Objetivo: Desarrollar un programa en V que permita gestionar una lista de compras en un supermercado, simulando la facturación y actualización de stock.

Descripción del Problema:

Dada la siguiente lista de compras, donde cada producto tiene un nombre y una cantidad deseada.

Lista de Compras

Nombre	Cantidad
Arroz	5
Aceite	10
Leche	4
Manteca	1
Fideos	3
Manzana	2

El programa deberá leer esta lista y compararla con un archivo binario previamente cargado, que contiene información sobre los productos disponibles en el supermercado. Este archivo tiene el siguiente formato:

Archivo de Productos:

Codigo	Nombre	Cantidad	Precio
1102	Leche	150	500
1103	Manteca	50	300
1104	Polenta	230	400
1105	Vino	340	500
1106	Fideos	200	600
1107	Aceite	0	400
1108	Arroz	345	300
1109	Yogur	60	400

```
Producto productos[] = {
    {1102, "Leche", 150, 500},
    {1103, "Manteca", 50, 300},
    {1104, "Polenta", 230, 400},
    {1105, "Vino", 340, 500},
    {1106, "Fideos", 200, 600},
    {1107, "Aceite", 0, 400},
    {1108, "Arroz", 345, 300},
    {1109, "Yogur", 60, 400}
  };
```

Requisitos del Programa:

1. **Comparación y Compra:**
    - El programa debe recorrer la lista de compras y verificar si los productos están disponibles en el archivo de productos.
    - Si el producto está disponible y hay suficiente stock, debe restar la cantidad comprada del stock y calcular el costo de la compra para ese producto.
    - Si el producto no está disponible o no hay suficiente stock, debe indicarse que no fue posible realizar la compra de ese producto.
2. **Actualización de Stock:**
    - Después de procesar toda la lista de compras, el programa debe actualizar el archivo binario con los nuevos valores de stock.
3. **Salida del Programa:**
    - El programa debe generar un resumen de la compra, indicando:
        - Los productos que se pudieron comprar.
        - El costo total de la compra.
        - Los productos que no pudieron comprarse, ya sea por falta de stock o porque no están disponibles en el supermercado.

Consideraciones:

- La lista de compras puede contener productos que no están en el archivo de productos del supermercado. En este caso, deben ser ignorados.
- El archivo binario debe ser leído y actualizado correctamente para reflejar los cambios en el stock después de la compra.
- Se espera que los estudiantes implementen la manipulación de listas enlazadas para gestionar la lista de compras y el archivo binario.

Estructuras del programa

Esta me servirá, para guardar el stock de productos

```
typedef struct {
   int codigo;
   char nombre[30];
   int cantidad;
   int precio;
} Producto;
```

Esta me servirá para armar la lista de compra

```
typedef struct nodo {
   char nombre[30];
   int cantidad;
   struct nodo *sig;
} Nodo;
```

Menu de Opciones

```
int menu(void) {
   int opcion;
   printf("Menu de Opciones:\n");
   printf("1. Cargar productos en el archivo\n");
   printf("2. Cargar lista de compras\n");
   printf("3. Ver lista de compras\n");
   printf("4. Ver stock de productos\n");
   printf("5. Realizar compra\n");
   printf("6. Ver productos que no se pudieron comprar\n");
   printf("7. Salir\n");
   printf("Seleccione una opción: ");
   scanf("%d", &opcion);
   return opcion;
}
```

Ahora la función que carga todos los productos directamente en el archivo

```
void cargarProductos(FILE *archivo) {
   Producto productos[] = {
      {1102, "Leche", 150, 500},
      {1103, "Manteca", 50, 300},
      {1104, "Polenta", 230, 400},
      {1105, "Vino", 340, 500},
      {1106, "Fideos", 200, 600},
      {1107, "Aceite", 0, 400},
      {1108, "Arroz", 345, 300},
      {1109, "Yogur", 60, 400}
   };
   int cantidad = sizeof(productos) / sizeof(productos[0]);

   fwrite(productos, sizeof(Producto), cantidad, archivo);
   printf("Productos cargados en el archivo con éxito.\n");
}
```

Ahora el informe del archivo por pantalla

```
void verStock(FILE *archivo) {
   Producto producto;
   rewind(archivo);
   printf("Stock de Productos:\n");
   printf("Codigo\tNombre\t\tCantidad\tPrecio\n");
   while (fread(&producto, sizeof(Producto), 1, archivo)) {
      printf("%d\t%-10s\t%d\t\t%d\n", producto.codigo, producto.nombre,
producto.cantidad, producto.precio);
   }
}
```

Ver la lista de compras

```
void verLista(Nodo *cabeza) {
   Nodo *actual = cabeza;
   printf("Lista de Compras:\n");
   while (actual != NULL) {
      printf("%s\t%d\n", actual->nombre, actual->cantidad);
      actual = actual->sig;
   }
}
```