



Estructura de Datos

Docente: Carlos Enrique Del Carpio Rojas

NRC: 70715

Sistema de gestion de tienda de abarrotes “nombre”

INTEGRANTES:

- Llano Coaguila, Anthony Bryan
- Ccoropuna Alejandro, Celso Gustavo
- Jara Machicao, Alim Erasmo
- Cueva Alanguia, Cristian
- Jose Carlos Rivero Mamani
- Leovigildo Domingo Caya Umiyauri

**PERÚ
2025**

CAPÍTULO 1: Análisis del Problema

1. Descripción del problema

El presente proyecto tiene como propósito desarrollar un sistema que permita gestionar de manera eficiente los productos de una tienda de abarrotes.

Se busca implementar un sistema sencillo en C++ que ayude a monitorear la información de los productos, facilitando el control de su nombre, precio y cantidad disponible. Este sistema permitirá realizar operaciones básicas como agregar, mostrar y eliminar productos, además de contar con un menú de navegación que mejore la interacción con el usuario.

El sistema está dirigido a la propia empresa y será utilizado por el personal encargado del área de almacén o administración, buscando optimizar sus tareas diarias mediante una herramienta confiable y fácil de usar.

2. Requerimientos del sistema

· Funcionales:

- Agregar producto: registrar un nuevo producto con sus respectivos datos (nombre, precio y cantidad).
- Mostrar productos: visualizar la lista de productos registrados con su información completa.
- Eliminar producto: quitar un producto del inventario.
- Menú principal: ofrecer opciones claras para acceder a las funciones del sistema.

· No funcionales:

- Los datos se almacenan de forma temporal (en memoria), no persistente.

3. Estructuras de datos propuestas

Se emplean estructuras tipo struct para definir las entidades principales del sistema:

Usuario: Contiene los campos nombre, apellido, edad y un puntero siguiente para enlazar varios usuarios.

```
struct usuario {  
    string nombre;  
    string apellido;  
    int edad;  
    usuario* siguiente;  
};
```

Producto: Contiene los campos nombre, precio, cantidad y un puntero siguiente para formar una lista enlazada de productos.

```
struct productos {  
    string nom_producto;  
    double precio_producto;  
    int cantidad_producto;  
    productos* siguiente_pro;  
};
```

Ambas estructuras forman listas enlazadas simples, donde cada nodo apunta al siguiente elemento. Esto permite agregar, recorrer y eliminar elementos dinámicamente sin usar arreglos de tamaño fijo.

4. Justificación de la elección

Las listas enlazadas fueron seleccionadas porque:

Permiten almacenar un número variable de elementos sin definir previamente un tamaño máximo.

Facilitan la inserción y eliminación dinámica de nodos sin necesidad de desplazar los elementos.

Permiten recorrer los registros fácilmente para mostrarlos o realizar búsquedas simples.

Capítulo 2: Diseño de la Solución

1. Descripción de estructuras de datos y operaciones:

Estructuras de datos:

usuario → para registrar la información de quienes utilizan el sistema.

productos → para registrar, mostrar y eliminar productos.

Operaciones principales:

agregarProducto() → agrega un nuevo nodo al final de la lista de productos.

eliminar() → elimina un producto específico por su nombre.

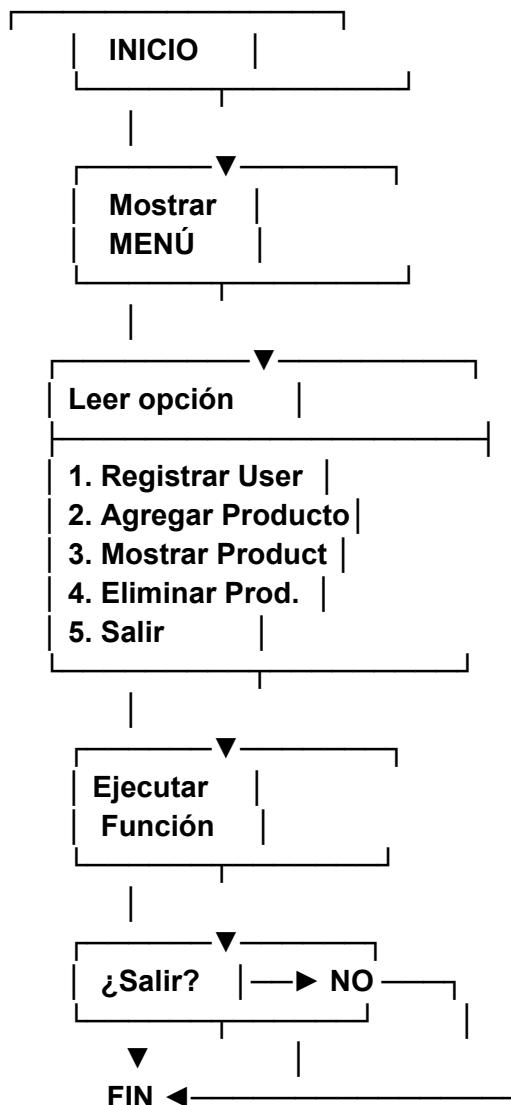
Menu() → muestra las opciones del sistema.

Uso de punteros (* y &) para recorrer, agregar o eliminar nodos dinámicamente

2. Algoritmos principales:

- Pseudocódigo para agregar proceso.
- Pseudocódigo para cambiar el estado del proceso.

3. Diagramas de Flujo



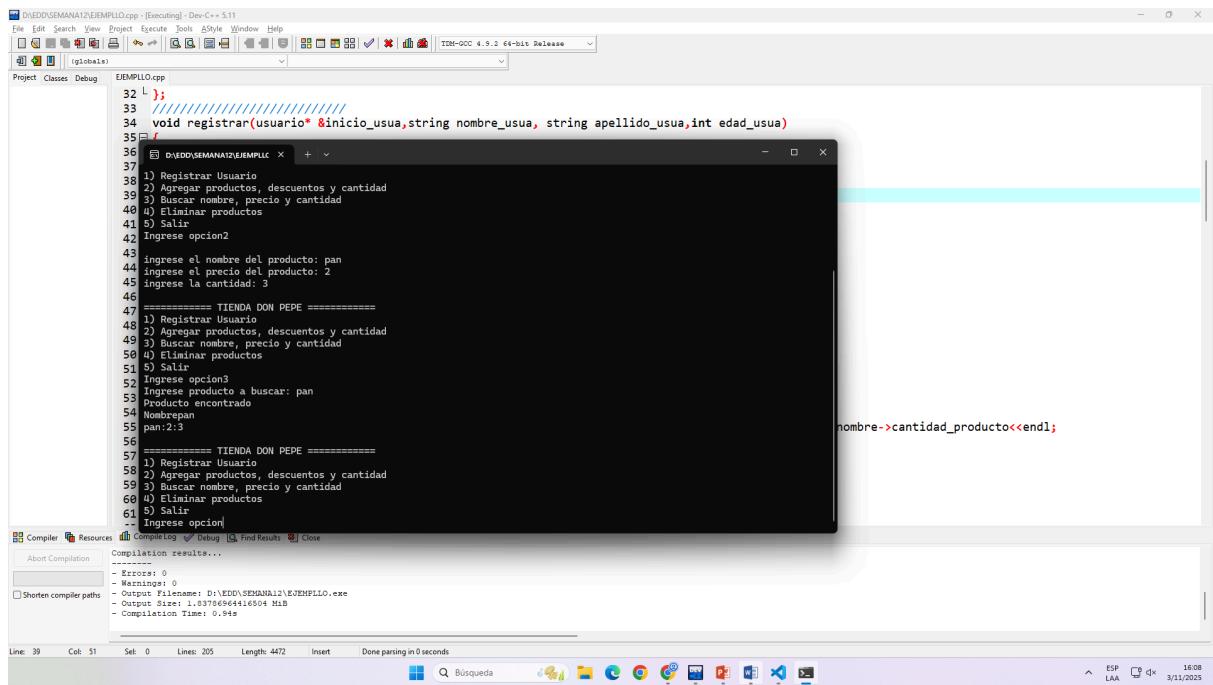
4. Justificación del diseño:

(Ventajas, eficiencia, etc.)

Capítulo 3: Solución Final

1. Código limpio, bien comentado y estructurado.

2. Capturas de pantalla de las ventanas de ejecución con las diversas pruebas de validación de datos

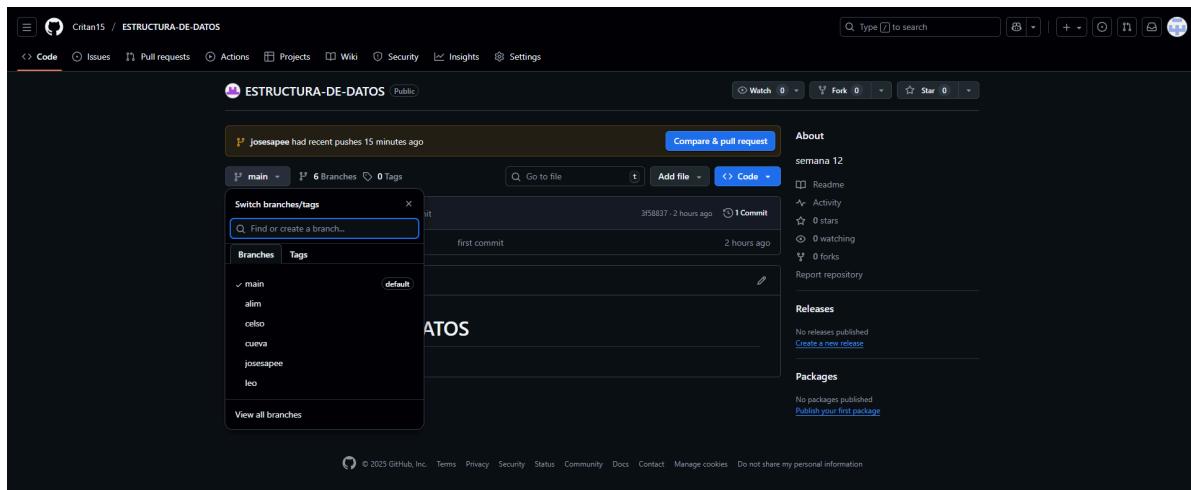


The screenshot shows the Dev-C++ IDE interface. The main window displays the source code of a C++ program named EJEMPLO1.cpp. The code includes comments for a menu system and a section for a store named 'TIENDA DON PEPE'. The output window shows the execution of the program, displaying menu options and user inputs like 'pan' and '2'. Below the main window, the compiler results show 0 errors and 0 warnings, with the output file being D:\EDO\SEMANA12\EJEMPLO1.exe. The status bar at the bottom provides information about the current line (Line: 39), column (Col: 51), and other development details.

3. Manual de usuario

Capítulo 4: Evidencias de Trabajo en Equipo

1. Repositorio con Control de Versiones (Capturas de Pantalla)



The screenshot shows a GitHub repository page for 'ESTRUCTURA-DE-DATOS'. The repository has 6 branches and 0 tags. A recent push from 'josesapee' is visible. The main branch is selected. The repository has 1 commit and 2 hours ago. The commit message is 'first commit'. On the right side, there's an 'About' section for 'semana 12' which includes a 'Readme' link, activity stats (0 stars, 0 forks), and release and package sections. The bottom of the page includes standard GitHub navigation links and a copyright notice.

- Registro de commits claros y significativos que evidencian aportes individuales (proactividad).
- Historial de ramas y fusiones si es aplicable.
- Evidencia por cada integrante del equipo.

- Enlace a la herramienta colaborativa

Commits	
 alim	 All users  All time
-o Commits on Nov 3, 2025	
Punto 3	 be3181b  
Alim930 committed 13 minutes ago	
Menu-Inicio	 4bec17e  
Alim930 committed 48 minutes ago	
Cambio menu	 129c0fe  
Alim930 committed 49 minutes ago	
Menu-Inicio	 1115d03  
Alim930 committed 50 minutes ago	
first commit	 3f58837  
EstudianteUC authored and EstudianteUC committed 1 hour ago	

2. Plan de Trabajo y Roles Asignados

- Documento inicial donde se asignan tareas y responsabilidades.
 - Llano Coaguila, Anthony Bryan -Avance del informe
 - Ccoropuna Alejandro, Celso Gustavo -Avance del informe
 - Jara Machicao, Alim Erasmo -Funciones menú y structs
 - Cueva Alanguia, Cristian -Funciones agregar, eliminar
 - Jose Carlos Rivero Mamani -Función mostrar todos los productos
 - Leovigildo Domingo Caya Umiyauri -Función Buscar

Cronograma con fechas límite para cada entrega parcial.

Reunion: 3:20pm

Aprender GitHub: 3:30pm

Primer avance: 3:40pm

Segundo avance3:50pm

Entrega final: 4pm

Entrega FINAL FINAL: 4:30pm

- Registro de reuniones o comunicación del equipo (Actas de reuniones.).

```
File Edit Search View Project Execute Tools ASide Window Help
D:\EDD\SEMANA12\EJEMPLO.cpp - DevC++ 5.11
Project Classes Debug EJEMPLO.cpp
136     cout<<"\n===== TIENDA DON PEPE ======\n";
137     cout<<"1) Registrar Usuario\n";
138     cout<<"2) Agregar productos, precios y cantidad\n";
139     cout<<"3) Buscar nombre, precio y cantidad\n";
140     cout<<"4) Eliminar productos\n";
141     cout<<"5) Mostrar todos los productos\n";
142     cout<<"6) Salir\n";
143     cout<<"Ingrese opcion: ";
144 }
145
146 int main()
147 {
148     usuario* inicio_usua=NULL;
149     productos* inicio_pro=NULL;
150
151     string nombre_pro,nombre_usua,apellido_usua, nombre_pro_elim;
152
153     double precio_pro;
154     int cantidad_pro,edad_usua;
155
156     int opcion;
157
158     do{
159         Menu();
160         cin >> opcion;
161         cin.ignore(1024,'\'\n\');
162
163         switch (opcion){
164             case 1:
165
166             break;
167
168             case 2:
169
170             break;
171
172             case 3:
173
174             break;
175
176             case 4:
177
178             break;
179
180             case 5:
181
182             break;
183
184             case 6:
185
186             break;
187
188             default:
189                 cout<<"Opcion incorrecta\n";
190
191         }
192     }while(opcion!=6);
193
194     cout<<"Gracias por usar el sistema\n";
195 }
```

Compiler (2) Resources Compile Log Debug Find Results Close

Line: 148 Col: 26 Sel: 0 Lines: 222 Length: 4054 Insert Done parsing in 0.016 seconds

Message

In function 'int main()':

[Error] NULL was not declared in this scope

ESP LAA 16:18 3/11/2025