Archivo de Documentación del Proyecto IVX

Versión 1.0

Por:
Manuel Alejandro Cerón

CONTENIDO

- **Otras Estructuras y Variables globales**
- **Las variables TURBO –RETRAZO**
- **Variables booleanas de control de procesos**
- **Las variables de control de la jugabilidad**

La Interfaz

Información adicional.

Introducción

Invaxion Marciana IVX es un juego de video creado como proyecto de semestre de la materia Introducción a la Informática, en el primer semestre de la carrera de Ingeniería de Sistemas de la Universidad del Cauca. El Objetivo principal del proyecto es crear un juego de estilo “mata marciano” o un clon del conocido juego “Space Invaders”, aunque el concepto general del juego es una copia, se pretenden agregar características propias que realcen el juego, y lo diferencien de la gran cantidad de juegos del mismo estilo que se pueden encontrar. Para el desarrollo del juego se decidió usar el lenguaje de programación C en el entorno Borland Turbo C++ ya que esto es lo que estamos estudiando actualmente en la materia. A través de este documento se pretende mostrar todas las características del proyecto.

Datos Generales y Archivos

El proyecto esta actualmente en su versión 1.0, y cuenta con las siguientes características:

- La nave principal del juego con capacidad de movimiento lateral, ubicada en la parte inferior de la pantalla, usa diferentes SPRITES para dar la sensación de rotación
- Capacidad de la Nave principal para disparar tres diferentes tipos de armas, los misiles, el rayo láser, y la bola de fuego, cada una con diferentes capacidades destructivas.
- La incorporación de 1 a 10 enemigos de tres tipos diferentes, cada uno con ataques diferentes, que se mueven en diferentes direcciones en la parte superior de la pantalla.
- Todos los puntos anteriormente mencionados funcionando en conjunto como un juego, además del sistema de puntuación, conteo de vidas, he indicación del arma actual en un panel en la derecha de la pantalla
- 90 Diferentes NIVELES con diferentes velocidades
- 3 distintos modos de dificultad
- Ayuda Integrada
- Una presentación, y una historia de introducción
- Un menú principal con tres diferentes opciones: JUGAR, OPCIONES, y Creditos
- Un menú de opciones que permite controlar la dificultad y otras cosas.
- Modo de PASSWORD para continuar desde un nivel posterior.
- Modo de RECORDS que se encarga de grabar los mejores puntajes obtenidos
- Pantallas de triunfo y derrotan que mejoran la interfaz.
- Posibilidad de ver los créditos de forma muy vistosa.
- Control de tiempo para cada nivel.

El juego tiene varios archivos, de los cuales su función es.

- Archivos “IVX.PRJ” y “IVX.DSK” Estos son los archivos generales del proyecto creado para TURBO C++, estos archivos contienen la información de la organización de cada uno de los módulos del proyecto
- Archivos *.CPP, *.H los archivos con estas extensiones representan cada uno de los módulos con código fuente del proyecto y sus respectivos archivos de cabecera
- Los Archivos ubicados en la carpeta “IVXGRAF” son todos los archivos de cada uno de los gráficos o SPRITES que usa el juego, cada uno tiene un nombre descriptivo
- La carpeta “IVXDATOS” contiene los demás archivos necesarios para el juego, como la fuente de letra y el mundo, están los programas que sirven para crear cada uno de estos archivos (Nota: Estos archivos tanto como los programas para crearlos están en una fase preliminares y por lo tanto no tienen documentación, es mejor, para el que no conozca bien el proyecto no tratar de cambiar estos archivos)
- El Archivo IVX.EXE representa el Programa Ejecutable del Juego y es el que permite jugar la versión actual del juego.

Uso del Juego

El uso del juego es muy sencillo:

- Las teclas [←] y [→] permiten mover la nave de forma lateral a través de la pantalla
- La Tecla [ESPACIO] permite disparar con el arma actual de la nave.
- La Tecla [F1] o la Tecla [ESC] Permite ir a la ayuda o salirse del juego
- La Tecla [P] sirve para hacer una pausa en el juego, para salir de la pausa presionar cualquier tecla
- Las telcas [ ] [ ] permiten moverse entre los menús
- La tecla [ENTER] Permite seleccionar o cambiar una opción del Menú

¿Cómo Compilar?

Para poder Ver, Modificar, Ejecutar, O Compilar el proyecto desde el código fuente, se deben seguir las siguientes instrucciones.

El juego esta creado especialmente para el entorno Borland Turbo C++, más específicamente la versión 3.0, para verlo se debe proceder así:

Primero abrir el entorno de Turbo C++, luego entrar en el menú <PROJECT> luego dar <OPEN PROJECT>, en el cuadro, buscar la carpeta donde esta ubicado

el proyecto, luego abrir el archivo “IVX.PRJ” ya dar clic en <OPEN> entonces se abrirá el proyecto.

En la parte superior aparecerá el modulo MAIN.CPP el cual es el modulo principal del proyecto, en la parte inferior izquierda aparecerá la ventana –Project- en la cual se puede seleccionar, dando doble clic cualquiera de los módulos que se desea ver.

Nota: Es muy importante que la persona que quiera ver el proyecto tenga experiencia en el trabajo con proyectos con diferentes módulos, ya que de lo contrario le va a ser muy difícil entender el código.

Introducción al Código Fuente

Para el desarrollo del proyecto IVX, el primer paso fue el análisis general, empezando la revisión de diferentes juegos del mismo estilo, para revisar exactamente las características, estos juegos fueron programas freeware obtenidos de Internet, una vez que se vio y analizó cada uno de los juegos, se pudo tener una clara visión de las características concretas que se querían para el juego. Luego de obtener las características del juego se realizo la interfaz de lo que se deseaba como objetivo final, aislando también los diferentes problemas que se necesitan para el desarrollo (movimiento de la nave, de los enemigos, disparos, etc.).

Siguiendo el modelo utilizado generalmente para el desarrollo de videojuegos se plantearon las tres estructuras necesarias para la construcción del juego que son el motor grafico, la interfaz, y el bucle principal, en cada estructura se decidió crear una estrategia de análisis diseño e implementación diferente.

Los Módulos del Proyecto

El proyecto IVX.prj usa 8 módulos diferentes los cuales son:

- **Modulo MAIN**: Es el modulo principal del juego, aquí se alberga la función principal y otras funciones que no se relacionan con ningún otro modulo es importante ver primero este modulo como una introducción al codito fuente
- **Modulo IVXENGIN:** Es el modulo que contiene todas las funciones del motor grafico que se explica en la siguiente sección.
- **Modulo GLOBALES:** Este es el módulo que guarda todas las variables globales del proyecto y que son usadas por varios módulos a la vez.
- **Módulo NAVE:** Este modulo guarda todas las funciones de la nave como las de movimiento, disparo, explosión etc.

- **Modulo MALOS:** Guarda todas las funciones relacionadas con los enemigos, se encarga de ponerlos en pantalla, moverlos, cargar los niveles, que disparen etc.
- **Modulo INI-FIN:** Se encarga de las rutinas de inicialización de datos y de finalización de procesos, por ejemplo iniciación y cierre del modo grafico, liberación de memoria etc.
- **Modulo INTERFAZ:** Se encarga de todo lo que tiene que ver con la interfaz del juego como los menús, la pantalla de inicio etc.
- **Modulo TECLADO:** Es una conversión de la librería “TECLADO.H” (que se explica posteriormente) convertida a módulo para que funcione con todos los demás

Además de los módulos anteriormente mencionados el proyecto cuenta con varios archivos de cabecera los cuales son:

- **MAIN.H** es el encargado de guardar todas las constantes, tipos de datos de todo el proyecto, además posee los prototipos de todas las funciones utilizadas en el proyecto menos las del motor grafico.
- **IVXENGINE.H** posee las constantes, tipos de datos, y prototipos de funciones del motor grafico.
- **TECLADO.H** posee los prototipos de funciones y constantes de cada una de las teclas para el sistema de teclado
- **GLOBALES.H** Contiene todas las variables que son utilizadas por mas de un modulo, las variables se guardan de la forma extern X para que no se declaren dos veces al tiempo.

El Motor Gráfico

El primer aspecto que se realizó fue el motor grafico, ya que este seria el pilar sobre el que se basarían los otros dos.

Para el desarrollo del motor del juego IVXENGINE, se realizó primero el análisis para determinar cuales eran los requerimientos necesarios para el motor, como el poder cargar y mostrar imágenes o sprites, el manejo de la paleta y deteccion de colisiones.

Después se realizó un sencillo y rápido diseño del motor, esta fase fue muy corta por que el mayor problema se presentaba en la implementación ya que las rutinas del motor grafico van muy ligadas a la maquina y es muy difícil crear un proceso general para este.

El Modo Grafico 13h

En el proceso de implementación se decidió probar el sistema grafico estándar del TURBO C++ BGI (Interfaz Grafica de Borland) pero luego de implementarlo se notaron grandes desventajas en este sistema como la velocidad y la imposibilidad de crear una pantalla virtual. Entonces se probó con el sistema grafico estándar VGA o también conocido como modo 13h.

El modo 13h es, como ya se dijo antes, el modo estándar VGA, y funciona con cualquier PC independientemente de su tarjeta grafica, trabaja a una resolución de 320x200 con 256 colores, las principales ventajas de este modo grafico son su velocidad, y el tener el control total sobre la pantalla para realizar por ejemplo una pantalla virtual.

Para mas información sobre el modo grafico 13h se pueden consultar:

Curso de Programación de Videojuegos (CPV) por Benjamín Moreno y Jesús Sánchez
Revista electrónica Gragon Scene de la Asociación de Desarrolladores Hispanos STRATOS
Curso de Programación de Videojuegos en C de PABLOSOFT
Tutorial de Programación Gráfica por FAC
Graphics Programming por Dentor y Grant Smith (ingles)

Todos estos cursos se pueden conseguir fácilmente en Internet, o si alguien esta interesado puede enviar un mensaje a rdceron@yahoo.com y se le harán llegar.

Los Archivos de Gráficos

Como el motor requería un sistema para crear los gráficos o SPRITES del juego, con un editor grafico, se necesito escoger un formato de archivos que fuera apropiado para el proyectos, al mirar los formatos gráficos mas populares como BMP, GIF o PCX se vio el problema de que estos formatos casi siempre tienen complicados sistemas de compresión y no todos trabajaban solo con 256 colores, entonces se decidió crear un formato de archivos propio el cual se llamo IVX en alusión al proyecto, este formato se basó en uno de los formatos gráficos mas simples llamado formato RAW o formato en crudo (cada byte significa uno de 256 colores) al cual se le agrego una cabecera de cuatro bytes para el ancho y el alto de la imagen.

El proceso de creación de los archivos IVX es el siguiente:

Primero se crea el modelo de la imagen en un programa de modelado 3D, el programa utilizado en el proyecto es el modelador de libre distribución STRATA 3D.
Luego se aplica RENDER a cada una de las perspectivas del modelo necesarias para los SPRITES (por ejemplo para la nave se necesitaron 7 perspectivas en

diferentes grados de rotación para dar el efecto de giro) de estas diferentes perspectivas se obtienen diferentes imágenes en formato BMP
Después las imágenes en formato BMP son editadas reduciéndolas y aplicándoles la paleta de colores usando el programa grafico Paint Shop Pro 7. Luego se guardan 2 copias una en formato PCX de respaldo y otra en formato RAW
Luego el archivo RAW se convierte en formato IVX utilizando un editor hexadecimal llamado Ultra Edit 9, insertándoles 4 bytes en los cuales se les pone los valores del ancho y del alto.

Formato IVX:

Primeros 2 bytes ancho, siguientes 2 bytes, alto, siguientes n bytes dado por n=ancho·alto son el contenido en bytes de la imagen (hay que recordar que si se trabaja con 256 colores cada color equivale a un byte)

La paleta de colores

Al utilizar un modo de 256 colores para el juego apareció la necesidad de manejar la paleta de colores que no es más que el valor de cada uno de los colores en porcentajes de RGB (Rojo Verde y Azul). El formato utilizado para cargar la paleta de colores fue un formato en crudo es decir 3 bytes para cada color donde cada byte representa el porcentaje RGB es decir 3x256 dando un archivo de 768 bytes.

El modo de creación de la paleta fue muy simple, primero se tomaron la mayoría de las imágenes y se redujeron a 256 colores en una paleta común utilizando el programa grafico Paint Shop Pro 7, así se obtuvo una paleta de colores en el formato del programa grafico, luego se guardo una imagen cualquiera con esa paleta en formato PCX del cual con un editor hexadecimal se le extrajeron los últimos 768 bytes para formar así el archivo de la paleta

El Juego de Caracteres

Para el desarrollo del motor fue muy necesario el poder escribir texto en modo grafico, y como no se estaban usando el modo grafico BGI se tuvo que crear rutinas propias para escribir y un juego de caracteres propios. Entonces se decidió crear un tipo de archivos que almacenara la fuente, que se pudiera incluir fácilmente en el motor, y se pudiera editar fácilmente.

Entonces se decidió usar caracteres de 8x8 píxeles, donde cada carácter seria representado por 64 bytes habiendo un total de 45 caracteres (25 para las letras, 10 para los números y 10 para caracteres varios) generando un archivo de 2880 bytes.

El formato del archivo de caracteres (.CAR) es así.

64 bytes por carácter donde cada byte puede ser 1 ó 0 donde 1 representa un píxel encendido y 0 uno apagado, y cada 8bytes representa una línea del carácter teniendo un total de 8 lineas. Ejemplo:

00111000
01000100
10000010
10000010
10000010
11111110
10000010
10000010
00000000

Formando así el carácter “A”.

Entre los caracteres especiales se destaca el numero 37 con forma de una nave espacial y que es utilizado para mostrar cuantas vidas le quedan al jugador.

La Pantalla Virtual

Para el desarrollo del motor grafico se utilizó un sistema de pantalla virtual el cual simplemente consiste en un sector en la memoria del mismo tamaño de la pantalla (320x200 bytes) en el cual se escriben primero los datos en lugar de a la pantalla real, luego simplemente se copia el contenido a la pantalla real.

El sistema de la pantalla virtual es algo muy simple pero que trae consigo muchas ventajas a la hora de la ejecución del juego. Entre ellas están:

Permite utilizar solo una vez la memoria de video, volcando solo una ves por cuadro la pantalla virtual, esto evita en horrible efecto de parpadeo y genera una animación fluida.
Al utilizar memoria normal en vez de memoria de video el proceso se vuelve un poco más rápido, ya que la memoria de video es más lenta.
Al utilizar memoria normal se puede usar rutinas del lenguaje C como memcpy etc.

La Estructura SPRITE

Para el almacenamiento de los gráficos en memoria, una vez cargados desde el disco se utilizó una estructura muy simple llamada SPRITE y que consiste en 2 datos enteros para el ancho y el alto, y un dato de tipo puntero que señalara los bytes correspondientes al contenido de la imagen.

Esta estructura permite la conexión entre las funciones de carga de imágenes y las que sirven para poner estas imágenes en pantalla, además de utilizarse en otras funciones como las de colisión y es la herramienta principal que usan las demás partes del juego.

El Sistema de Teclado

Después que se termino de realizar el diseño y se paso a la implementación surgió un enorme problema con todas las funciones que requerían el uso de alguna u otra forma el uso del teclado, la razón es que las funciones que usualmente brinda el C++ para el uso del teclado, generalmente congelan el programa antes de que uno pueda saber que tecla fue la que se pulso. Para resolver este problema se consulto a una lista de correo electrónico y ellos aconsejaron el uso de una librería no estándar de C++ creada por Dohn Lushine y de libre uso, a la que se llamo "TECLADO.H" la cual permitía manejar un arreglo de 128 caracteres los cuales serian 1 0 dependiendo de que tecla se presionara. De esta forma se pudo hacer sentencias como if (keys[KEY_ENTER])… las cuales facilitaron enormemente el desarrollo del juego.

El Bucle Principal

Para el desarrollo de esta parte del proyecto se decidió crear una nueva estrategia de Análisis Diseño e Implementación. Por experiencia con otros proyectos, en los cuales se realizaba completamente el diseño de todo el proyecto y luego se realizaba la implementación y que generalmente, por la gran relación que hay entre el juego y la maquina resultaba una enorme diferencia entre el diseños y la implementación resultando que al momento de implementarlo había prácticamente que rediseñarlo. Por esta realizar un conjunto de algoritmos que pudieran funcionar independiente y conjuntamente y de los cuales se decidió hacer análisis diseño e implementación para cada uno, todos estos algoritmos funcionarían dentro de una estructura o conjunto de estructuras denominado Bucle Principal.

Otra gran ventaja que tiene este es que, como el proyecto se esta construyendo en un tiempo largo, el sistema permite que se pueda enfocar los esfuerzos a un sector pequeño despreocupándose del programa global, evitando así el problema de "perder el hilo" del proyecto.

El sistema se puede apreciar de esta manera:

ESTRUCTURAS GLOBALES

BUCLE PRINCIPAL
Funciones… (Por ejemplo mover nave)

Funciones… (Por ejemplo disparar)
Funciones… (Por ejemplo mover enemigos)

El hecho consiste en que el bucle principal es una unidad funcional que puede trabajar con todos, algunos o ningún algoritmo, y cada uno, basado en estructuras globales, puede trabajar independientemente, trabajar solo o conjuntamente con otros algoritmos. Así por ejemplo el algoritmo mover nave puede trabajar solo o en conjuntos con el algoritmo mover enemigos y viceversa.

La Estructura Nave

Esta estructura global es la base para el funcionamiento de todos los algoritmos que utilizan la nave, es una simple estructura a contiene todos los datos del personaje principal del juego.

Entre los datos que maneja la estructura nave están las imágenes, las vidas la posición etc.

La estructura nave refleja su importancia en la conexión y la interacción de diferentes algoritmos entre si, por ejemplo en el algoritmo de detectar colisiones esta estructura ayuda a conectar distintos algoritmos entre si.

La Estructura Malo

Esta es la estructura encargada de la base de todos los algoritmos relacionados con los enemigos. Al igual que la estructura nave ayuda también a la interacción de estos con otros.

La estructura malos utiliza un SPRITE declarado globalmente denominado enemigos y el cual no se declara dentro de la estructura, por que hay solo tres imágenes de los enemigos muchos mas de los enemigos que en realidad hay (se repiten) por esta razón es necesario declarar las imágenes fuera de la estructura,

También por una razón parecida se declara afuera, la estructura Disparo Enemigo, la razón es que solo hay un disparo enemigo al tiempo y muchos enemigos.

La Estructura Nave de Apoyo

Esta estructura se encarga de controlar a la nave de apoyo que sale de la parte superior de la pantalla, contiene datos como su posición, así como los datos del objeto de recompensa que sale de ella

Otras Estructuras y Variables Globales

Otras variables y estructuras globales son por ejemplo la estructura Estrellas encargada del posicionamiento en 3 dimensiones de la estrellas, las variables de posicionamiento de los enemigos en bloque, que sirven para determinar las posición general de los enemigos, o la variable i, que sirve para contar cada uno de los cuadros (o Frames) del juego ayudando a saber en que momento realizar ciertas acciones.

Las variables TURBO – RETRAZO

Las variables TURBO y RETRAZO tienen en la implementación del juego controlar la velocidad de este, ambas de pueden combinar de distintas maneras para que brinden diferentes velocidades y de diferentes calidades.
La variable RETRAZO tiene como objetivo controlar la velocidad agregando, como su nombre lo dice, un tiempo de retrazo a cada cuado, así al agregar unos milisegundos de mas a cada cuadro se logra que el juego valla un poco mas lento y se mas fácil.
La variable TURBO realiza el efecto contrario a la de RETRAZO, en lugar de volver mas lento el juego lo que hace es que lo vuelve mas rápido, esto es útil cuando se quiere mas velocidad y no se puede llevando la variable retrazo a 0. La forma como trabaja la variable TURBO es que es un factor de todas las acciones que involucran movimiento en el juego, si la variable TURBO se vuelve 0 entonces no habrá movimiento y todo será quieto. Por ejemplo los enemigos se mueven un píxel por cuadro por el factor TURBO, o sea que si el cuadro la variable TURBO es 1 los enemigos se moverán un píxel por cuadro, pero si es 2 se moverá a el doble.
La variable TURBO le aumenta velocidad al juego pero le quita suavidad al movimiento es por eso que es mejor usar en lo mas posible la variable RETRAZO para controlar la velocidad.

Variables booleanas de control de procesos

Dentro del juego existen usa serie de variables booleanas que sirven para controlar la ejecución o no ejecución de ciertos procesos dentro del juego, son usadas por lo general para manejar las opciones y realizar un juego mas personalizadas, un ejemplo de estas variables son: MENSAJES_ACTIVADOS que sirve para tener la opción de poner o no poner los mensajes ubicados en la parte de arriba, así mismo funcionan las ESTRELLAS_ACTIVADAS, y APOYO_ACTIVADO, estas variables, junto con las de control de jugabilidad son usadas también para la definición de distintos niveles de dificultad. Otras variables booleanas de control de procesos son por ejemplo PRIMERA VEZ, que sirve para realizar procesos solo mientras no se lleve un juego activo, o CONTINUE_USADO que sirve para renovar las vidas cada ves que se usa un continue. Además de las

anteriores, las mas usadas dentro del juego son las variables del tipo SALIR o MENU que sirven para controlar la ejecución de ciclos (ver sección de interfaz)

Las variables de control de la jugabilidad

Las variables globales de control de jugabilidad son las que controlan las partes más importantes del juego y que permiten variar el mismo para conseguir diferentes efectos. Estas variables son usualmente utilizadas como limites para realizar determinadas acciones. Entre estas variables encontramos:

- MAX_MALOS: es utilizada para definir el numero de enemigos que habrán en cada nivel (no pueden ser mas de 10)
- MAX_ACELERACION: define que tanta aceleración máxima puede lograr la nave principal. Este es un factor que afecta en gran medida la dificultad del juego, ya que a mayor aceleración la nave será mas rápida pero será mucho mas difícil de manejar
- MAX TITILANDO: define cuantos cuadros estará titilando la nave después de recibir un impacto y perder una vida
- MAX_RESISTENCIA: define cuantos disparos necesitan los enemigos para morir, esto es crucial para el aumento de dificultad en los niveles posteriores.
- MAX_TIEMPO: esta variable sirve para indicar cuanto tiempo necesita un cada nivel para completarse, entre menor sea, mas rápido se debe terminar el nivel o se perderá.

La Interfaz

Esta fue la parte mas sencilla del proyecto, para realizarla se trabajo sobre el motor IVXENGINE se realizó un diseño para un sistema de menús utilizando el nuevo sistema grafico y se implemento dicho diseño para que funcionara de manera diferente en cada una de las funciones pero todas con el mismo núcleo, esto facilito bastante la fabricación de todas las funciones de la interfaz, y consiste en lo siguiente:

Para los menús más sencillos una variable booleana usualmente llamada MENU la cual controlaba un ciclo. Al presionar una tecla o varias teclas esta variable era cambiada para finalizar el ciclo, por ejemplo:

```
MIENTRAS (MENU)
SI TECLA_X ENTONCES
                    OPERACIÓN_X;
                    OPERACIÓN_Y;
                    MENU=FALSO;
SI TECLA_Y ENTONCES
```

```
                OPERACIÓN_A;
                OPERACIÓN_B;
                MENU=FALSO;
```

De esta forma era fácil controlar las pantallas en las que se requería una respuesta de SI o NO, como por ejemplo las pantallas de DERROTA y TRIUNFO. Para conseguir un efecto mas vistoso, en algunos menús como los de ayuda se incorporo los textos dentro del ciclo y se utilizó la función para colocar estrellas. de esta manera aparecían los textos y las estrellas moviéndose en el fondo, esperando a que se presione una tecla.

El segundo y más complicado método de menús incluían métodos de selección de múltiples opciones utilizando teclas de flechas. Para realizarlo se utilizó el mismo método del ciclo con la variable booleana pero se le agrego una variable entera para la opción y un arreglo de enteros del tamaño del nuecero de opciones. De tal manera que al momento de imprimir cada opción estas se imprimieran del color indicado por el arreglo. Así todo el contenido del arreglo tiene el mismo color a excepción del numero indicado por la opción. Por ejemplo

```
MIENTRAS (MENU)
COLOR[1]=AMARILLO;
COLOR[2]=AMARILLO;
COLOR[OPCIÓN]=ROJO

TEXTO(“OPCIÓN 1”,COLOR[1])
TEXTO(“OPCIÓN 2”,COLOR[2])

SI TECLA_ARRIBA ENTONCES
                OPCIÓN=OPCIÓN+1
SI TECLA_ABAJO ENTONCES
OPCIÓN=OPCIÓN-1
SI TECLA ENTER ENTONCES
MENU=FALSO;

SI OPCIÓN=1 ENTONCES OPERACIÓN_X
SI OPCIÓN=2 ENTONCES OPERACIÓN_Y
```

De esta forma se consiguen todos los elementos del menú de un color menos el de la opción seleccionada, permitiendo así, de forma muy vistosa subir o bajar de opción utilizando las flechas de teclado y finalizando al presionar ENTER, para después del ciclo examinar la opción y realizar una operación determinada por cada una. Este método se utilizó en funciones como la del menú principal o el menú de opciones.

El tercer y último método que se utilizo para el desarrollo de la interfaz fue el de texto SCROLL vertical que cosiste en escribir un texto en pantalla durante un ciclo

de tal forma que este fuera subiendo por la pantalla una variable de posicionamiento en el eje Y. Es un método muy sencillo y se puede ilustrar de esta manera:

```
        MIENTRAS (MENU)
TEXTO(“TEXTO1”, 10,POSICIÓN_Y)
TEXTO(“TEXTO2”, 10,POSICIÓN_Y)
TEXTO(“TEXTO3”, 10,POSICIÓN_Y)
TEXTO(“TEXTO4”, 10,POSICIÓN_Y)
POSICIÓN_Y=POSICIÓN_Y-1;
SI POSICIÓN_Y<0 ENTONCES MENU=FALSO;
```

De esta forma se daba la sensación de que el texto esta subiendo, también se le puede agregar una condicional para que termine al presionar una tecla, o al igual que en los métodos anteriores agregar estrellas de fondo o cualquier otra decoración. Este método es utilizado en funciones como los créditos o la historia.

Información adicional.

Invaxion Marciana IVX es un proyecto realizado en la Universidad del Cauca (Colombia). Por Manuel Alejandro Cerón.

Si alguien desea obtener más información sobre el proyecto y no la encuentra en este manual puede escribir a ceronman@yahoo.com y se le tratara de responder cualquier duda.

Este proyecto podrá ser utilizado, grabado, modificado, vendido, comprado, actualizado etc. por cualquier persona siempre y cuando se incluyan los nombres de los autores originales en el.

Gracias por interesarse en el proyecto.