

**Desarrollo de Software**

**Soccer Manager**

**GERALDINE ANDREA REYES**

****

****

**ANDRES GUAMPE**

**JEISSON LINARES**

# Tabla de Contenido

Table of Contents

[Tabla de Contenido 2](#_Toc75187602)

[Introducción 2](#_Toc75187603)

[Objetivo 3](#_Toc75187604)

[Reto Final: Proyecto MARKET CYCLE 3](#_Toc75187605)

[Descripción del Reto 3](#_Toc75187606)

[Aspectos a tener en cuenta 4](#_Toc75187607)

[MÉTODO IDEAL 5](#_Toc75187608)

[Identificación el Problema 5](#_Toc75187609)

[Problema 5](#_Toc75187610)

[Interesados 5](#_Toc75187611)

[Objetivo 5](#_Toc75187612)

[Restricciones 5](#_Toc75187613)

[Definición del problema 6](#_Toc75187614)

[Información que conozco 6](#_Toc75187615)

[Información que debo conocer 6](#_Toc75187616)

[Estrategia 6](#_Toc75187617)

[Ejemplos particulares 6](#_Toc75187618)

[Algoritmos 7](#_Toc75187619)

[Estrategia de solución 7](#_Toc75187620)

[Requisitos de Software 15](#_Toc75187621)

[Logros 21](#_Toc75187622)

[Implementación en Python 21](#_Toc75187623)

[Pruebas **Error! Bookmark not defined.**](#_Toc75187624)

# 

# Introducción

En este documento podrá encontrar un detalle especifico de la ejecución del Proyecto Soccer Manager donde podrá visualizar información de los desarrolladores del proyecto, Planteamiento del Problema aplicado al Desarrollo de Software en lenguajes de programación C#, JavaScript, HTML, CSS, Razor. Este problema planteado será definido a partir de necesidades de la vida cotidiana en un contexto deportivo, el cual requiere de un sistema de administración eficaz y automatizado, con un menú que permita que distintos usuarios puedan ejecutar acciones de un CRUD y a su vez permitiendo que toda esta información quede debidamente almacenada.

## Objetivo

Implementar programa web enfocado áreas deportivas en específico de Futbol, utilizando los conocimientos adquiridos durante el curso de desarrollo de software para solucionar el problema planteado de automatización y control de partidos de futbol, el programa deberá tener expresiones lógicas, condicionales, bucles y módulos de C#, que involucre la toma de decisiones en una interfaz amigable para los usuarios finales; Lo anterior se lograra descomponiendo el problema inicial en requerimientos de usuario, requerimientos funcionales y requerimientos no funcionales, manejables para facilitar la implementación del programa y así poder construir funciones para organizar el código fuente y facilitar la reutilización del mismo, invocándolas con los argumentos predefinidos y parametrizados con nombres nemotécnicos, que permitan facilitar la comprensión y el seguimiento del programa, también llevando a cabo sesiones con el stake-holder para verificar avances del proyecto, donde podremos aplicar estrategia del modelo Kanban.

# Análisis del Problema

# Roles y Participantes

* Analista funcional – Andres Guampe
* Analista de negocio – Geraldine Reyes
* Experto del dominio - Jeisson Linares

#### Responsabilidades de Análisis

* Interactuar con clientes para entender necesidades y problema.
* Validar los productos para garantizar que dan valor estableciendo planteamiento del problema.
* Obtener, especificar y analizar requisitos de usuario y requerimientos Funcionales.
* Crear un modelo de prototipo donde el usuario pueda entender el alcance del desarrollo.

# Planteamiento del Problema

|  |  |
| --- | --- |
| Fecha | 8 de septiembre de 2021 |
| Temas | Definición del planteamiento del problema |

# Interacción con el cliente

El deporte es considerado como la forma de actividad física especialmente el Fútbol el cual, por medio de la organización de eventos deportivos, como competiciones, torneos entre otros, ayuda mejorar la calidad de vida, la salud mental y fortaleciendo el tejido social. Por estas razones, es necesario la aplicación de nuevas tecnologías en la sistematización del proceso, que permita la ágil recopilación de información, logrando la consulta en tiempo real y la solución a la falta de respuesta oportuna que requieren los equipos y jugadores inscritos a los torneos.

Por lo anterior El equipo de Futbol Sporting F.C ha detectado la necesidad de que la organización y administración en general de los torneos se pueda realizar de manera automatizada con el fin de evitar el uso de papel y procesos físicos de inscripción a los torneos y de seguimiento de resultados de los partidos, también de la publicación de programación y tablas de posiciones, ya que estas pueden sufrir daños y/o alteraciones inadecuadas, en donde a futuro la evidencia no puede ser hallada o consume mucho tiempo en digitalizarlo en herramientas ofimáticas no muy amigables con los cálculos de resultados y publicación de programaciones o tablas de posiciones.

El cliente requiere de una aplicación web que le permita inicialmente el registro de jugadores, entrenadores, organizadores de torneos e inclusive de espectadores, que permita por medio de distintos roles la ejecución de funciones especificas por ejemplo para el organizador crear torneos, para el entrenador inscribirse en torneos, jugador inscribirse en equipo, entre otras que se profundizaran más adelante.

La aplicación debe reducir la necesidad de los organizadores de torneos permitiéndoles via web y en tiempo real crear un torneo o liga de futbol, donde le solicite detalles del mismo

Con los datos anteriores el organizador podrá publicar su torneo y por consiguiente otros roles como el de entrenador o líder de un equipo a través de la interfaz web podrán sobre torneos ya publicados por un organizador ver sus detalles e inscribir uno o mas equipos los cuales deberán ir conformados por jugadores que también tendrán acceso web y podrán ver programación de partidos, resultados de los mismos y tabla de posiciones.

En el registro inicial al acceder a la aplicación deberá solicitarle a cualquiera de los roles la siguiente información: Nombres, Apellidos, Tipo de identificación, Numero de identificación, Numero de celular, Correo electrónico, Fecha de Nacimiento, Genero y por último rol que desempeñara en la aplicación en la cual deberían estar los propuestos por el cliente: Organizador, Entrenador, Jugador, Arbitro y Espectador. Permitiéndoles actualizar el rol cuando lo necesiten ya que un jugador también puede ser espectador y un entrenador también puede convertirse en organizador.

Al seleccionar el rol de organizador le debería llevar a una interfaz donde le pida seleccionar entre dos opciones:

1. Crear Torneo: En esta opción el organizador deberá diligenciar los siguientes datos: el nombre del torneo, la cantidad de canchas que tiene disponibles, el nombre de las canchas y su ubicación, el valor de la inscripción, el porcentaje total de la inscripción que brindara como premiación, Numero de posiciones que premiara, Porcentaje de premiación para cada posición, la fecha inicial y final del torneo, rama, categoría, numero de rondas, reglamento(PDF adjunto), una opción que le permita definir si es torneo o liga, el mínimo y máximo número de equipos para que se pueda llevar a cabo el torneo, el número de jugadores mínimo y máximo permitidos por cada equipo, los días que se llevara a cabo el torneo o que el organizador tiene disponibilidad de canchas, al final debe encontrar una opción de publicar torneo y otra de cancelar o volver al menú Organizador.
2. Administrar Torneos:En esta opción el organizador podrá ver el listado de los torneos que ha creado, podrá editar algunos detalles de cada torneo, como máximo número de equipos, fecha inicial, fecha final, días que se llevará a cabo el torneo, numero de canchas disponibles, al final le deberá dar la opción de publicar actualización o volver al menú administrar. En este menú también debe mostrarse una opción de eliminar torneo siempre y cuando este no tenga equipos inscritos.
3. Administrar árbitros: Le permitirá ver un listado de árbitros que han solicitado inscribirse a sus torneos ya creados, aquí podrá validar sus detalles, sus certificados y agregarlo o eliminarlo como juez para el torneo que selecciono el árbitro.
4. Administrar Partidos: Le brindara las siguientes opciones:
   1. Programar partido: En esta opción le permitirá seleccionar un equipo local, un equipo visitante, cancha y sus detalles, fecha y hora, numero de ronda, Árbitros, en esta opción de árbitros deberá seleccionar arbitro de línea, arbitro de mesa, arbitro principal de una lista desplegable que tenga de los árbitros admitidos para este torneo, al final le debe brindar la opción de publicar partido.
   2. Listar partidos: Le permitirá ver una lista de los partidos que ha programado y junto a ellos la opción editar o cancelar.
5. Mi Perfil: Le debe mostrar sus detalles de registro y modificarlos o actualizarlos.

Al seleccionar el Rol Participante le debe llevar a una interfaz donde le solicite seleccionar si es jugador, entrenador (líder de equipo), Arbitro:

1. Entrenador (líder de equipo): Podrá ver un submenú que le muestre las siguientes opciones:
   1. Mis Torneos: En esta opción se listarán los torneos en los que tiene equipos inscritos, inscribir jugadores al equipo y los detalles del torneo, dentro del torneo ver resultados y tablas de posiciones
   2. Mis partidos: Le debe listar un cronograma de los próximos partidos que debe jugar sus equipos.
   3. Torneos: Se listarán todos los torneos publicados y le brindara la opción de ver detalles, Inscribir Equipo. Al seleccionar inscribir equipo le deberá solicitar los siguientes detalles: Nombre del equipo, Color de Uniforme, Inscribir jugadores y al dar inscribir le deberá permitir buscar por algún atributo (pendiente de definir) del jugador y al encontrarlo agregarlo al equipo, permitiendo disolver el papeleo del siguiente formato de inscripción:

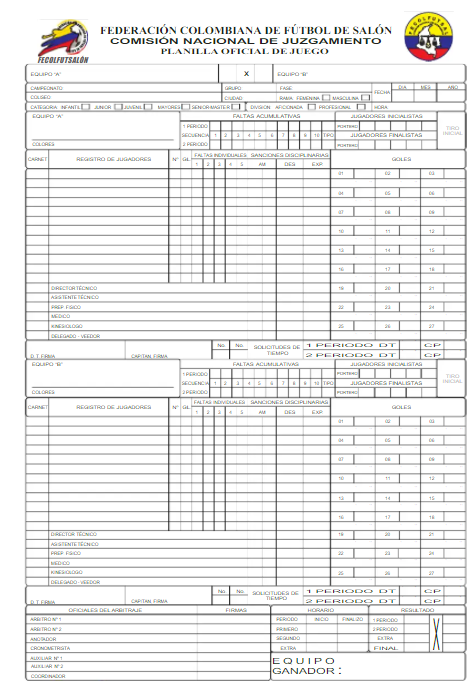


* 1. Mis Equipos: Le permite listar los equipos que ha inscrito en torneos y le permite eliminar jugadores o agregar en caso de que no cumpla aun con el número máximo de jugadores.
  2. Mi Perfil: Le debe mostrar sus detalles de registro y modificarlos o actualizarlos.

1. Jugador: Le deberá mostrar un submenú donde pueda visualizar las siguientes opciones:
   1. Mis torneos: En esta opción se listarán los torneos en los que está inscrito y los detalles del torneo, dentro del torneo ver resultados y tablas de posiciones.
   2. Mis partidos: Le debe listar un cronograma de los próximos partidos que debe jugar.
   3. Torneos: Se listarán todos los torneos publicados y le brindara la opción de ver detalles, Inscribir Equipo.
   4. Mis Equipos: Podrá ver un listado de equipos a los que está inscrito y editar detalles como posición y numero de dorsal dentro de cada equipo.
   5. Mi Perfil: Le debe mostrar sus detalles de registro y modificarlos o actualizarlos.

En el Rol de Arbitro le debe llevar a un menú donde le permita ver lo siguiente:

1. Mis Torneos: En esta opción se listarán los torneos en los que los organizadores lo han aprobado para ser arbitro.
2. Mis partidos: Le debe listar un cronograma de los próximos partidos que debe asistir y frente a cada uno la opción detalles donde le mostrara el nombre y ubicación de la cancha, fecha y hora, el nombre de los dos equipos. Adicional a detalles debe haber un botón que se llame dirigir donde le llevara aun formulario que contenga la mayoría de los siguientes campos:
   1. Nombre de equipo local, Nombre de equipo visitante, Color uniforme local, Color uniforme visitante, Nombre de torneo, Ronda, Fecha, Hora, Nombre de Cancha, ciudad, categoría, rama, Goles equipo visitante, Goles equipo Local, Faltas equipo visitante, faltas equipo local. Donde permita que el siguiente formulario físico de arbitraje se pueda llevar en un formulario web.



1. Torneos: Se listarán todos los torneos publicados y le brindara la opción de ver detalles, Inscribir, al brindar opción inscribir le debe solicitar adjuntar el certificado que lo cataloga como árbitro oficial.
2. Mi Perfil: Le debe mostrar sus detalles de registro y modificarlos o actualizarlos.

# Nombre de la aplicación

**Soccer Manager**

# Logo de la App



# Reto Final: Proyecto MARKET CYCLE

|  |  |
| --- | --- |
| Fecha | 19 de junio de 2021 |
| Temas | Método IDEAL: Proyecto MARKET CYCLE |

# Descripción del Reto

La empresa llamada “Market cicle” se dedica a la fabricación y comercialización de bicicletas

todo terreno y sus respectivos accesorios, para la empresa, uno de sus principales objetivos

es mantener contentos y bien remunerados a todos sus empleados, en especial al

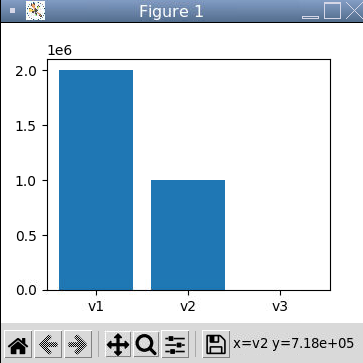
departamento de ventas y por ello ha diseñado un plan de incentivos para ellos, de esta

manera pretende mantener felices a sus empleados y sobre todo incrementar las ventas en

esta época de pandemia.

# Aspectos a tener en cuenta

* **NOTA:** Si el código es invalido mostrar un mensaje de error y no se podrá realizar el registro de la venta.
* Una vez se ha validado que el vendedor si pertenece a la empresa con el código asignado debe llenar un formulario para registrar la venta los campos del formulario son: (al ser formulario, se debe pensar en una interfaz amigable para el usuario)
  + Marca de la bicicleta Specialized [1] treck [2] BMC [3]
  + Color de la bicicleta roja [1] negra [2] roja-negra [3]
  + Tamaño de la bicicleta (S-M-L-XL)
  + Precio de la bicicleta
  + Nombre del cliente
  + Cantidad de bicicletas vendidas
  + Precio total de la venta (cantidad \* precio)
  + IVA (10% nacionales y 20% para importadas) aplicar constantes para cada caso
  + Fecha de la venta (en formato DD/MM/AAAA). La del sistema
* Se debe mostrar la factura de ser posible en el siguiente formato:
* Mostrar una grafica que permita visualizar cuantas ventas ha realizado cada vendedor, recorriendo todas las facturas, buscando el vendedor y sumando el valor de sus ventas. EL eje X debe ir con el código de cada uno de los vendedores y en el eje Y debe ir en un diagrama de barras el total sumado de todas las ventas. Utilizar Matplotlib.



# MÉTODO IDEAL

## Identificación el Problema

### Problema

El almacén MARKET CYCLE desea implementar un programa para el área de ventas, donde puedan llevar un control de todas sus ventas y les permita generar facturas a sus clientes, adicional solicita poder ver un gráfico de las ventas que realizan sus vendedores.

### Interesados

* Cliente: Market Cycle
* Usuario: Vendedores, Administrativos, Clientes del almacén.

### Objetivo

Mostrar factura de venta y grafica de ventas por cada vendedor.

### Restricciones

La factura debe incluir detalles del cliente, del producto, del almacén, Totales con y sin IVA, detalles del vendedor.

# Definición del problema

### Información que conozco

* Formato de factura que espera el cliente
* Tipo de grafica que espera el cliente
* Se puede trabajar la identificación de las entidades con códigos
* Se pueden utilizar módulos y librerías para la visualización amable de los datos.
* Productos, colores, tallas
* Productos Nacionales se les aplica 10% de IVA
* Productos importados se les aplica el 20% de IVA
* Precios de las bicicletas sin IVA

### Información que debo conocer

* ¿Cuántas graficas y de que tipo desea visualizar el usuario?
* Diferentes combinaciones entre los productos ya definidos

# Estrategia

### Ejemplos particulares

#### Ejemplo 1:

Un usuario con nombre Pepito Perez identificado con CC numero 193829381 se ha acercado e ingresado al almacén MARKET CYCLE porque quiere adquirir una cicla para su esposa y otra para su hija, nos menciona que las quiere de acuerdo a la talla de cada una, para su esposa busca una Talla M y para su hija una talla S, el vendedor mostrara las tres marcas actuales que existen que son Specialized, Treck y BMX, sus tipos los cuales son importada y nacional y en base a esto su precio base y su precio final con el IVA.

#### Ejemplo 2:

Una usuaria de nombre Juana Rodriguez con numero de cedula 873937674, pertenece a un club de enseñanza de ciclismo, la misma para poder dar sus clases correctamente requiere la compra de 4 bicicletas entre importadas y nacionales y de diferentes tallas para que la puedan usar todos sus alumnos. La usuaria quiere que dos de sus ciclas sean importadas y las otras dos nacionales, e indica que de ser posible la marca de las importadas sea Specialized y que deben ser rojas ya que los colores de su club el mayoritario es el rojo. EL vendedor deberá hacer una factura que incluya los cuatro productos aplicándoles el respectivo IVA para importadas del 20% y para nacionales del 10% y en una sola factura listar todos los detalles que le sirvan a usuaria a identificar cual es cada producto y su precio total.

# Algoritmos

### Estrategia de solución

1. Realizar la interfaz de un menú principal que permita al usuario seleccionar una de las siguientes opciones:
   1. VENDEDORES
   2. PRODUCTOS
   3. CLIENTES
   4. VENTAS
   5. GRÁFICOS
   6. SALIR
2. Las 3 primeras opciones del menú principal (Vendedores, Productos, Clientes) deben contener un submenú con las siguientes opciones:
   1. INSERTAR
   2. LISTAR
   3. CONSULTAR
   4. ACTUALIZAR
   5. ELIMINAR
   6. GUARDAR
   7. SALIR
3. La opción del menú principal VENTAS contiene el siguiente submenú:
   1. NUEVA VENTA
   2. CONSULTAR FACTURAS
   3. NOTA DE CRÉDITO (actualizar o eliminar factura)
   4. SALIR
4. La opción del menú principal GRAFICAS contiene el siguiente submenú:
   1. VENTAS POR VENDEDOR
   2. VENTAS POR MARCA
   3. SALIR
5. La opción SALIR del menú principal debe hacer que el programa salga, se solicitara confirmación antes de salir.
6. Para la opción INSERTAR del submenú Vendedores se deben solicitar los siguientes datos:
   1. CÓDIGO DE VENDEDOR
   2. NOMBRE DE VENDEDOR
   3. APELLIDO DE VENDEDOR
   4. NUMERO DE IDENTIFICACIÓN (C.C.)
   5. El código de vendedor se asignará por medio de una función para que este quede como constante y no se pueda modificar, solamente eliminar.
   6. El nombre, apellido y número de identificación serán solicitados al usuario, validando con una función que valide el tipo de entrada del usuario, que no permita espacios en blanco o iniciar con caracteres especiales.
   7. Almacenar los datos código, nombre, apellido, documento de identificación en una lista para su posterior visualización, modificación o consulta.
   8. Mostrar al usuario un mensaje con color Verde si el vendedor se ha insertado correctamente y en segundo plano guardar información en un archivo.dat para que no vaya a haber perdida de información.
   9. Mostrar en una tabla la información que fue ingresada.
   10. Se preguntará al usuario si desea ingresar otro vendedor brindando opción de SI y NO.
7. Para la opción LISTAR del submenú vendedores se deben mostrar los datos capturados en la primera opción de INSERTAR en una tabla para que el usuario pueda ver detalladamente la información.
   1. Utilizar una función que contenga el módulo panda para que se pueda dar un diseño optimo a la tabla que se presentara al usuario.
   2. Con la ayuda del módulo tabulate se dará un formato mas presentable al usuario
   3. Implementar función que permita mostrar resultados de 10 en 10 para no saturar la pantalla de información
   4. Al final de cada LISTA agregar una opción para mostrar más o para SALIR de listar y que vuelva al Submenú Vendedores.
8. Para la opción CONSULTAR del submenú Vendedores se debe implementar una función que permita al usuario Buscar por el número de código de vendedor la información correspondiente del mismo (Nombre, apellido e identificación).
   1. La función debe permitir buscar elemento por elemento de la lista Vendedores en la que guardamos inicialmente la información y extraiga la que contenga el código digitado por el usuario.
   2. Validar que la entrada sea valida y mostrar mensaje de error cuando no lo sea
   3. Si la entrada es válida mostrar el elemento ordenadamente.
   4. Brindar opción al usuario de que cuando termine de visualizar la información que necesita consultar salga al presionar una tecla y vuelva al submenú vendedores.
9. Para la opción ACTUALIZAR del submenú Vendedores debemos crear una función que le permita al usuario seleccionar que información desea actualizar escogiendo entre NOMBRE, APELLIDO, IDENTIFICACIÓN.
   1. Preguntar al usuario el código de vendedor del que desea modificar información.
   2. Crear un submenú preguntándole al usuario que desea actualizar.
   3. Hacer función que permita que la opción seleccionada sea buscado y extraído su subíndice y mostrar al usuario la información que va a actualizar.
   4. El usuario deberá introducir la información por la que va a modificar.
   5. Se mostrará un mensaje indicando que se ha actualizado con éxito
   6. Se mostrará un mensaje en tabla mostrando la nueva información.
   7. Se preguntará al usuario si desea actualizar otro código de vendedor brindando la opción SI y NO.
   8. Si el usuario selecciona NO deberá volver al submenú vendedores.
10. Para la opción ELIMINAR del submenú vendedores debemos crear una función que permita que el usuario ingrese el código del vendedor que desea eliminar.
    1. La función buscara elemento por elemento y al encontrarlo traerá la información y la mostrara al usuario preguntándole si esta seguro de que desea eliminar.
    2. Las opciones serán SI y NO, si utiliza el SI deberá eliminar toda la información de la lista, si utiliza el NO deberá volver al submenú vendedores
11. Para la opción GUARDAR del submenú vendedores debemos implementar una función que almacene la información en un archivo. Dat
    1. Cuando el usuario seleccione esta opción se mostrará un mensaje de que los datos de vendedor fueron guardados exitosamente en el archivo.dat
12. Para la opción SALIR del menú vendedores debe retornar al usuario al menú principal.
13. Para la opción INSERTAR del submenú Productos se deben solicitar los siguientes datos:
    1. CÓDIGO DE PRODUCTO
    2. MARCA DE PRODUCTO
    3. COLOR DEL PRODUCTO
    4. TALLA
    5. TIPO (Importada, Nacional)
    6. VALOR POR UNIDAD Y SIN IVA
    7. El código de producto se asignará por medio de una función para que este quede como constante y no se pueda modificar, solamente eliminar.
    8. La marca, el color, serán solicitados al usuario, validando con una función el tipo de entrada del usuario, que no permita espacios en blanco o iniciar con caracteres especiales.
    9. La talla la deberá seleccionar el usuario entre 4 opciones constantes:
       1. S
       2. M
       3. L
       4. XL
    10. El tipo deberá ser seleccionado también de un pequeño menú donde una opción sea NACIONAL y la otra IMPORTADA
    11. Almacenar los datos recopilados en una lista nombrada como productos, para su posterior visualización, modificación o consulta.
    12. Mostrar al usuario un mensaje con color Verde si el producto se ha insertado correctamente y en segundo plano guardar información en un archivo.dat para que no vaya a haber perdida de información.
    13. Mostrar en una tabla la información que fue ingresada.
    14. Se preguntará al usuario si desea ingresar otro producto brindando opción de SI y NO.
    15. Si el usuario selecciona que no deberá volver al submenú de Productos.
14. Para la opción LISTAR del submenú productos se deben mostrar los datos capturados en la primera opción de INSERTAR en una tabla para que el usuario pueda ver detalladamente la información.
    1. Utilizar una función que contenga el módulo panda para que se pueda dar un diseño optimo a la tabla que se presentara al usuario.
    2. Implementar función que permita mostrar resultados de 10 en 10 para no saturar la pantalla de información
    3. Al final de cada LISTA agregar una opción para mostrar más o para SALIR de listar y que vuelva al Submenú Productos.
15. Para la opción CONSULTAR del submenú Productos se debe implementar una función que permita al usuario Buscar por el número de código de Producto la información correspondiente del mismo (Código, Marca, color, talla, tipo, valor).
    1. La función debe permitir buscar elemento por elemento de la lista Productos en la que guardamos inicialmente la información y extraiga la que contenga el código digitado por el usuario.
    2. Validar que la entrada sea válida y mostrar mensaje de error cuando no lo sea
    3. Si la entrada es válida mostrar el elemento buscado ordenadamente.
    4. Brindar opción al usuario de que cuando termine de visualizar la información que necesita consultar salga al presionar una tecla y vuelva al submenú productos.
16. Para la opción ACTUALIZAR del submenú productos debemos crear una función que le permita al usuario seleccionar que información desea actualizar escogiendo entre MARCA DE PRODUCTO, COLOR DEL PRODUCTO, TALLA, TIPO (Importada, Nacional), VALOR POR UNIDAD Y SIN IVA.
    1. Preguntar al usuario el código de producto del que desea modificar información.
    2. Crear un submenú preguntándole al usuario que desea actualizar.
    3. MARCA DE PRODUCTO
    4. COLOR DEL PRODUCTO
    5. TALLA
    6. TIPO (Importada, Nacional)
    7. VALOR POR UNIDAD Y SIN IVA
    8. Hacer función que permita que la opción seleccionada sea buscado y extraído su subíndice y mostrar al usuario la información que va a actualizar.
    9. El usuario deberá introducir la información por la que va a modificar.
    10. Se mostrará un mensaje indicando que se ha actualizado con éxito
    11. Se mostrará un mensaje en tabla mostrando la nueva información.
    12. Se preguntará al usuario si desea actualizar otro código de producto brindando la opción SI y NO.
    13. Si el usuario selecciona NO deberá volver al submenú productos.
17. Para la opción ELIMINAR del submenú productos debemos crear una función que permita que el usuario ingrese el código del producto que desea eliminar.
    1. La función buscara elemento por elemento y al encontrarlo traerá la información y la mostrara al usuario preguntándole si está seguro de que desea eliminar.
    2. Las opciones serán SI y NO, si utiliza el SI deberá eliminar toda la información de la lista, si utiliza el NO deberá volver al submenú productos.
18. Para la opción GUARDAR del submenú productos debemos implementar una función que almacene la información en un archivo. dat
    1. Cuando el usuario seleccione esta opción se mostrará un mensaje de que los datos del producto fueron guardados exitosamente en el archivo.dat
19. Para la opción SALIR del menú productos debe retornar al usuario al menú principal.
20. Para la opción INSERTAR del submenú Clientes se deben solicitar los siguientes datos:
    1. CÓDIGO DE CLIENTE
    2. NOMBRE
    3. APELLIDO
    4. NUMERO DE IDENTIFICACIÓN (C.C. o NIT)
    5. El nombre, apellido y número de identificación serán solicitados al usuario, validando con una función que valide el tipo de entrada del usuario, que no permita espacios en blanco o iniciar con caracteres especiales.
    6. Almacenar los datos código, nombre, apellido, documento de identificación en una lista llamada Clientes para su posterior visualización, modificación o consulta.
    7. Mostrar al usuario un mensaje con color Verde si el cliente se ha insertado correctamente y en segundo plano guardar información en un archivo.dat para que no vaya a haber perdida de información.
    8. Mostrar en una tabla la información que fue ingresada.
    9. Se preguntará al usuario si desea ingresar otro cliente brindando opción de SI y NO.
    10. Si la opción es NO deberá retornar al usuario al submenú Clientes.
21. Para la opción LISTAR del submenú Clientes se deben mostrar los datos capturados en la primera opción de INSERTAR en una tabla para que el usuario pueda ver detalladamente la información.
    1. Utilizar una función que contenga el módulo panda para que se pueda dar un diseño optimo a la tabla que se presentara al usuario.
    2. Implementar función que permita mostrar resultados de 10 en 10 para no saturar la pantalla de información
    3. Al final de cada LISTA agregar una opción para mostrar más o para SALIR de listar y que vuelva al Submenú Clientes.
22. Para la opción CONSULTAR del submenú Clientes se debe implementar una función que permita al usuario Buscar por el número de código de Producto la información correspondiente del mismo (Código, nombre, apellido y número de identificación).
    1. La función debe permitir buscar elemento por elemento de la lista Clientes en la que guardamos inicialmente la información y extraiga la que contenga el código digitado por el usuario.
    2. Validar que la entrada sea válida y mostrar mensaje de error cuando no lo sea
    3. Si la entrada es válida mostrar el elemento buscado ordenadamente.
    4. Brindar opción al usuario de que cuando termine de visualizar la información que necesita consultar salga al presionar una tecla y vuelva al submenú Clientes.
23. Para la opción ACTUALIZAR del submenú Clientes debemos crear una función que le permita al usuario seleccionar que información desea actualizar escogiendo entre nombre, apellido y número de identificación.
    1. Preguntar al usuario el código de cliente del que desea modificar información escogiendo entre NOMBRE, APELLIDO, IDENTIFICACIÓN
    2. Hacer función que permita que la opción seleccionada sea buscado y extraído su subíndice y mostrar al usuario la información que va a actualizar.
    3. El usuario deberá introducir la información por la que va a modificar.
    4. Se mostrará un mensaje indicando que se ha actualizado con éxito
    5. Se mostrará un mensaje en tabla mostrando la nueva información.
    6. Se preguntará al usuario si desea actualizar otro código de cliente brindando la opción SI y NO.
    7. Si el usuario selecciona NO deberá volver al submenú clientes.
24. Para la opción ELIMINAR del submenú clientes debemos crear una función que permita que el usuario ingrese el código del cliente que desea eliminar.
    1. La función buscara elemento por elemento y al encontrarlo traerá la información y la mostrara al usuario preguntándole si está seguro de que desea eliminar.
    2. Las opciones serán SI y NO, si utiliza el SI deberá eliminar toda la información de la lista, si utiliza el NO deberá volver al submenú clientes.
25. Para la opción GUARDAR del submenú clientes debemos implementar una función que almacene la información en un archivo. dat
    1. Cuando el usuario seleccione esta opción se mostrará un mensaje de que los datos del cliente fueron guardados exitosamente en el archivo.dat
26. Para la opción SALIR del menú clientes debe retornar al usuario al menú principal.
27. Para la opción NUEVA VENTA del submenú VENTAS debemos tener los siguientes datos:
    1. CÓDIGO DE FACTURA
    2. CÓDIGO DEL CLIENTE
    3. FECHA DE VENTA
    4. CÓDIGO DEL PRODUCTO
    5. CANTIDAD
    6. CÓDIGO DE VENDEDOR
    7. De los datos anteriores el usuario deberá digitar el código del cliente, el código del producto, y la cantidad de cada producto.
    8. Se deberá realizar una función que permita al ingresar el código del cliente, traiga el nombre, apellido y numero de identificación para luego mostrarlos en la factura. Los deberá almacenar en una lista llamada factura
    9. Se debe implementar una función que traiga la hora del sistema y almacenarla en la lista factura.
    10. Se debe realizar otra función que permita que al ingresar el código del producto traiga la marca, el color, la talla, el tipo, valor unitario y IVA y todos los detalles incluido el precio total.
    11. AL tener los anteriores datos del producto deberá concatenar con el subíndice marca, el color, la talla, en una variable llamada descripción
    12. En una variable llamada subtotal deberá realizar el calculo de cantidad por valor unitario
    13. En una variable llamada valor IVA deberá realizar el calculo del valor unitario mas el IVA de acuerdo a cada producto
    14. En una variable llamada Valor total deberá calcular el valor subtotal mas el valor del IVA
    15. En una lista llamada Valor Neto de Subtotal deberá ir almacenando el VALOR TOTAL.
    16. En una lista llamada Valor Neto IVA deberá ir almacenando el Valor IVA.
    17. En una lista llamada Valor Neto Total deberá ir almacenando el calculo de la suma entre el Valor Neto Subtotal y Valor neto IVA
    18. Se debe realizar una función que pregunte al usuario si desea agregar otro producto brindando las opciones SI y NO
    19. Si el usuario selecciona la opción SI deberá ejecutar de nuevo el proceso desde el punto 27.
    20. Si el usuario selecciona la opción NO el sistema deberá guardar con un código de factura generado automáticamente toda la información solicitada y calculada desde el punto 27.
    21. Mostrar al usuario el código de factura con el que fue registrada la venta y en una variable guardar el estado de la factura como Activa.
    22. AL presionar ENTER el sistema le dejara visualizar la factura con los siguientes detalles:
        1. Nombre de almacén
        2. Dirección
        3. NIT de almacén
        4. Numero de factura
        5. Código de cliente
        6. Nombre y apellido del cliente
        7. CC o NIT
        8. Fecha de la venta
        9. Código del producto
        10. Descripción del producto
        11. Tipo del producto
        12. Cantidad
        13. Valor unitario
        14. Subtotal
        15. Valor IVA
        16. Valor Total
        17. Valor neto Subtotal
        18. Valor Neto IVA
        19. Valor Neto Total
        20. Código de vendedor
        21. Nombre y apellido de vendedor
        22. Identificación del vendedor
        23. Estado de la factura
        24. Email de almacén
        25. Teléfono de almacén
    23. Cuando el usuario pueda visualizar la factura brindar la opción de volver al submenú ventas presionando una tecla.
    24. En segundo plano La información debe ser guardada en una lista llamada facturas.
    25. Se debe mostrar un mensaje que indique que la factura fue guardada exitosamente.
28. Para la opción CONSULTAR FACTURAS se debe implementar una función que al ingresar el numero de factura permita mostrar la factura.
    1. Solicitar al usuario ingresar el numero de factura
    2. Validar el valor ingresado por el usuario y mostrar error cuando no sea válido.
    3. Buscar elemento por elemento de la lista Facturas el código ingresado por el usuario si no lo encuentra mostrar mensaje de no encontrado.
    4. Si encuentra el valor deberá mostrar la factura.
    5. Cuando el usuario pueda visualizar la factura brindar la opción de volver al submenú ventas presionando una tecla.
29. Para la opción NOTA DE CRÉDITO del submenú Ventas se debe implementar una función que permita al usuario modificar una factura a través del mecanismo legal de facturación “Nota de crédito”, esto debido a que legalmente las facturas no pueden modificarse o eliminarse, pero se pueden anular por medio de una nota de crédito y el usuario deberá luego generar otra factura. La función deberá solicitar al usuario el numero de factura que desea anular.
    1. Cuando el usuario ya ingresé el numero de la factura se deberá validar que la información ingresada sea válida y si no lo es mostrar un mensaje de error y llevarlo a que lo ingrese nuevamente.
    2. La función deberá buscar la factura ingresada por el usuario elemento por elemento en la lista facturas, si no la encuentra deberá mostrar mensaje de no encontrada y llevarlo a que ingrese el numero nuevamente.
    3. Cuando la encuentre debe mostrarla
    4. Al final una opción que pregunte si esta seguro de que quiere anularla con opción SI y NO.
    5. Si el usuario selecciona la opción NO retornara al submenú de ventas.
    6. Si el usuario selecciona SI la función deberá cambiar el estado de la variable Estado de factura a Anulada.
    7. Al traer la factura traerá los mismos valores del numero de la factura a anular y los pondrá negativos, para que anule los valores de esa venta y no afecte los reportes de ventas.
    8. Almacenar la factura modificada en la lista facturas.
    9. Guardar la información.
    10. Mostrar al usuario un mensaje indicando que la factura se ha anulado correctamente
    11. Mostrar factura modificada
    12. Permitir al usuario visualizarla y salir al submenú ventas al presionar una tecla
30. Para la Opción SALIR del submenú Ventas deberá redirigir al usuario al menú principal.
31. Para la opción VENTAS POR VENDEDOR del submenú Graficas se debe implementar una función que permita buscar elemento por elemento de la lista facturas y extraer el código del vendedor y concatenarlo con el nombre
    1. Si encuentra el vendedor sumara el valor de la venta total y lo ira acumulando hasta recorrer todas las facturas en una variable llamada suma
    2. Se armará una lista llamada eje X con la concatenación del código y nombre del vendedor
    3. Se armará una lista llamada eje Y en donde se almacenará la suma del total vendido de cada vendedor
    4. Ya teniendo los valores de los dos ejes se implementará con ayuda del módulo mathplotlib la realización de la gráfica de barras
    5. Mostrar la grafica
32. Para la opción VENTAS POR MARCA del submenú Graficas se debe implementar una función que permita buscar elemento por elemento de la lista facturas y extraer la marca del producto
    1. Si encuentra la marca sumara el valor total de esa venta y lo ira acumulando en una variable que se llame suma de marcas
    2. Se debe armar una lista llamada marcas que contenga las marcas de bicicletas encontradas en el recorrido de las facturas
    3. Se armará una lista llamada suma de marcas que ira almacenando el valor de la suma de cada marca
    4. Ya teniendo los valores de los dos ejes se implementará con ayuda del módulo mathplotlib la realización de la gráfica de torta
    5. Mostrar la grafica
33. Para la Opción SALIR del submenú Graficas deberá redirigir al usuario al menú principal.

### Requisitos de Software

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **CÓDIGO** | **PR\_01\_MP\_01** | | | |
| **PROPÓSITO** | MOSTRAR EL MENU PRINCIPAL | |  |  |
| **PARÁMETROS DE ENTRADA** | **IDENTIFICADOR** | **DESCRIPCIÓN** | **TIPO** | **RESTRICCIÓN** |
| NONE | NO REQUIERE PARAMETROS | STR | NONE |
| **VALORES DE RETORNO** | PRINT | IMPRIME EL MENU PRINCIPAL | STR | NONE |
| **ENCABEZADO FUNCIÓN** | DEF MENUPRINCIPAL(): | | | |
|  |  |  |  |  |
| **CÓDIGO** | **PR\_01\_MP\_02** | | | |
| **PROPÓSITO** | PERMITE LIMPIAR LA PANTALLA PARA NO ACUMULAR INFORMACION INNECESARIA EN PANTALLA DE USUARIO | | | |
| **PARÁMETROS DE ENTRADA** | **Identificador** | **Descripción** | **Tipo** | **Restricción** |
| NONE | NO REQUIERE PARAMETROS | STR | NONE |
| **VALORES DE RETORNO** | OS.SYSTEM | FUNCION DEL SO QUE BORRA LA PANTALLA | MODULO DE BIBLIOTECA | SE DEBE TENER EN CUENTA EL SO QUE SE VA A UTILIZAR |
| **ENCABEZADO FUNCIÓN** | def setLimpiar() | | | |
|  |  |  |  |  |
| **CÓDIGO** | **PR\_01\_MP\_03** | | | |
| **PROPÓSITO** | PERMITE PONER COLOR A TEXTOS ESPECIFICOS | | | |
| **PARÁMETROS DE ENTRADA** | **Identificador** | **Descripción** | **Tipo** | **Restricción** |
| COLOR | SE DEFINEN DENTRO DE LA FUNCION COLORES BASICOS PARA SER UTILIZADOS | NONE | NONE |
| **VALORES DE RETORNO** | RETURN | COLOR | MODULO | NONE |
| **FIRMA / ENCABEZADO FUNCIÓN** | def setColor(col) | | | |
|  |  |  |  |  |
| **CÓDIGO** | **PR\_01\_MP\_04** | | | |
| **PROPÓSITO** | PONER UN MENSAJE CON COLOR PREDEFINIDO | | | |
| **PARÁMETROS DE ENTRADA** | **Identificador** | **Descripción** | **Tipo** | **Restricción** |
| COLOR | SE UTILIZA LA FUNCION SETCOLOR PARA DEFINIR EL COLOR DEL MENSAJE | FUNCION | def setColor(color) |
| TEXTO | SE DEFINE EL MENSAJE QUE SE QUIERE MOSTRAR | STR |  |
| **VALORES DE RETORNO** | PRINT | IMPRIME EL MENSAJE DEFINIDO CON EL COLOR DEFINIDO |  |  |
| **FIRMA / ENCABEZADO FUNCIÓN** | def mensaje(color, texto): | | | |
|  |  |  |  |  |
| **CÓDIGO** | **PR\_01\_MP\_05** | | | |
| **PROPÓSITO** | PERMITE QUE EL USUARIO CONFIRME UNA ACCION SOBRE EL PROGRAMA PULSANDO LA TECLA ENTER | | | |
| **PARÁMETROS DE ENTRADA** | **Identificador** | **Descripción** | **Tipo** | **Restricción** |
| COLOR | SE UTILIZA LA FUNCION SETCOLOR PARA DEFINIR EL COLOR DEL MENSAJE | FUNCION | def setColor(color) |
| TEXTO | SE DEFINE EL MENSAJE QUE SE QUIERE MOSTRAR ANTES DE PRESIONAR ENTER O PARA QUE EL USUARIO LEA QUE DEBE PRESIONAR ENTER | STR |  |
| **VALORES DE RETORNO** | PRINT | IMPRIME EL MENSAJE DEFINIDO CON EL COLOR DEFINIDO |  |  |
| **FIRMA / ENCABEZADO FUNCIÓN** | def pausaEnter(color, texto) | | | |
|  |  |  |  |  |
| **CÓDIGO** | **PR\_01\_MP\_06** | | | |
| **PROPÓSITO** | PERMITE QUE EL PROGRAMA HAGA UNA PAUSA EN SEGUNDOS PARA QUE EL USUARIO LEA UN MENSAJE DEFINIDO CON UN COLOR | | | |
| **PARÁMETROS DE ENTRADA** | **Identificador** | **Descripción** | **Tipo** | **Restricción** |
| COLOR | DEFINE CON LA FUNCION SET COLOR EL COLOR EN QUE SE QUIERE MOSTRAR EL MENSAJE | FUNCION | NONE |
| TEXTO | DEFINE EL MENSAJE A MOSTRAR PARA EL USUARIO | STR | DEBE SER UN MENSAJE CORTO Y ESPECIFICO |
| SEGUNDOS | DEFINE EL NUMERO DE SEGUNDOS QUE QUIERE PAUSAR EL PROGRAMA | FUNCION | NONE |
| **VALORES DE RETORNO** | TIME.SLEEP | UTLIZA FUNCION DEL SISTEMA TIME .SLEEP PARA PAUSAR EL PROGRAMA | NONE | NONE |
| **FIRMA / ENCABEZADO FUNCIÓN** | def pausaSegundos(color, texto, segundos) | | | |
|  |  |  |  |  |
| **CÓDIGO** | **PR\_01\_MP\_07** | | | |
| **PROPÓSITO** | DEFINE EL DISEÑO DE LA CABECERA DEL PROGRAMA | | | |
| **PARÁMETROS DE ENTRADA** | **Identificador** | **Descripción** | **Tipo** | **Restricción** |
| TITULO | SE AGREGA EL TITULO DEL ENCABEZADO DE ACUERDO A LA SECCION EN LA QUE SE DEFINA LA FUNCION | STR | NONE |
| **VALORES DE RETORNO** | PRINT | IMPRIME EL ENCABEZADO | STR | NONE |
| **FIRMA / ENCABEZADO FUNCIÓN** | def setCabecera(titulo) | | | |
|  |  |  |  |  |
| **CÓDIGO** | **PR\_01\_MP\_08** | | | |
| **PROPÓSITO** | LAS OPCIONES DEL SUBMENU DEL CRUD QUE QUERAMOS MOSTRAR | | | |
| **PARÁMETROS DE ENTRADA** | **Identificador** | **Descripción** | **Tipo** | **Restricción** |
| TITULO | SE DEFINE EL TITULO PARA QUE PODAMOS LLAMAR LA FUNCION SET CABECERA | STR | NONE |
| **VALORES DE RETORNO** | PRINT | IMPRIME LAS OPCIONES DEL SUBMENU | STR |  |
| **FIRMA / ENCABEZADO FUNCIÓN** | def menu(titulo) | | | |
|  |  |  |  |  |
| **CÓDIGO** | **PR\_01\_MP\_09** | | | |
| **PROPÓSITO** | VALIDA QUE LA ENTRADA DEL USUARIO SEA UN NUMERO ENTERO, NO ACEPTA ESPACIOS, CARACTERES ESPECIALES, O TEXTO | | | |
| **PARÁMETROS DE ENTRADA** | **Identificador** | **Descripción** | **Tipo** | **Restricción** |
| NUMERO | SOLICITA EL NUMERO QUE SE QUIERE VALIDAR | INPUT | NONE |
| **VALORES DE RETORNO** | RETURN | NUMERO | INT | NONE |
| **FIRMA / ENCABEZADO FUNCIÓN** | def Validar\_Enteros(num) | | | |
|  |  |  |  |  |
| **CÓDIGO** | **PR\_01\_MP\_10** | | | |
| **PROPÓSITO** | BUSCA EN LA LISTA UN CÓDIGO Y RETORNA TRUE O FALSE DEPENDIENDO SI LO ENCUENTRA | | | |
| **PARÁMETROS DE ENTRADA** | **Identificador** | **Descripción** | **Tipo** | **Restricción** |
| LISTA | SOLICITA UNA LISTA PARA BUSCAR EL VALOR | LIST | NONE |
| CODIGO | SOLICITA UN CODIGO O UN VALOR DEFINIDO QUE TENGAN TODOS LOS ELEMENTOS DE LA LISTA O EL VALOR BUSCADO | STR | NONE |
| **VALORES DE RETORNO** | RETURN | TRUE, FALSE | BOOLEAN | NONE |
| **FIRMA / ENCABEZADO FUNCIÓN** | def validarDuplicado(lista, cod) | | | |
|  |  |  |  |  |
| **CÓDIGO** | **PR\_01\_MP\_11** | | | |
| **PROPÓSITO** | MUESTRA EL REGISTRO SOLICITADO | | | |
| **PARÁMETROS DE ENTRADA** | **Identificador** | **Descripción** | **Tipo** | **Restricción** |
| REGISTRO | SOLICITA LA LISTA QUE QUEREMOS MOSTRAR EN UN ORDEN ESPECIFICO | LIST | NONE |
| **VALORES DE RETORNO** | PRINT | IMPRIME EL REGISTRO |  |  |
| **FIRMA / ENCABEZADO FUNCIÓN** | def mostrar (registro) | | | |
|  |  |  |  |  |
| **CÓDIGO** | **PR\_01\_MP\_12** | | | |
| **PROPÓSITO** | MUESTRA LOS ELEMENTOS DE LA LISTA DE FORMA VERTICAL | | | |
| **PARÁMETROS DE ENTRADA** | **Identificador** | **Descripción** | **Tipo** | **Restricción** |
| LISTA | SOLICITA LA LISTA QUE SE QUIERE MOSTRAR | LIST | NONE |
| **VALORES DE RETORNO** | PRINT | IMPRIME LA LISTA EN FORMA VERTICAL |  |  |
| **FIRMA / ENCABEZADO FUNCIÓN** | def listar(lista) | | | |
|  |  |  |  |  |
| **CÓDIGO** | **PR\_01\_MP\_13** | | | |
| **PROPÓSITO** | BUSCA UN CODIGO EN LA LISTA, SI LO ENCUENTRA RETORNA SU INDICE, PARA EXTRAERLO Y MOSTRARLO Y SINO LO ENCUENTRA RETORNA -1 | | | |
| **PARÁMETROS DE ENTRADA** | **Identificador** | **Descripción** | **Tipo** | **Restricción** |
| LISTA | SOLICITA LA LISTA EN LA QUE SE QUIERE BUSCAR EL ELEMENTO | LIST | NONE |
| CODIGO | SOLICITA EL ELEMENTO QUE SE QUIERE BUSCAR | VAR |  |
| **VALORES DE RETORNO** | RETURN | INDICE OR POSICION |  |  |
| **FIRMA / ENCABEZADO FUNCIÓN** | def buscar(lista, codigo) | | | |
|  |  |  |  |  |
| **CÓDIGO** | **PR\_01\_MP\_14** | | | |
| **PROPÓSITO** | CARGA EL ARCHIVO DONDE SE GUARDARÁN LOS REGISTROS | | | |
| **PARÁMETROS DE ENTRADA** | **Identificador** | **Descripción** | **Tipo** | **Restricción** |
| LISTA | LOS DATOS DEL ARCHIVO SE CARGAN EN LA LISTA | LIST |  |
| ARCHIVO | SE CREA UN ARCHIVO DONDE SE GUARDARAN LOS AVLORES DE LA LISTA | DAT | NONE |
| **VALORES DE RETORNO** | RETURN | LISTA |  |  |
| **FIRMA / ENCABEZADO FUNCIÓN** | def cargar (lista,archivo) | | | |
|  |  |  |  |  |
| **CÓDIGO** | **PR\_01\_MP\_15** | | | |
| **PROPÓSITO** | GUARDA LOS ELEMENTOS DE UNA LISTA EN UN ARCHIVO.DAT | | | |
| **PARÁMETROS DE ENTRADA** | **Identificador** | **Descripción** | **Tipo** | **Restricción** |
| LISTA | SOLICITA UNA LISTA PARA PODERLA GUARDAR EN EL ARCHIVO | LIST |  |
| ARCHIVO | SOLICITA EL NOMBRE DEL ARCHIVO YA CREADO DONDE SE DEBE GUARDAR LOS VALORES DE LA LISTA | DAT |  |
| **VALORES DE RETORNO** | PRINT | IMPRIME LOS MENSAJES DE ÉXITO AL GUARDAR O DE ERROR SEGÚN CORRESPONDA |  |  |
| **FIRMA / ENCABEZADO FUNCIÓN** | def guardar(lista,archivo) | | | |
|  |  |  |  |  |
| **CÓDIGO** | **PR\_01\_MP\_16** | | | |
| **PROPÓSITO** | DEFINE UN ENCABEZADO PARA MOSTRAR EN ORDEN LOS ELEMENTOS LLAMADOS POR EL USUARIO | | | |
| **PARÁMETROS DE ENTRADA** | **Identificador** | **Descripción** | **Tipo** | **Restricción** |
| TITULOS | SOLICITA LOS TITULOS DEL ENCABEZADO A MOSTRAR | STR |  |
| **VALORES DE RETORNO** | PRINT | TITULO | STR |  |
| **FIRMA / ENCABEZADO FUNCIÓN** | def encabezado(TITULOS): | | | |

# Logros

### Implementación en Python

Link de acceso:

[Reto Final - Replit](https://replit.com/@Grupo78Ciclo1/Reto-Final#vendedores.dat)