Documentación Inventario

El programa busca crear una solución de un inventario de objetos limitados con las funciones de :

- Agregar
- Eliminar
- Modificar
  Para el manejo del mismo.

Librerias y Constantes

Utilizamos dos Librerias

- **<studio.h>**: Básica para programas de C
- **<string.h>:** Manipulación de Strings.
  Utilizamos dos constantes
- **item_maximo :** Constante que determina la cantidad maxima de items que tendrá el inventario.
- **item_longi:** Determina que tantos caracteres puede tener el nombre de los objetos.

Programa

Se utilizó **estructuras de datos** para almacenar los objetos del inventario y sus características, el uso de datos de tipo estructuras nos permite guardar varios datos asociados que manejan un diferente tipo.

Definimos la estructura:

```c
struct Item{
    char nombre[item_longi];
    int cantidad;
    float precio;
};
```

Cada objeto contendría un nombre (string), una cantidad (entero) y un precio(decimal).

Variables Globales

Se usa un **arreglo de estructuras** para almacenar las diferentes entradas del inventario, asignándole el máximo de ítems que contendrá como magnitud del array. Además declaramos un contador de items que nos sirve para llevar la cuenta de cuantos objetos existen en nuestro inventario.

C

```c
 //Array de Estructuras
struct Item inventario[item_maximo];

//Contador de items
int num_items = 0;
```

Funciones

Se definen las funciones mencionadas con anterioridad.

Agregar Items

La función agregar comienza con la verificación de si el inventario está lleno para permitir o negar el ingreso de otros datos.

C

```c
    if(num_items >= item_maximo){
        printf("Inventario Lleno\n");
        return;
```

Después se crea un nuevo dato de tipo estructura llamado "Nuevo Item" que nos servirá para completar las características que vamos a agregar por separado.

C

```c
    //Creación de item de ingreso
    struct Item nuevo_item;
```

En la agregación se pide el ingreso de el nombre, cantidad y precio del objeto a agregar al inventario y se le asigna al dato creado anteriormente.
Para el ingreso de el nombre se utiliza *"fgets()* para obtener la cadena de caracteres que se ingrese por el usuario.

```c
    printf("Ingrese el nombre del objeto:\n");
        fgets(nuevo_item.nombre, item_longi, stdin);
        nuevo_item.nombre[strcspn(nuevo_item.nombre, "\n")] = '\0';
```

- El uso de la línea 3 es para cambiar el ultimo carácter de la cadena ingresada que es un salto de línea, por un carácter nulo que representa el final de la cadena.
  El ingreso de el precio y la cantidad se hacen de la misma manera con el uso de un scanf y su asignación a la ubicación de la variable.

```c
            printf("Ingrese la Cantidad: ");
        scanf("%d", &nuevo_item.cantidad);

        printf("Ingrese el precio: ");
        scanf("%f", &nuevo_item.precio);
        getchar(); //Limpieza Buffer
```

- La línea 6 que contiene el *"getchar()"* se encarga del limpiar el buffer del programa, refiriéndose a que limpia la cola de inputs que puede dejar el uso de scanf.

Pasamos a la asignación de el nuevo item en el lugar correspondiente con la ayuda del contador definido con anterioridad.

```c
    inventario[num_items] = nuevo_item;
        num_items++;

        printf("Item Agregado.\n");
```

Eliminar Items

La función de eliminar primero requiere de un atributo nombrado como índice que permitirá saber cual es el ítem a eliminar.
Primero se verifica que el ítem seleccionado sea valido, mediante una sentencia condicional que mira su el índice esta en los valores existentes en el inventario.

```c
    void Eliminar(int indice){
```

```
    //Asegurar item válido.
        if(indice < 0 || indice >= num_items){
            printf("Indice inválido.\n");
            return;
        }
```

Para ejecutar la eliminación usamos un bucle for que sustituya el elemento que queremos eliminar por el elemento siguiente del arreglo y que siga "adelantando" los elementos en uno, así por ejemplo, si eliminamos el elemento 2, el elemento 3 pasará a ser el 2 y el 4 pasará a ser el 3.

C

```
    for (int i = indice; i < num_items - 1; i++){
            inventario[i] = inventario[i + 1];
        }
```

Para después restar en 1 la variable global que representaba los elementos del inventario.

C

```
        num_items --;
        printf("Item Eliminado.\n");
    }
```

Modificar Items

Para modificar Items de igual manera usamos un atributo índice que nos indicará que elemento del inventario queremos modificar.
También nos aseguramos que el inventario no esté vacío y que el índice elegido coincida con los presentados en el inventario.

C

```
    void Modificar(int indice){
        //Asegurar item válido
        if (num_items == 0){
            printf("Inventario Vacio\n");
            return;

        }
        if(indice < 0 || indice >= num_items){
            printf("Indice inválido.\n");
```

```
            return;
        }
```

Después de la verificación pasamos a desplegar un menú que nos permita identificar que característica del objeto del inventario se desea modificar.

C

```
    //
        printf("Item a modificar: %s\n", inventario[indice].nombre);
        printf("Elije modificación: \n");
        printf("1. Modificar nombre\n");
        printf("2. Modificar cantidad\n");
        printf("3. Modificar precio\n");

        int opcion;
        scanf("%d",&opcion);
        getchar();
```

Después de desplegar el menú, declaramos una variable local llamada *"opcion"* que después por el *scanf* se asignará la preferencia del usuario; terminamos con un *getchat()* para prevenir errores con el buffer.

Dependiendo de la opción que se elija, se ejecutará un proceso distinto. respectivo a la característica que queremos modificar.

C

```
    //Casos de modificación
        switch(opcion){
            case 1:
                printf("Ingresa el nuevo nombre: ");
                fgets(inventario[indice].nombre, item_longi, stdin);

    inventario[indice].nombre[strcspn(inventario[indice].nombre, "\n")] =
    '\0';
                break;

            case 2:
                printf("Ingresa la nueva cantidad: ");
                scanf("%d", &inventario[indice].cantidad);
                getchar();
                break;

            case 3:
                printf("Ingresa el nuevo precio: ");
```

```
                scanf("%f", &inventario[indice].precio);
                getchar();
                break;

            default:
                printf("Opción Inválida\n");
                return;

        }

        printf("Item modificado.\n ");
```

- El el cambio de nombre, se vuelve a hacer un cambio de un caracter de salto de linea por un caracter nulo que representa el fin de la cadena de caracteres.
- Cuando se usa el *scanf* se sigue usando después la limpieza de buffer para la eliminación de posibles errores.

Mostrar Inventario

Para mostrar el inventario de ese momento, definimos una función que imprima, mediante un bucle, las características de los elementos disponibles en el inventario.

C

```
    void Mostrar(){
        //Asegurar Inventario
        if(num_items == 0){
            printf("Inventario Vacío.\n ");
            return;
        }

        //
        printf("INVENTARIO. \n");
        printf("-----------------------------------------------------
    \n");

        for (int i = 0; i < num_items; i++){
            printf("ID: %d | %s\n",i ,inventario[i].nombre);
            printf("    Cantidad: %d\n",inventario[i].cantidad);
            printf("    Precio: %.2f\n",inventario[i].precio);
        }
         printf("-----------------------------------------------------");
    }
```

- Al principio de la función, volvemos a verificar que el inventario no esté vacío.

Ejecución

Dentro del *"int main"* que es la parte principal del programa, primero definimos las variables que usaremos con más frecuencia que son:

- Opcion (Entero)
- indice(Entero)
  Su principal función será expresar los elementos a los que queremos acceder.

C

```c
int main(){

    int indice, opcion;
```

Dentro del main, usamos un bucle while que mantiene un 1 como condición de verdad, que hace que el programa se repita hasta que se decida terminar de ejecutarlo. después se despliega un menú de opciones que tiene como objetivo el guiarnos a las diferentes funciones disponibles en el programa.

C

```c
while (1) {

        //Menú de opciones
        printf("\n    MENU INVENTARIO   \n");
        printf("1. Mostrar Inventario\n");
        printf("2. Agregar item\n");
        printf("3. Eliminar item\n");
        printf("4. Modificar item\n");
        printf("5. Salir\n");

        printf("Seleccione una opción: ");
        scanf("%d", &opcion);
        getchar();
```

- Podemos observar de igual manera la sección encargada de asignar la opción elegida a la variable declarada con el mismo nombre anteriormente.
  Para la ejecución del programa utilizamos un *"switch"* que nos permite tener diferentes casos los cuales corresponden al menú presentado anteriormente.

C

```
    //Ejecución
            switch(opcion){
                case 1:
                    Mostrar();
                    break;

                case 2:
                    Agregar();
                    break;

                case 3:
                    Mostrar();

                    printf("Ingresa el ID del item a eliminar:\n");
                    scanf("%d", &indice);
                    getchar();
                    Eliminar(indice);
                    break;

                case 4:
                    Mostrar();

                    printf("Ingresa el ID del item a modificar:\n");
                    scanf("%d", &indice);
                    getchar();
                    Modificar(indice);
                    break;

                case 5:
                    printf("Programa Terminado.");
                    return 0;

                default:
                    printf("Opción Invalida\n");

            }
        }

        return 0;
    }
```

- Consideramos los *"case 3"* y *"case 4"* ya que se llama a la función *"Mostrar"*, para luego mediante *"scanf"* dejar que el usuario decida en que elemento del inventario se quiere ejecutar ya sea la eliminación o modifficación del mismo
- El programa terminará cuando el usuario elija la opción de *"case 5"* que es salir del programa