

```

1  /*
2   * Pablo_Villa 874773
3   * 01/12/23
4   */
5
6  #include <iostream>
7  using namespace std;
8
9  const int MAX_PRODUCTOS = 100;
10 const int PRODUCTO_NO_ENCONTRADO = -1;
11 const int ALTA = 1;
12 const int BAJA = 2;
13 const int LISTADO = 3;
14 const int ENTRADA = 4;
15 const int SALIDA = 5;
16 const int BUSCAR = 6;
17 const int FIN = 7;
18
19 struct producto_almacen {
20     int codigo = 0;
21     string descripcion = "";
22     int existencias = 0;
23 };
24
25 void anadir_producto(producto_almacen productos[], int &numero_productos);
26 void leer_producto(producto_almacen &producto);
27 void listar_productos(producto_almacen productos[], int numero_productos);
28 void anadir(producto_almacen productos[], int &numero_productos);
29 void eliminar(producto_almacen productos[], int &numero_productos);
30 void borrar_producto(producto_almacen productos[], int &numero_productos);
31 void buscar(producto_almacen productos[], int &numero_productos);
32 int busqueda(const producto_almacen productos[], int codigo, int numero_productos);
33 void cargar_datos_desde_archivo(componente componentes[], int &numero_componentes);
34 void guardar_datos_en_archivo(componente componentes[], int numero_componentes);
35
36 const string NOMBRE_ARCHIVO = "datos_componentes.txt";
37
38 int main(){
39     producto_almacen productos[MAX_PRODUCTOS];
40     int opcion = 0;
41     int numero_productos = 0;
42
43     cargar_datos_desde_archivo(componentes, numero_componentes);
44
45     while(opcion != FIN){
46         cout << ALTA << "->Alta " << BAJA << "->Baja " << LISTADO << "->Listado " << ENTRADA
47             << "->Entrada " << SALIDA << "->Salida " << BUSCAR << "-> Buscar " << FIN << "->Acabar: ";
48
49         cin >> opcion;
50
51         switch(opcion){
52             case ALTA:
53                 anadir_producto(productos, numero_productos);
54                 break;
55
56             case BAJA:
57                 borrar_producto(productos, numero_productos);
58                 break;
59
60             case LISTADO:
61                 listar_productos(productos, numero_productos);
62                 break;
63
64             case ENTRADA:
65                 anadir(productos, numero_productos);
66                 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]);

```

```

133     }
134 }
135
136 void anadir(producto_almacen productos[], int &numero_productos){
137     int codigo = 0;
138     int existencias = 0;
139     bool encontrado = false;
140     int posicion = 0;
141
142     cout << "codigo: ";
143     cin >> codigo;
144     cout << "Añadir existencias: ";
145     cin >> existencias;
146
147     posicion = busqueda(productos, codigo, numero_productos);
148     if (posicion == PRODUCTO_NO_ENCONTRADO) {
149         cout << "El código introducido no corresponde con ningún producto." << endl;
150     } else {
151         productos[posicion].existencias = productos[posicion].existencias + existencias;
152         cout << "Se han almacenado (" << existencias << ") de "
153             << productos[posicion].descripcion << " con código " << productos[posicion].codigo << endl;
154     }
155 }
156
157 void eliminar(producto_almacen productos[], int &numero_productos){
158     int codigo = 0;
159     int existencias = 0;
160     bool encontrado = false;
161     int posicion = 0;
162
163     cout << "Codigo: ";
164     cin >> codigo;
165     cout << "Eliminar existencias: ";
166     cin >> existencias;
167
168     posicion = busqueda(productos, codigo, numero_productos);
169     if (posicion == PRODUCTO_NO_ENCONTRADO) {
170         cout << "El código introducido no corresponde con ningún producto." << endl;
171     } else {
172         productos[posicion].existencias -= existencias;
173         cout << "Se han retirado (" << existencias << ") de "
174             << productos[posicion].descripcion << " con código " << productos[posicion].codigo << endl;
175     }
176 }
177
178 void buscar(producto_almacen productos[], int &numero_productos){
179     int codigo = 0;
180     int posicion = 0;
181     cout << "Introduce un código que desees buscar ";
182     cin >> codigo;
183
184     posicion = busqueda(productos, codigo, numero_productos);
185     if(posicion == PRODUCTO_NO_ENCONTRADO){
186         cout << "El código introducido no corresponde a ningún producto" << endl;
187     } else {
188         mostrar_producto(productos[posicion]);
189     }
190 }
191
192 int busqueda(const producto_almacen productos[], int codigo, int numero_productos){
193     for (int i = 0; i < numero_productos; i++){
194         if(codigo == productos[i].codigo){
195             return i;
196         }
197     }
198     return PRODUCTO_NO_ENCONTRADO;

```

```

199 }
200
201 void cargar_datos_desde_archivo(componente componentes[], int &numero_componentes){
202     ifstream archivo(NOMBRE_ARCHIVO);
203
204     if (archivo.is_open()){
205         while (!archivo.eof() && numero_componentes < MAX_COMPONENTES){
206             archivo >> componentes[numero_componentes].codigo;
207             archivo.ignore(); // Ignorar el espacio después del código
208             getline(archivo, componentes[numero_componentes].descripcion);
209             archivo >> componentes[numero_componentes].existencias;
210             numero_componentes++;
211         }
212
213         archivo.close();
214     } else {
215         cout << "No se pudo abrir el archivo para cargar datos." << endl;
216     }
217 }
218
219 void guardar_datos_en_archivo(componente componentes[], int numero_componentes){
220     ofstream archivo(NOMBRE_ARCHIVO);
221
222     if (archivo.is_open()){
223         for (int i = 0; i < numero_componentes; i++){
224             archivo << componentes[i].codigo << " " << componentes[i].descripcion << " "
225                 << componentes[i].existencias << endl;
226         }
227
228         archivo.close();
229     } else {
230         cout << "No se pudo abrir el archivo para guardar datos." << endl;
231     }
232 }

```