**Manual de Procedimiento**

**PROGRAMACION II**

**Equipo:**

**Ángel David Escobedo Castillo**

**Roberto Carlos Saldívar Armendáriz**

**José Eduardo Lam Cansino**

**Maestro: Adrian Ríos**

1. **Introducción.**
2. **Ingreso al sistema.**
3. **Archivos default.**
4. **Menú principal:**
   1. **Altas.**
   2. **Bajas.**
   3. **Modificaciones.**
   4. **Consultas.**
5. **Salir.**

**Introducción**

En este manual se explicara detalladamente el código para la ejecución del programa.

Se explicara de una manera muy clara y brevemente para el entendimiento y el funcionamiento lógico que tiene dicho programa, para así hacer más fácil las actualizaciones o mantenimiento del sistema.

Además una vez explicado dicho procesamiento y pasos que tiene el programa, se mostrara al final el diagrama de flujo por si quedo alguna duda sobre cómo trabaja y como funciona.

Para su correcto funcionamiento

**Ingreso al sistema**

Antes de entrar a lo que es el menú principal nos pregunta si queremos entrar al sistema si nuestra respuesta es no saldrá del programa finalizándolo.

En caso que la respuesta sea si nos pedirá el usuario y la contraseña las cuales están explicitas en el programa las cuales están previamente validadas en caso de que te equivoques en ingresar tanto contraseña como usuario.

**Archivos Default**

Una vez dentro del sistema se te desplegara una opción la cual te pregunta si deseas cargar los registros default, ahora ¿Qué función nos blindan los registro default?

Los registro default son archivos binarios que están previamente creados y que cargamos al llamar la Clase **Cargar\_clase\_default\*.**

**\*Clase Cargar\_clase\_default.**

Esta función crea y manda a llamar a los archivos (3) y se carga en el programa, las funciones son**:**

**Crear\_Archivo\_Aereo().**

**Crear\_Archivo\_Terrestre().**

**Crear\_Archivo\_Maritimo().**

En estas funciones creadas en la clase Cargar\_clase\_default son utilizadas para ingresar en el archivo binario por default los atributos de los tres principales vehículos, para esto se utiliza una plantilla de clase para su formato un arreglo de la clase vector que nos facilitara agregar datos, y una excepción que nos avisara si el archivo fue creado con éxito o no ha sido creado.

**La clase Excepciones().**

Esta clase será fue creado con el propósito de poder darle un margen de error al no poder abrir un archivo así como informarnos el fallo que sufre dicho archivo.

**NOTA:** esta clase se utilizara cada vez que se abra el archivo en cualquier parte del código es por eso que la usamos como archivo cabecera.

**Menú Principal**

Después de haber decidido cargar o no los archivos por default, se despliega un menú que muestra lo siguiente:

OPCIONES DE ADMINISTRADOR.

1. ALTA.
2. BAJA.

3) CONSULTA.

4) MODIFICACIONES.

5) SALIR.

A continuación se describirá brevemente para después pasar a su descripción detallada del funcionamiento de que cada punto del menú principal:

**Altas:** Nos blindara la oportunidad de agregar otro vehículo con los atributos que tú le describas así como guárdalos en el archivo binario de dicho vehículo.

**Bajas:** Aquí podremos eliminar un vehículo que dimos de alta así como también uno que este por default, todo esto en el archivo binario.

**Modificaciones:** En esta opción podremos modificas algún atributo especifico del algún tipo de vehículo ya sea alguno que hayamos creado (alta) u otro que sea default, todo esto en el archivo binario.

**Consultas:** Aquí podremos consulta nuestras altas y modificaciones que tenemos en el archivo binario.

**Salir:** esta opción nos permite salir del menú principal y del programa en su totalidad

**ALTAS**

En la opción altas se manda llamar a clase Modulo\_Altas()\*. Las cual nos desplegara un submenú que no pedirá que ingresemos de qué tipo de vehículo queremos dar de alta después de haber seleccionado nos pedirá que ingresemos los atributos de vehículo al terminar creara un objeto dinámico para el tipo de vehículo que seleccionamos para después destruirlo seguidamente de eso nos pedida una tecla para continuar y acabara el proceso que puede ser repetido cuantas veces quiera.

**Clase Modulo Agregar().**

En esta Clase se crean varios arreglos de los diferentes atributos dependiendo del vehículo, se le pide al usuario que ingrese lo atributos que contendrá el registro, así mismo creara objetos dinámicos del tipo de vehículo que hayamos seleccionado, después abrirá el archivo binario y los almacena dentro y cerrara el archivo.

En caso de que el archivo no se pueda abrir se Manda a llama a la clase Excepciónes() que nos informara que el archivo está dañado o que hubo error en el proceso tanto de apertura como de cierre del archivo.

**BAJAS**

Ahora nos regresaremos en el menú seleccionando Ninguno en todos los casos hasta llegar al menú anterior y seleccionamos Bajas.

Luego se muestra el mismo menú donde se pide el tipo de vehículo que queramos seleccionar, en este caso para borrarlo.

Después se muestra una lista de los vehículos con su respectiva clave, y se te pide ingresar el número de la clave del vehículo a borrar, lo borra y ya puedes elegir si salir, o seguir borrando.

**MODIFICACIONES**

En esta opción se manda llamar a la clase **Modulo\_Modificasiones().**

**Modulo\_ Modificasiones ():**

Dentro de esta clase te desplegara un submenú donde te pedirá ingresar el tipo de vehículo que desea modificar, dependiendo del tipo de vehículo que escojamos mandara a llamar la clase del tipo de vehículo, la cual despliega otro submenú en el cual te pide que elijas la clave del atributo que deseas modificar .

Pasamos a abril el archivo así como también el llamado de la clase excepciones para validar si el archivo se abrió con éxito.

Utilizando la función de búsqueda **seek**, nos posicionamos al inicio del archivo con la ayuda de un ciclo y un contador vamos leyendo el archivo con la ayuda de la función **read** hasta encontrar el atributo deseado se sale del ciclo y te desplegara la opción de modificar el atributo deseado una vez hecho este procedimiento se escribirá en el archivo binario se cerrara y terminara la función.

Este proceso se podrá repetir cuantas veces sea necesaria la modificaron (es) deseadas.

**CONSULTAS**

En esta opción se manda a llamar a la clase **Modulo\_consultas().**

**Clase Modulo\_Consultas().**

Esta nos desplegara un submenú en el cual nos desplegara los tipo de vehículos dependiendo lo que elijas Terrestre, Marino o Aéreo, los cuales se toman de su respectivo archivo binario.

Mandamos llamar a la clase OpenCv que nos ayuda a mostrar la imagen del tipo de vehículo que seleccionamos en otra venta

**OpenCV:** es una herramienta para el manejo de imágenes. Tanto para elvisula estudio como para el dev c++.

Nosotros la utilizamos para desplejar las imágenes de los respectivos vehículos.

Podremos consultar las veces que veamos necesario.

**Nota:** Todos los valores que se piden al usuario son mandados a llamar desde las clases vehículo, terrestre, aéreo, marino. Con funciones set y getline.