Prueba de Caja Blanca

"Implementación de un sistema e inventario para el local Chaskibots"

Integrantes:
Cobeña Bravo Davis Ariel
German Luna Jhon Alexis
Ramos Vilca Paulo Alejandro

Fecha Enero 2025

Historia de revisión

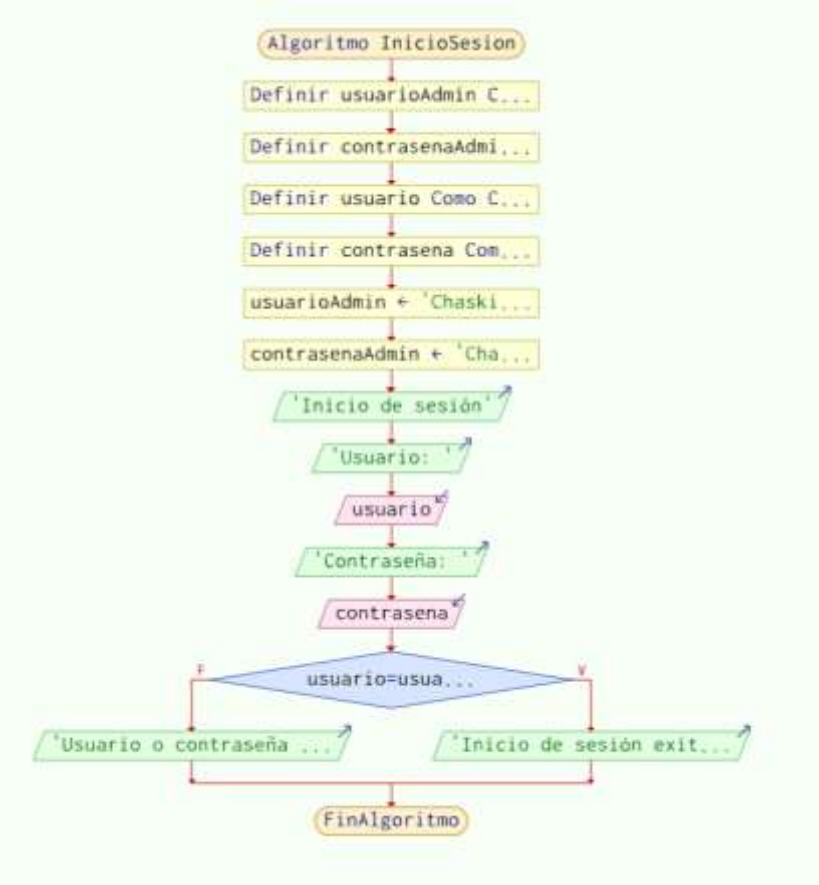
Fecha	Versión	Descripción	Autores
16 – 01 – 2025	1	Versión inicial	Cobeña Davis Jhon German Ramos Paulo
22 – 01 – 2025	2	Realización pruebas caja blanca hasta REQ005	Cobeña Davis Jhon German Ramos Paulo
22 – 01 – 2025	3	Corrección caja blanca	Cobeña Davis Jhon German Ramos Paulo
11 – 02 – 2025	4	Corrección caja blanca	Cobeña Davis Jhon German Ramos Paulo
12 – 02 – 2025	5	Última versión caja blanca	Cobeña Davis Jhon German Ramos Paulo

Prueba caja blanca de Inicio de sesion

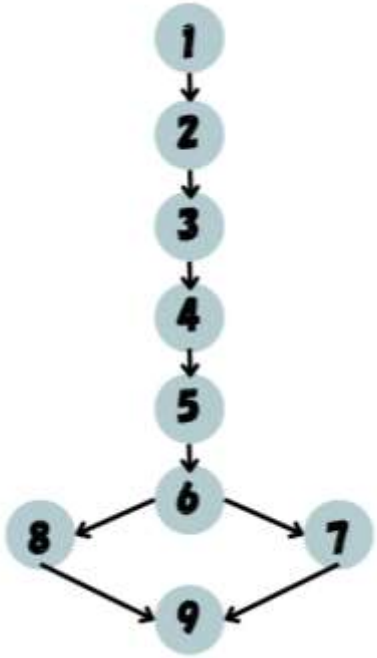
1. CÓDIGO FUENTE

```c
#include <stdio.h>
#include <string.h>

int main() {
    // Usuario y contraseña predefinidos
    char usuarioAdmin[]= "ChaskiBots1";
    char contrasenaAdmin[] = "ChaskiBots1";

    // Variables para ingresar datos
    char usuario[20];
    char contrasena[20];

    printf("Inicio de sesión\n");
    printf("Usuario: ");
    scanf("%s", usuario);
    printf("Contraseña: ");
    scanf("%s", contrasena);

    // Verificar credenciales
    if (strcmp(usuario, usuarioAdmin) == 0 && strcmp(contrasena, contrasenaAdmin) == 0) {
        printf("Inicio de sesión exitoso.\n");
    } else {
        printf("Usuario o contraseña incorrectos.\n");
    }
}
```

2. DIAGRAMA DE FLUJO (DF)



3. GRAFO DE FLUJO (GF)



4. IDENTIFIACCIÒN DE LAS RUTAS (Camino basico)

RUTAS

R1: 1, 2, 3, 4, 5, 6, 7, 9
R2: 1, 2, 3, 4, 5, 6, 8, 9

5. COMPLEJIDAD CICLOMÁTICA

Se puede calcular de las siguientes formas:

- V(G) = número de nodos predicados(decisiones)+1
  V(G)= 1 + 1 = 2

- V(G) = A – N + 2
  V(G)= 9 – 9 + 2 =2

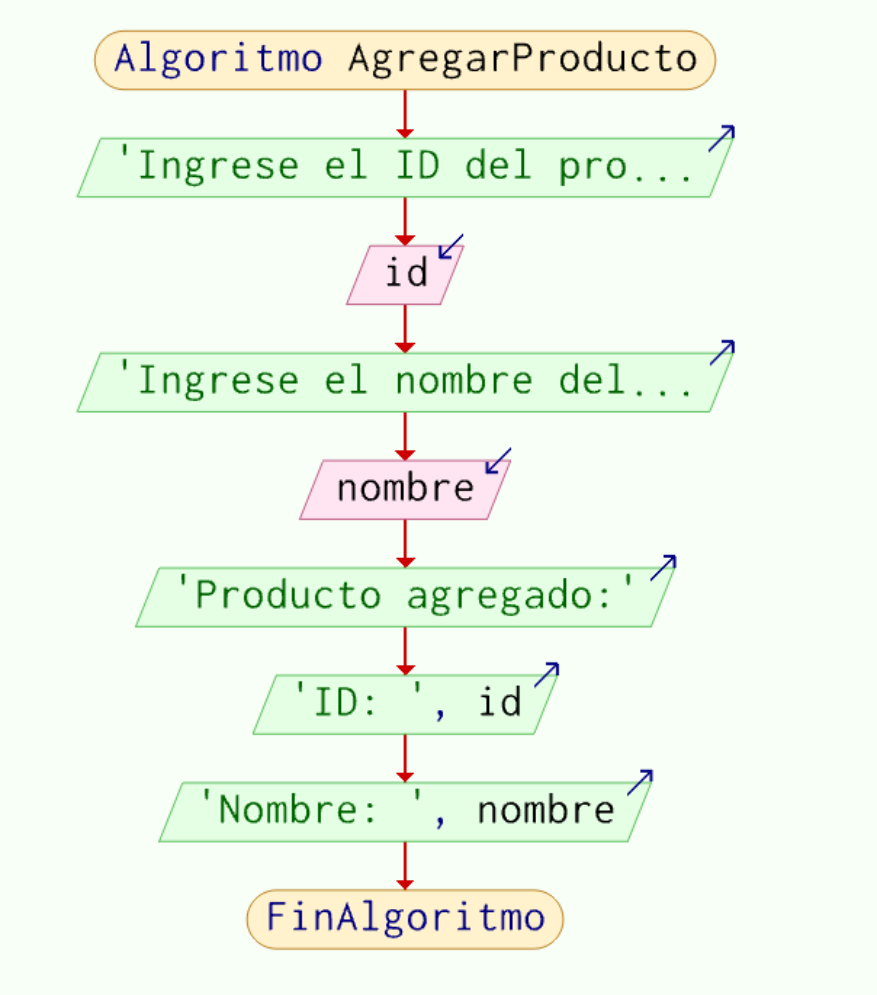
DONDE:
P: 1
A: 9
N: 9

Prueba caja blanca de Añadir un producto

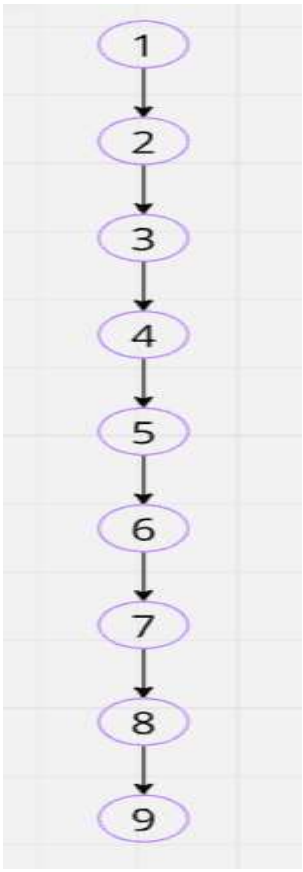
1. **CÓDIGO FUENTE**

```c
#include <stdio.h>
#include <string.h>
int main() {
    int id;
    char nombre[50];

    // Pedir al usuario que ingrese el nombre y el ID del producto
    printf("Ingrese el ID del producto: ");
    scanf("%d", &id);

    printf("Ingrese el nombre del producto: ");
    scanf("%s", &nombre);
    // Mostrar los datos ingresados
    printf("\nProducto agregado:\n");
    printf("ID: %d\n", id);
    printf("Nombre: %s\n", nombre);
    return 0;
}
```

2. **DIAGRAMA DE FLUJO (DF)**



3. Grafo de flujo (GF)



4. IDENTIFIACCIÒN DE LAS RUTAS (Camino basico)

RUTAS

R1: 1,2,3,4,5,6,7,8,9

5. COMPLEJIDAD CICLOMÁTICA

Se puede calcular de las siguientes formas:

- V(G) = número de nodos predicados(decisiones)+1
  V(G)= 0 + 1 = 1

- V(G) = 8 – 9 + 2
  V(G)= -1 + 2 = 1

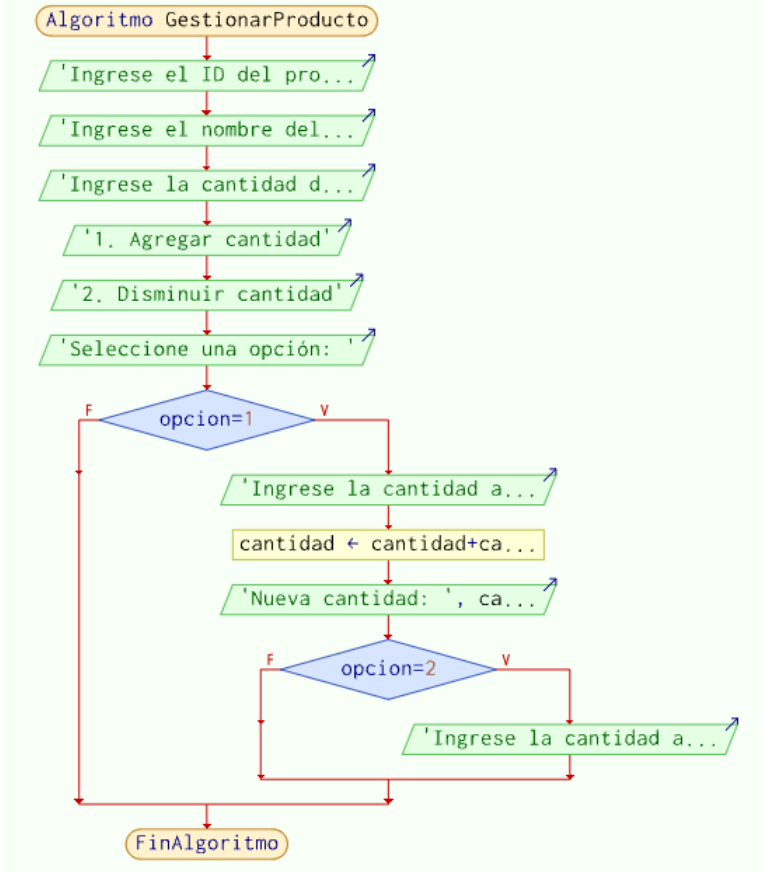
DONDE:
P: 0
A: 8
N: 9

Prueba caja blanca de Modificar un producto

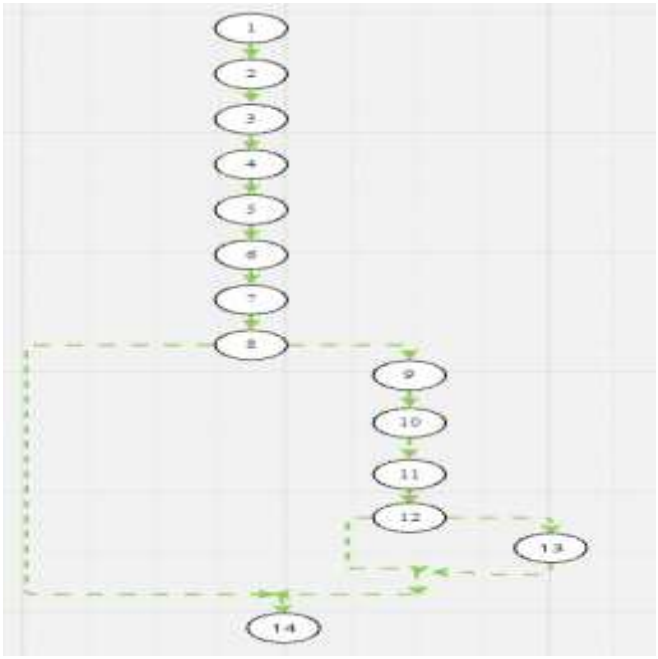
1. **CÓDIGO FUENTE**

```c
#include <stdio.h>
#include <string.h>
int main() {
    int id;
    char nombre[50];
    int cantidad;
    printf("Ingrese el ID del producto: ");
    scanf("%d", &id);
    printf("Ingrese el nombre del producto: ");
    scanf("%s", &nombre);
    printf("Ingrese la cantidad del producto: ");
    scanf("%d", &cantidad);
    int opcion;
    do {
        printf("\n¿Desea agregar o disminuir la cantidad?\n");
        printf("1. Agregar cantidad\n");
        printf("2. Disminuir cantidad\n");
        printf("3. Salir\n");
        printf("Seleccione una opción: ");
        scanf("%d", &opcion);
        if (opcion == 1) {
            printf("Ingrese la cantidad a agregar: ");
            scanf("%d", &cantidad);
            cantidad += cantidad;
            printf("Nueva cantidad: %d\n", cantidad);
        } else if (opcion == 2) {
            printf("Ingrese la cantidad a disminuir: ");
            scanf("%d", &cantidad);
            if (cantidad <= cantidad) {
                cantidad -= cantidad;
                printf("Nueva cantidad: %d\n",cantidad);
            } else {
                printf("No puedes reducir la cantidad a un valor negativo.\n");
            }
        } else if (opcion == 3) {
            printf("Saliendo...\n");
        } else {
            printf("Opción inválida. Intenta nuevamente.\n");
        }
    } while(opcion != 3);
    return 0;
}
```

2. **DIAGRAMA DE FLUJO (DF)**



3. Grafo de flujo (GF)



4. IDENTIFIACCIÒN DE LAS RUTAS (Camino basico)

RUTAS

R1: 1, 2, 3, 4, 5, 6, 7, 8, 14
R2: 1, 2, 3, 4, 5, 6, 8, 9, 10, 11, 12 ,14
R3: 1, 2, 3, 4, 5, 6, 8, 9, 10, 11, 12 ,13, 14

5. COMPLEJIDAD CICLOMÁTICA

Se puede calcular de las siguientes formas:

- V(G) = número de nodos predicados(decisiones)+1
  V(G)= 2 + 1 = 3

- V(G) = A – N + 2
  V(G)= 15 – 14 + 2 =3

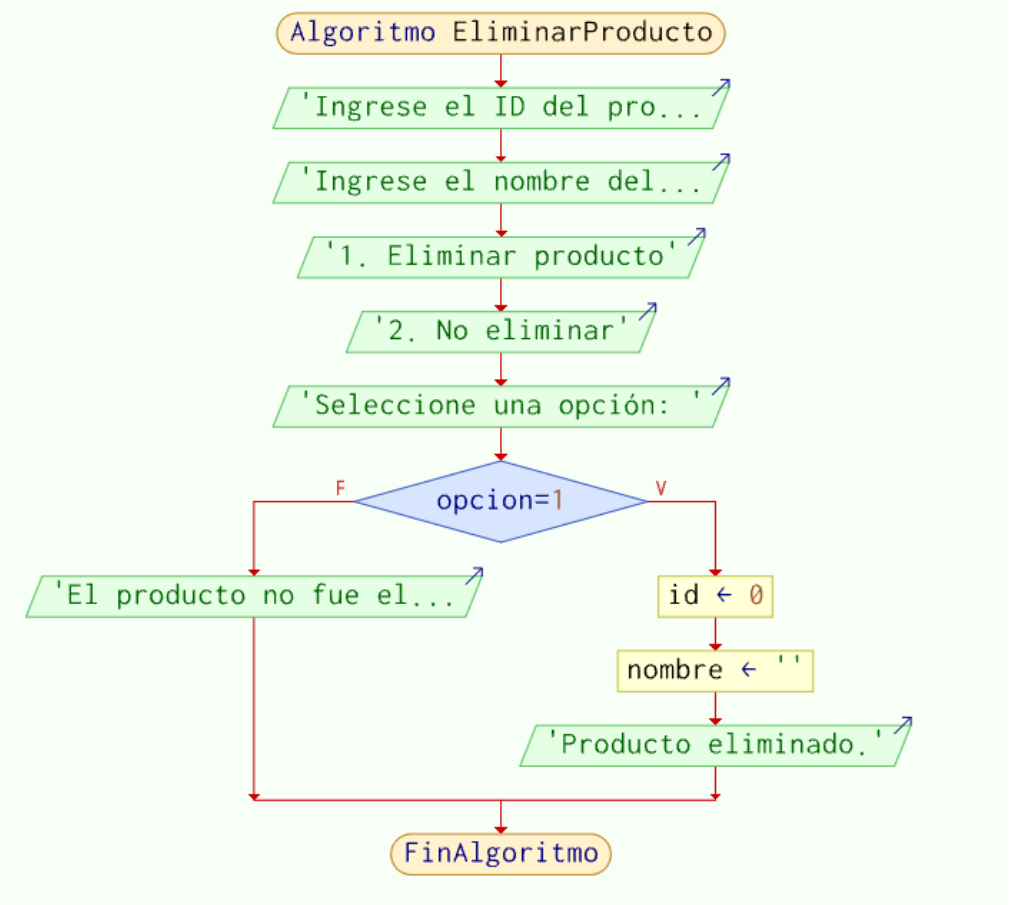
  DONDE:
  P: 2
  A: 15
  N: 14

Prueba caja blanca de Eliminar un producto

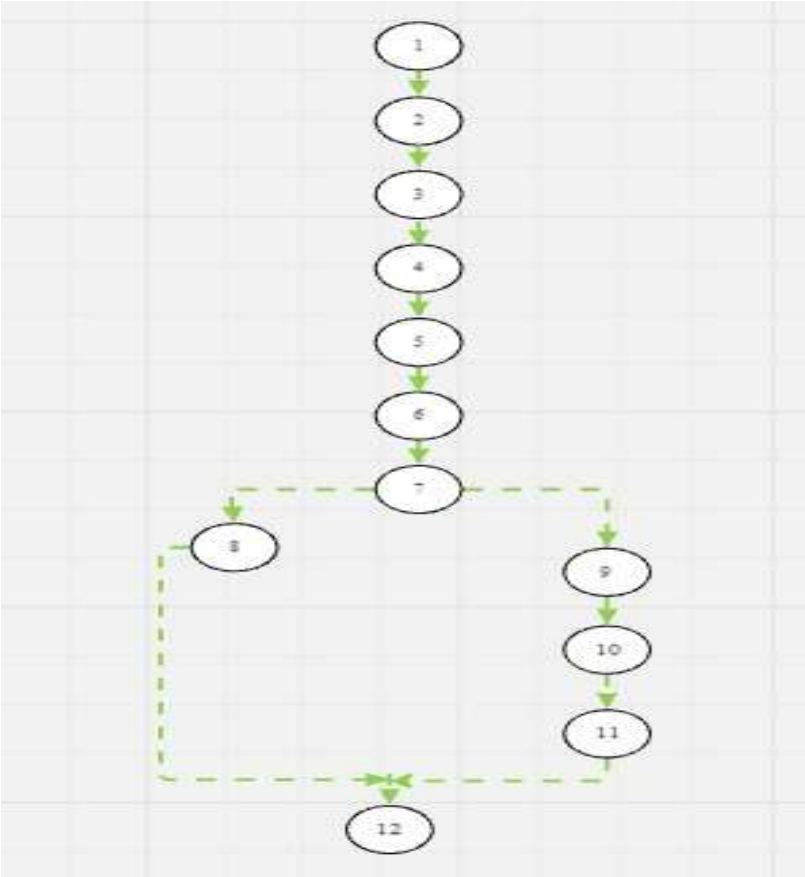
1. **CÓDIGO FUENTE**

```c
#include <stdio.h>
#include <string.h>
int main() {
    int id;
    char nombre[50];
    int opcion;
    printf("Ingrese el ID del producto: ");
    scanf("%d", &id);
    printf("Ingrese el nombre del producto: ");
    scanf ("%s", &nombre);
    printf("1. Eliminar producto\n");
    printf("2. No eliminar\n");
    printf("Seleccione una opción: ");
    scanf("%d", &opcion);
    if (opcion == 1) {
        // Eliminamos el producto (borramos los datos)
        id = 0;
        strcpy(nombre, "");
        printf("\nProducto eliminado.\n");
    } else {
        printf("\nEl producto no fue eliminado.\n");
    }
    return 0;
}
```

2. **DIAGRAMA DE FLUJO (DF)**



3. Grafo de flujo (GF)



4. IDENTIFIACCIÒN DE LAS RUTAS (Camino basico)

RUTAS

R1: 1, 2, 3, 4, 5, 6, 7, 8, 12
R2: 1, 2, 3, 4, 5, 6, 7, 9, 10, 11, 12

5. COMPLEJIDAD CICLOMÁTICA

Se puede calcular de las siguientes formas:

- V(G) = número de nodos predicados(decisiones)+1
  V(G)= 1 + 1 = 2

- V(G) = A – N + 2
  V(G)= 12 – 12 + 2 = 2

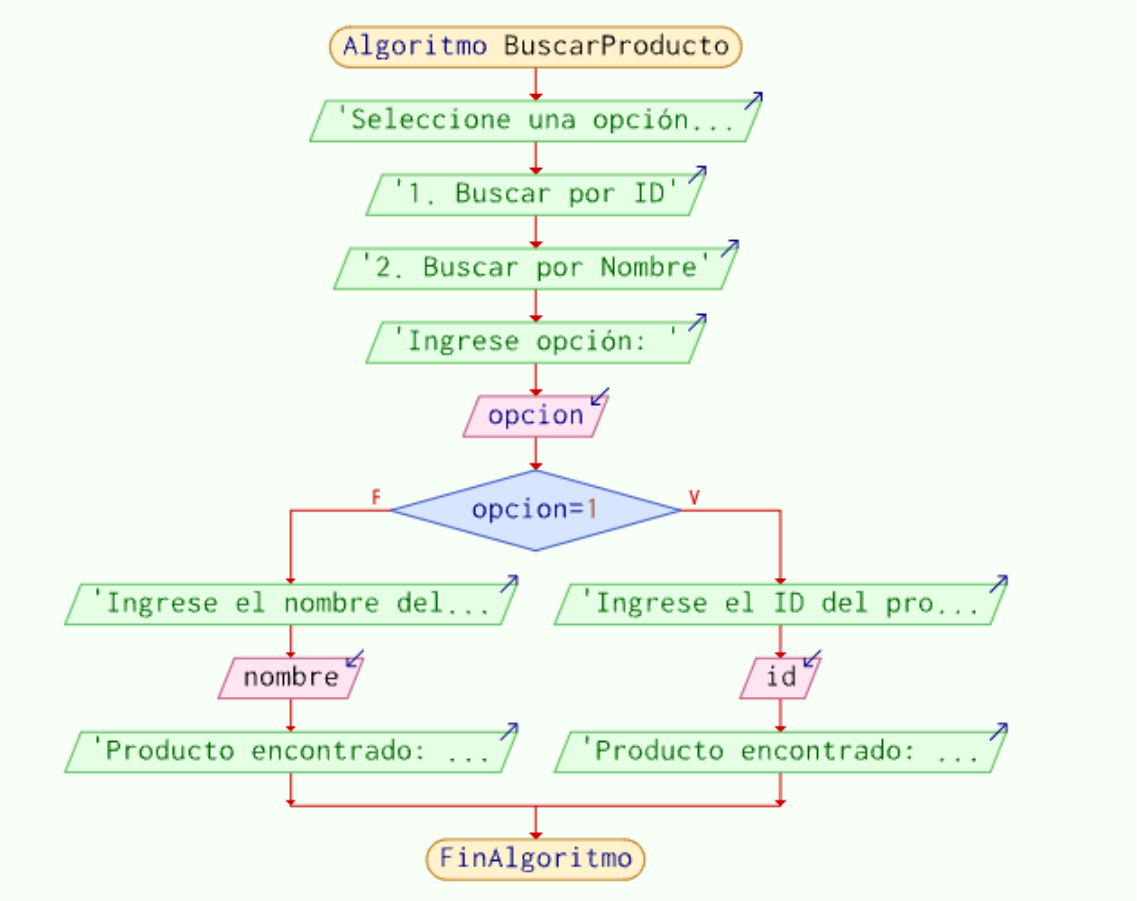
  DONDE:
  P: 1
  A: 12
  N: 12

Prueba caja blanca de Buscar un producto

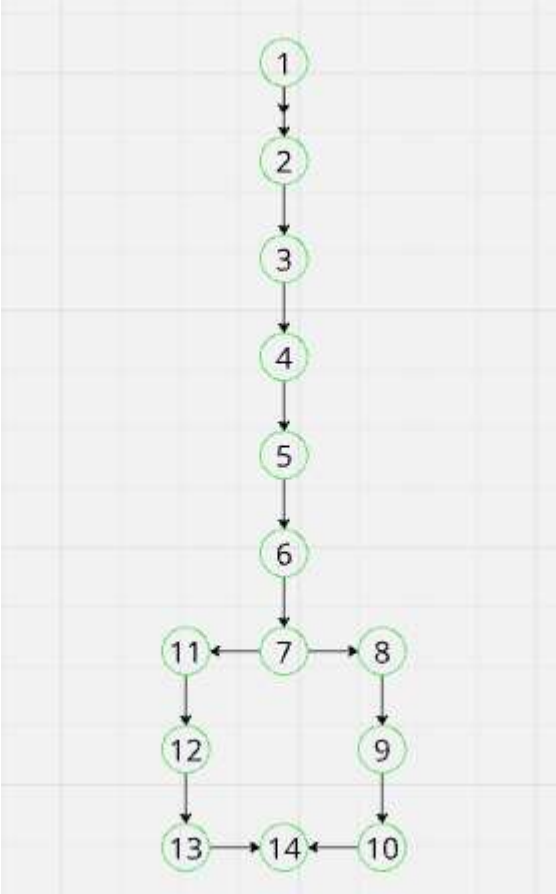
1. **CÓDIGO FUENTE**

```c
int main() {
    int n;
    struct Producto productos[n];
    int opcion, id;
    char nombre[50];
    printf("\nSeleccione una opción de búsqueda:\n");
    printf("1. Buscar por ID\n");
    printf("2. Buscar por Nombre\n");
    printf("Ingrese opción: ");
    scanf("%d", &opcion);
    if (opcion == 1) {
        printf("Ingrese el ID del producto: ");
        scanf("%d", &id);
        buscar_por_id(productos, n, id);
    } else if (opcion == 2) {
        printf("Ingrese el nombre del producto: ");
        scanf(" %[^\n]s", nombre);  // Para leer el nombre con espacios
        buscar_por_nombre(productos, n, nombre);
    } else {
        printf("Opción no válida.\n");
    }
    return 0;
}
```

2. **DIAGRAMA DE FLUJO (DF)**



3. **Grafo de flujo (GF)**



4. **IDENTIFIACCIÒN DE LAS RUTAS (Camino basico)**

RUTAS

R1: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 14
R2: 1, 2, 3, 4, 5, 6, 7, 11, 12, 13, 14

5. **COMPLEJIDAD CICLOMÁTICA**

Se puede calcular de las siguientes formas:

- V(G) = número de nodos predicados(decisiones)+1
  V(G)= 1 + 1 = 2

- V(G) = N – A + 2
  V(G)= 14 – 14 + 2 = 2

  DONDE:
  P: 4
  A: 14
  N: 14