_archivo(componente componentes[], int &numero_componentes){
    ifstream archivo(NOMBRE_ARCHIVO);

    if (archivo.is_open()){
        while (!archivo.eof() && numero_componentes < MAX_COMPONENTES){
            archivo >> componentes[numero_componentes].codigo;
            archivo.ignore(); // Ignorar el espacio después del código
            getline(archivo, componentes[numero_componentes].descripcion);
            archivo >> componentes[numero_componentes].existencias;
            numero_componentes++;
        }

        archivo.close();
    } else {
        cout << "No se pudo abrir el archivo para cargar datos." << endl;
    }
}

void guardar_datos_en_archivo(componente componentes[], int numero_componentes){
    ofstream archivo(NOMBRE_ARCHIVO);

    if (archivo.is_open()){
        for (int i = 0; i < numero_componentes; i++){
            archivo << componentes[i].codigo << " " << componentes[i].descripcion << " "
                    << componentes[i].existencias << endl;
        }

        archivo.close();
    } else {
        cout << "No se pudo abrir el archivo para guardar datos." << endl;
    }
}
```