Imagen que contiene dibujo, señal

Descripción generada automáticamenteINSTITUTO POLITECNICO NACIONAL Icono

Descripción generada automáticamente con confianza baja

ESCUELA SUPERIOR DE INGENIERIA MECANICA Y ELECTRICA

UNIDAD CULHUACAN

Monroy Cortines Jacobo Emiliano

programación Orientada A Objetos

Osornio Soto Roberto

Proyecto Final

“VIDEOGAME SOLIDWORLD32”

Reporte Completo

# GRUPO: 2CV35

**Índice**

***INTRODUCCION……………………………………………...…3***

***ANALISIS ORIENTADO A OBJETOS…………………….…..5***

***DISEÑO ORIENTADO A OBJETOS…………………………9***

**Implementación de Lenguaje Orientado a Objetos en C++…15**

**CODIFICACION…………………………………………….25**

***MANTENIMIENTO……………………………………………..51***

***CONCLUSION………………………………………………….51***

***BIBLIOGRAFIA………………………………………………52***

**Introducción.**

***Objetivo***

Crear un programa relacionado a los videojuegos, aplicando la programación Orientada a Objetos.

***Inicio***

Para iniciar este proceso debemos empezar con el orden correspondiente, empezando con la idea principal, la base de todo, posteriormente, debemos entrar con el Análisis Orientado a Objetos, para después continuar con el Diseño Orientado a Objetos, revisar los procesos que identifican este tipo de programación.

El alma de este programa se basa en los Retro videojuegos, recordando un poco como es que empezó este mundo maravilloso, el objetivo principal de este programa es la conciencia de cuanto es el tiempo que ha pasado desde que la programación sirvió para mas que “programas cotidianos” y nos dio paso a un método de des estrés como son los videojuegos sin utilizar una interfaz gráfica.

La maravilla del lenguaje de programación C++, nos permitió lograr tener este proyecto aplicando la programación Orientada a Objetos, a esto suele denominarse como” Lenguaje Orientado a Objetos”.

Quedando claro con todos estos conceptos, no podemos hablar como tal de cómo es que se tiene pensado hacer el objetivo, ya que, pertenece al proceso de abstracción (que veremos mas adelante con el Análisis Orientado a Objetos). Sin embargo, podemos hablar de cual es la idea para obtener el objetivo.

Para poder lograr el objetivo es necesario un tiempo y dedicación, pero sobre todo el conocimiento del lenguaje Orientado a Objetos, así como conocimiento en el lenguaje de C++, además de seguir el proceso correcto, tener la paciencia de programar y un orden obligatorio en el código.

***Objetivo del programador***

Este espacio esta diseñado para que el programador de unas palabras antes de aplicar la programación Orientada a Objetos, para que, en la conclusión veamos como fue el proceso de la elaboración.

Este proyecto me entusiasma demasiado, ya que, actualmente es uno de los mas ambiciosos que tengo, a pesar de que el programa dentro del campo laboral es algo sencillo, pero, para empezar, creo que es algo que puede ser bueno para el desarrollo de mi persona.

Siempre eh pensado que la creación de estos videojuegos era algo sencillo de hacer, pero la realidad es totalmente diferente, viendo la complejidad de estos, me eh propuesto seguir con este programa, este es el motivo original de porque no decidí continuar con otra idea.

la utilidad de este programa en el mundo cotidiano es bastante implícita, lo que quiero decir con esto es que, para el desarrollo humano puede llegar a ser importante, pero para el entorno social puede pasar desapercibido, no le resto importancia a este tipo de programas, para nada, sin embargo, si es claro que este programa este hecho para cierto público en específico.

Esta es la maravilla de la programación, puede haber bastantes programas, pero siempre habrá solo un publico en especifico que lo quiera consumir.

Para este programa tome en cuenta que es lo que me gusta de la programación, que es la complejidad bidireccional en los videojuegos (los arreglos Bidimensionales y unidimensionales), sin embargo, todo esto fue posible con POO, este me ayudo demasiado puesto que fue más fácil llevar el orden del programa, sabiendo que es lo que estoy haciendo sin la duda constante de en qué línea fue donde uno se equivoca.

Para la conclusión de esta parte del Proyecto, espero poder con el desarrollo de mi programa, sabiendo el conocimiento básico, para tener un seguimiento al aprendizaje de la POO y el LOO de C++. Además de saber cómo es que es el proceso de creación de estos programas.

***Conclusión temprana***

Para el desarrollo de este programa debemos tener el conocimiento Básico en POO y su LOO que fue C++, y en este caso, dominar temas específicos. Pero siempre respetando el orden de la programación junto con todos sus procesos.

**Análisis Orientada a Objetos (AOO).**

Para el análisis Orientado a Objetos, debemos empezar con el proceso más importante de la POO (desde mi opinión)

***Abstracción***

Lo que queremos lograr es un videojuego sin interfaz gráfica, solo con código respecto a C++, el código de este debe ser practico, pero a la vez debemos hacer que sea difícil acceder a los datos que pueda perjudicar la experiencia del videojuego.

Ahora, en nuestra abstracción, creemos que el mejor juego para empezar sería un videojuego de naves espaciales, que viaja por el espacio esquivando asteroides, conjunto a eso tiene un arma con el cual puede derribar los asteroides.

En nuestro proceso de abstracción, encontramos el primer objeto, seria nuestra nave, el cual creemos que la mejor opción, para nuestro diseño, sea una nave grande, amplia de arriba y mas pequeña de abajo, con el objetivo de que sea distinto al modelo convencional de una nave.

El segundo objeto, creemos que lo mejor seria que nuestros asteroides se representen de una forma simple, el cual solo viaja de abajo hacia arriba de forma aleatoria (no 100% aleatoria, solo usando una función rand) y su modelo sea una bolita pequeña para que sean más obstáculos por los cuales esquivar.

Estos dos elementos comparten en común algo importante, lo cual son sus coordenadas, las coordenadas en x & y, para saber su espacio en el programa. Por lo cual podemos tener herencia en este elemento.

Por último, el diseño de la bala, la bala es un elemento que hace que elimine los asteroides permitiéndole el paso a nuestra nave. Este simplemente seguirá una trayectoria opuesta a el asteroide, con tal de sacar al asteroide del camino de la nave, este también tendrá un sistema de puntaje que como objetivo podremos hacer que el usuario alcance el máximo puntaje posible.

***Proceso de adaptación***

¿Como podemos hacer esto?

Pues primero debemos saber que atributos y métodos deben tener cada una de nuestras clases con las que podamos hacer nuestras clases

La primera clase es *las coordenadas,* la cual será nuestra superclase, esta clase nos debe permitir saber la posición de nuestros objetos. Como atributos debemos poner la coordenada x y la coordenada y. En cuanto sus métodos, lo principal es el constructor, que nos permitirá inicializar nuestros atributos, el segundo es el destructor, para cuando termine el programa, no guarde los datos si el jugador decide reiniciar el programa y así ocupe menos memoria, por último, dos métodos que nos regresen nuestras coordenadas nuevamente.

Nuestra segunda clase es *nuestro personaje,* ahora este será nuestra subclase numero uno, puesto que ocupamos que su superclase nos de sus coordenadas. Sus Atributos serian las coordenadas antes dichas, como implementamos un choque con los asteroides, también debemos implementar un sistema de vidas, el cual debe tener 5 oportunidades por vida, entonces nuestros dos atributos independientes son corazones, que nos dirán cuantas oportunidades tenemos antes de que se acabe nuestra vida, el segundo atributo es las vidas, cuando estas se acaben, el jugador perderá. Aquí tendremos 9 métodos, los primeros dos son constructores y destructores, el tercer método será para decir si el jugador tiene o no vidas aun, el cuarto la impresión y contador de corazones, el quinto debe ser la impresión de nuestro personaje, en este caso una nave, el sexto es el borrar esa nave, el séptimo es para poder mover nuestra nave, el octavo es cuando no tenemos vidas y terminamos el juego o si tenemos para seguir continuando, el ultimo es la muerte del personaje, el cual imprime una animación de muerte y resta vidas o termina el juego.

Nuestra tercera clase es *el asteroide,* el cual es la segunda subclase de nuestra superclase coordenadas, teniendo solo 2 atributos, que son las coordenadas anteriormente mencionadas. Pero este tiene 5 métodos, los dos primeros son constructor y destructor, el tercer método es la creación de nuestro asteroide, el cuarto será la dispersión de la bala, y el ultimo es la colisión de nuestro asteroide con nuestro personaje, para saber si existe una colisión con nuestro personaje y así eliminarle un corazón o vida.

Nuestra cuarta clase es *la bala,* al igual que las dos anteriores esta es una subclase de las coordenadas para saber para donde ira la bala, sin embargo, este ya no nos regresa su valor de posición ya que continua su trayectoria hasta llegar al límite establecido. Entonces solo tomamos sus atributos como coordenadas, y sus métodos quedan como el constructor, destructor, una que nos permita ver su trayectoria y otra que nos diga el choque entre la bala y el asteroide para saber si hay una eliminación del asteroide o no.

NOTA IMPORTANTE: las dos primeras subclases, ocupan los métodos de la superclase, además del constructor y destructor, necesitamos saber donde nos encontramos en el plano x, y, utilizando una función que nos regrese el valor x & y.

***Análisis de Funciones***

Respecto a la POO, tenemos definido nuestros análisis, pero en los videojuegos, no solo ocupamos los objetos, si no nuestras funciones.

Solo desglosare rápidamente cuál es su propósito como función y como es que están declarados.

La primera es muy importante, se trata de las coordenadas de todo nuestro programa, desde las clases, hasta nuestras funciones, se trata de nuestra función gotoxy, que nos da la coordenada de nuestro objeto, ya sea definida o definida por funciones if.

La segunda trata de nuestro menú de inicio, que nos dirá cual opción deseamos, si jugar o salir del juego.

La tercera es el retorno de nuestra decisión, si queremos salir, llevara a la pantalla de final del programa, pero si deseamos jugar, nos llevara a el juego, el primero te permite escoger una opción, el segundo te permite llevarte a esa opción.

Posteriormente la nuestra función inicio, llevara a una pantalla de carga, con el objetivo de dar consejos en pantalla y después llevará a una función portada, que dirá la historia de nuestro videojuego, así como sus controles.

Nuestra quinta función, solo es una función vacía que nos da las condiciones de movimiento, trazando los limites de juego, para que todo tenga una resolución de juego, además de borrar los asteroides y condicionar las trayectorias. Esta función se puede saltar y poner en el main, sin embargo, para que nuestra función main este más limpia, decidimos crear una función que tape todo eso.

La sexta función es para ocultar nuestra barra de escritura que suele aparecer.

Por último, una función que traza el margen de juego, el área asignada para jugar en el programa, también la ocupamos para que el programa muestre esos límites en los menús.

***Conclusión de análisis.***

Para concluir con nuestro Análisis Orientado a Objetos, debemos tener en claro que este paso es fundamental para tener una idea de que deseamos realizar con exactitud.

Realmente este paso suele ser omitido, pero por no saber la importancia de este, ahora que tenemos un análisis de cómo será nuestro Diseño Orientado a Objetos, también conocido como DOO, ya podemos crear nuestras primeras ideas, implementando la herencia, el encapsulamiento y en ocasiones el polimorfismo.

En conclusión, ya tenemos claro cuales son nuestras ideas principales:

* Clases.
* Qué tipo de Clases.
* Cuantas son Superclases.
* Cuantas son Subclases.
* Cuáles son sus atributos.
* Cuáles son sus métodos.
* Como pensamos implementar esos métodos.
* Entre otros.

Además, de implementar el Análisis de funciones, que de igual manera nos permite saber que es lo que queremos hacer en nuestras funciones y el saber para que estarán diseñadas esas funciones.

Por punto de agregación, debemos agregar que todo esto debe llevar un orden, con limpieza podremos llevar a cabo nuestro proyecto y saber detectar las fallas de este, al igual que para su mantenimiento será difícil que no se realice de manera adecuada.

**Diseño Orientado a Objetos.**

Ya implementado nuestro análisis, podemos seguir adelante con nuestro diseño, empezaremos con cómo es que planeamos organizar nuestras clases, tanto su herencia, encapsulamiento, entre otros.

Posteriormente implementaremos el diagrama respectivo, haciéndonos ver cómo será el diseño de nuestra programación y sus características principales.

Luego demostramos como son estas clases en implementación con LOO de C++.

Y, por último, llegaremos a ver esos resultados que deseamos tener, y los hechos. Concluyendo así el Diseño Orientado a Objetos, también conocido como DOO.

***Organización de clases***

* SUPERCLASE: nuestra superclase será llamada “Coordenadas” dando referencia a su función.
* SUBCLASE 1: nuestra primera subclase es llamada “Personaje” dando referencia a ser el principal de nuestro programa, pero este es una nave, no como tal un personaje (nuestro objeto nave, nuestra clase personaje).
* SUBCLASE 2: esta clase será llamada “Meteoro” en referencia a los meteoritos, anteriormente los eh llamado asteroides, porque esta es la manera correcta de llamar a el objeto, sin embargo, para abreviar más el código y para facilidad del mantenimiento, lo eh decidido llamar Meteoro.
* SUBCLASE 3: la ultima clase es llamada “Bala” sencillamente esta es llamada si porque es la bala que impacta al asteroide.

Ya organizadas nuestras clases, debemos saber cuál será su herencia y su encapsulamiento y el porqué.

NOTA: todos estos elementos, tienen en conjunto común de las coordenadas, desde ahí uno puede partir para encontrar la herencia de las clases, aunque solo sea una clase que se divide para que funcionara en las 3 subclases, es muy importante, ya que, así nos ahorramos códigos y líneas (que se traduce en menos memoria).

***Herencia.***

La herencia de nuestro programa parte de la premisa de las Coordenadas, todos estos elementos en conjunto comparten el mismo factor, es por eso por lo que, en vez de realizar el mismo código para las 3 clases, podemos realizar una clase que herede todas estas condiciones.

*Herencia 1.*

En nuestra primera herencia vemos el comportamiento de la subclase 1, el cual, nos indicara la posición de nuestro personaje, regresando nuestros datos x & y.

*Herencia 2.*

En nuestra segunda herencia veremos el comportamiento de la subclase 2, el cual solo indica la aparición del meteoro en el eje x, y la dispersión de arriba hacia abajo del eje y, en esta también ocupamos su posición, regresando nuestros datos x & y para poder saber si colisiona con nuestro personaje y si quita las vidas o no.

*Herencia 3.*

La ultima herencia es el comportamiento de la subclase 3, que solo nos dará los datos de posición x & y, aquí no pediremos que nos regresen los datos, ya que, solo cancela a la posición del meteoro.

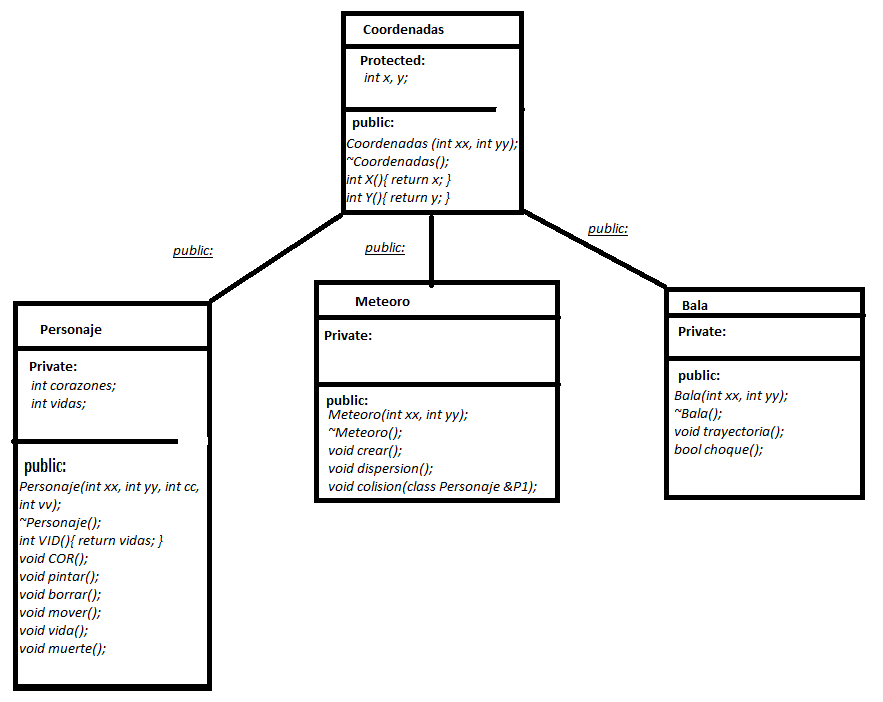
***Encapsulamiento.***

* *Coordenadas:* en nuestra superclase, el tipo de encapsulamiento es protected, protegiendo los atributos x & y, para que ninguna persona pueda acceder a ellos desde el main. Pero en la herencia, podamos acceder a ellos sin problemas.
* *Personaje:* su encapsulamiento es de tipo private, para que nadie pueda modificar las vidas y corazones que tiene por partida, y así no afectar la experiencia de usuario, además de contar con los dos atributos de Coordenadas.
* *Meteoro:* este solo cuenta con los dos atributos de Coordenadas.
* *Bala:* este solo cuenta con los atributos de Coordenadas

Ningún método fue añadido al encapsulamiento puesto que no es relevante no acceder a ellos y perjudica al mantenimiento.

***Diagrama de clases.***

*Herencia:*

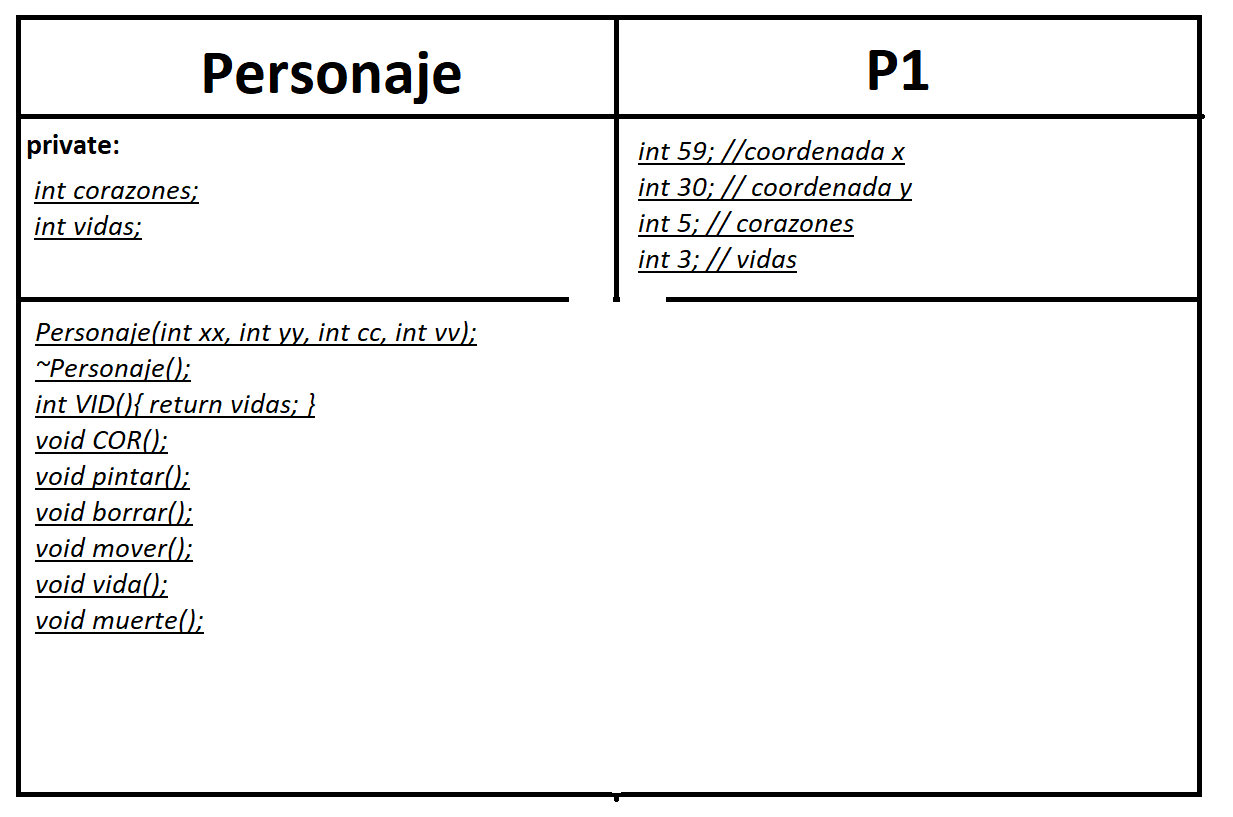
**

En este caso el tipo de herencia que se ocupo fue de tipo publica, ya que, de esa manera, al tener los atributos de tipo protegido, podemos acceder a ellos en nuestros métodos en herencia.

Cabe aclarar, que los constructores de cada una de nuestras clases, están declarando los atributos de la superclase, si no nuestras clase no podría inicializar aquellos atributos.

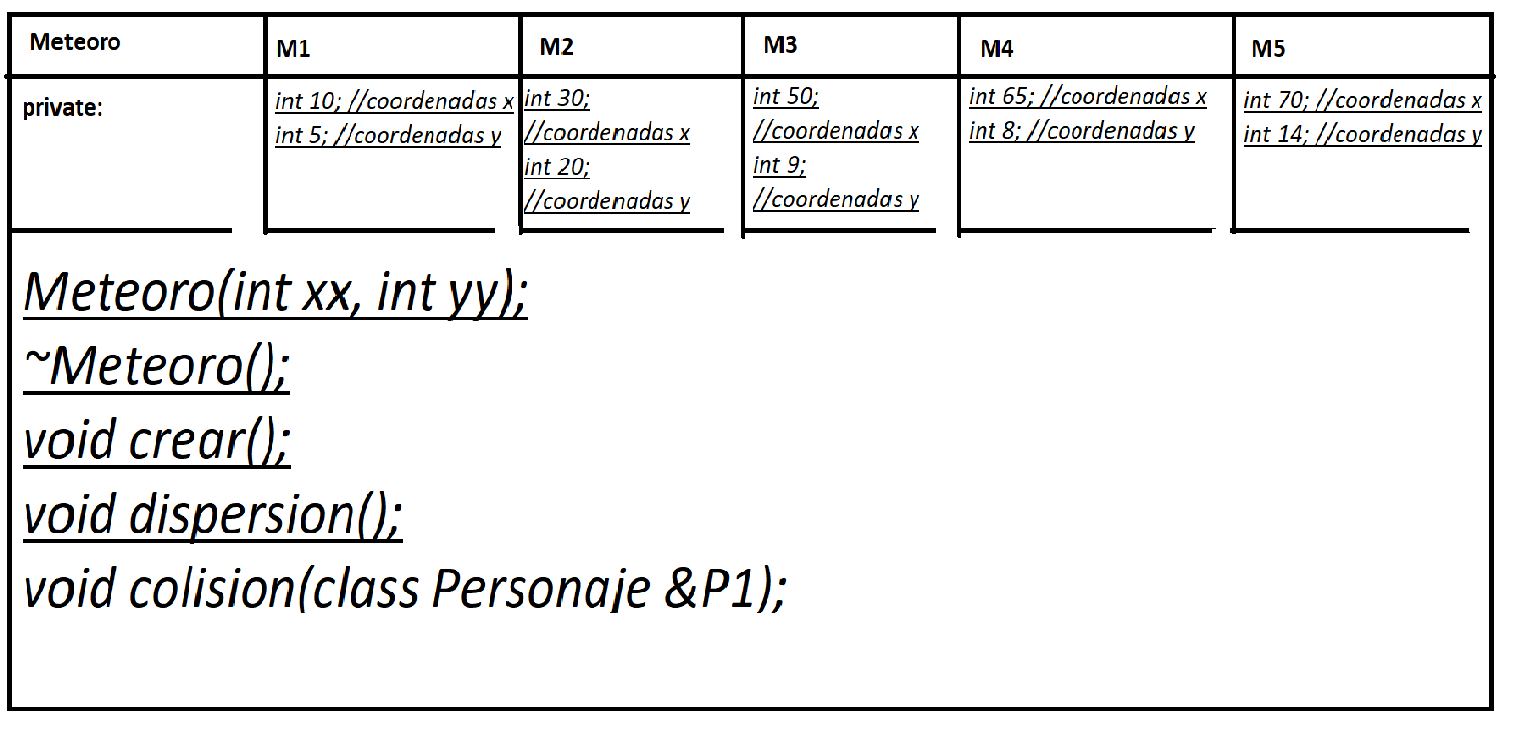
***Diagrama de Clases y Objetos.***

*Personaje:*



Este diagrama explica como es que se comporta la herencia de nuestro objeto, sin necesidad de declarar nuevamente, solo atributos nuevos.

Los métodos son los correspondientes para que nuestro personaje pueda ser implementado en el programa.

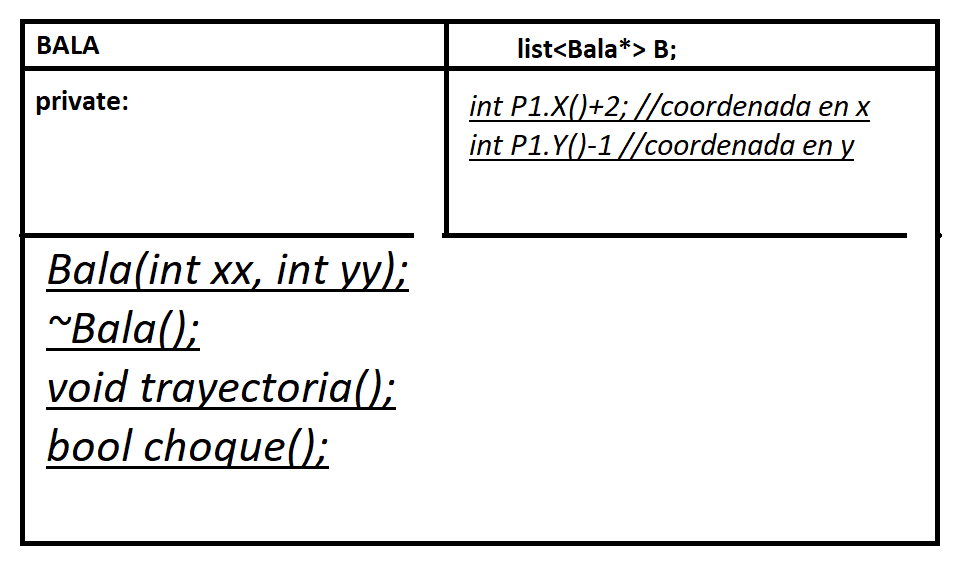
*Meteoro:*

En este diagrama explico cómo es que 5 objetos pertenecen a la misma clase, explicando como son los parámetros que hay que seguir, con la herencia de la super clase podemos decir su posición inicial (aunque después esta cambia por la función rand) y cuales son los atributos que la conforman.

En los métodos podemos visualizar un constructor, a pesar de que en su clase no tengamos ningún atributo declarado, debemos poner el constructor para inicializar nuestro objeto.

Al igual, el destructor se declara para que no almacene los datos pasados y no consuma memoria de forma innecesaria.

*Bala:*



Ahora denotamos lo mismo que lo anterior.

Pero vemos una sutil diferencia, este objeto fue declarado con un puntero, por lo cual sus coordenadas son dependientes de que es lo que haga nuestro objeto P1, sin embargo, sus propiedades en los métodos no varían con esto.

**Implementación de Lenguaje Orientado a Objetos en C++.**

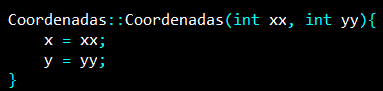
*Superclase Coordenadas*

Texto

Descripción generada automáticamente

Aquí está la implementación del LOO en la Superclase.

Los métodos, son declarados de la siguiente forma:

//constructor

Texto

Descripción generada automáticamente//destructor

*Subclase Personaje*

Texto

Descripción generada automáticamente

Aquí está la implementación del LOO respecto a la Subclase Personaje, note la herencia declarada para las coordenadas.

Los métodos son:

Texto

Descripción generada automáticamente

//constructor declarado con herencia.

Imagen que contiene Texto

Descripción generada automáticamente

//destructor

Texto

Descripción generada automáticamenteTexto

Descripción generada automáticamente con confianza media

Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

*Subclase Meteoro*

Texto

Descripción generada automáticamente

Subclase meteoro con sus métodos y atributos, al igual que su respectiva herencia.

//constructor

//destructor

Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente

*Subclase Bala*

Texto

Descripción generada automáticamente

Subclase meteoro con sus métodos y atributos, al igual que su respectiva herencia.

//constructor

//destructor

Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente

Con esto concluimos el Diseño Orientado a Objetos.

Se denota que la subclase es la que permanece por encima de las otras 3 clases puesto que cada una de las 3 depende fuertemente de sus atributos. Ya que sin ellos no se puede saber la posición de cada uno de ellos y como es que esta funciona.

***Diseño de Funciones.***

No me extenderé mucho en el tema, solo mostraré como son su prototipo, llamada y definición para conocer mejor cómo funciona el proyecto.

*Prototipo de funciones*

Texto

Descripción generada automáticamente

*Llamada de funciones*

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente

*Definición de Funciones*

Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente

Forma

Descripción generada automáticamente con confianza media

Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente Texto

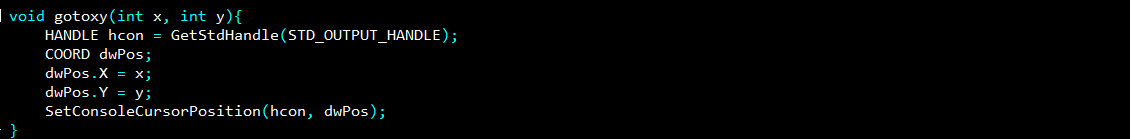
Descripción generada automáticamente

Forma

Descripción generada automáticamente con confianza media

Imagen que contiene Forma

Descripción generada automáticamente



**CODIFICACION**

**#include<iostream>**

**#include<stdio.h>**

**#include<conio.h>**

**#include<stdlib.h>**

**#include<iomanip>**

**#include<windows.h>**

**#include<list>**

**#define TECLA\_AR 72**

**#define TECLA\_AB 80**

**#define ENTER 13**

**using namespace std;**

**void gotoxy(int x, int y);**

**void inicio();**

**int retorno(int);**

**void Portada();**

**void menu();**

**void movimiento();**

**void ocultar();**

**void limites();**

**class Coordenadas{**

**protected:**

**int x, y;**

**public:**

**Coordenadas (int xx, int yy);**

**~Coordenadas();**

**int X(){ return x; }**

**int Y(){ return y; }**

**};**

**class Personaje: public Coordenadas{**

**private:**

**int corazones;**

**int vidas;**

**public:**

**Personaje(int xx, int yy, int cc, int vv);**

**~Personaje();**

**int VID(){ return vidas; }**

**void COR();**

**void pintar();**

**void borrar();**

**void mover();**

**void vida();**

**void muerte();**

**};**

**class Meteoro: public Coordenadas{**

**private:**

**public:**

**Meteoro(int xx, int yy);**

**~Meteoro();**

**void crear();**

**void dispersion();**

**void colision(class Personaje &P1);**

**};**

**class Bala: public Coordenadas{**

**private:**

**public:**

**Bala(int xx, int yy);**

**~Bala();**

**void trayectoria();**

**bool choque();**

**};**

**Coordenadas::Coordenadas(int xx, int yy){**

**x = xx;**

**y = yy;**

**}**

**Personaje::Personaje(int xx, int yy, int cc, int vv): Coordenadas(xx, yy){**

**corazones = cc;**

**vidas = vv;**

**}**

**Meteoro::Meteoro(int xx, int yy): Coordenadas(xx, yy){}**

**Bala::Bala(int xx, int yy): Coordenadas(xx, yy){}**

**Coordenadas::~Coordenadas(){}**

**Personaje::~Personaje(){**

**limites();**

**gotoxy(37,17);cout<<"Todos los jugadores han sido eliminados";**

**}**

**Meteoro::~Meteoro(){}**

**Bala::~Bala(){}**

**void Personaje::COR(){**

**corazones--;**

**Beep(1500, 100);**

**}**

**void Personaje::pintar(){**

**ocultar();**

**gotoxy(x,y-1);printf("%c%c%c%c%c",200,203,167,203,188);**

**gotoxy(x,y);printf(" %c%c%c",200,206,188);**

**gotoxy(x,y+1);printf(" %c%c%c",201,202,187);**

**gotoxy(x,y+2);printf(" %c %c",207,207);**

**}**

**void Personaje::borrar(){**

**gotoxy(x,y-1); printf(" ");**

**gotoxy(x,y); printf(" ");**

**gotoxy(x,y+1); printf(" ");**

**gotoxy(x,y+2); printf(" ");**

**}**

**void Personaje::mover(){**

**if(kbhit()){**

**char tecla = getch();**

**borrar();**

**if(tecla == 'a' && x>3) x--;**

**if(tecla == 'd' && x+5<115) x++;**

**if(tecla == 'w' && y>5) y--;**

**if(tecla == 's' && y+5<35) y++;**

**pintar();**

**vida();**

**}**

**}**

**void Personaje::vida(){**

**gotoxy(90,1);printf("VIDAS REST:%d",vidas);**

**gotoxy(104,1);printf("SALUD:");**

**gotoxy(110,1);printf(" ");**

**for(int i = 0 ; i < corazones; i++){**

**gotoxy(110+i,1);printf("%c",3);**

**}**

**}**

**void Personaje::muerte(){**

**if(corazones == 0){**

**borrar();**

**gotoxy(x,y-1); printf(" \*\* ");**

**gotoxy(x,y); printf(" \*\*\*\* ");**

**gotoxy(x,y+1); printf(" \*\*\*\* ");**

**gotoxy(x,y+2); printf(" \*\* ");**

**Sleep(200);**

**borrar();**

**gotoxy(x,y-1); printf("\* \*\* \*");**

**gotoxy(x,y); printf(" \* \*\* \* ");**

**gotoxy(x,y+1); printf(" \* \*\* \* ");**

**gotoxy(x,y+2); printf("\* \*\* \*");**

**Beep(200, 500);**

**Sleep(200);**

**borrar();**

**vidas--;**

**corazones = 5;**

**vida();**

**pintar();**

**}**

**}**

**void Meteoro::crear(){**

**gotoxy(x,y);printf("%c",184);**

**}**

**void Meteoro::dispersion(){**

**gotoxy(x,y);printf(" ");**

**y++;**

**if (y>32){**

**x = rand()%110 + 4;**

**y = 4;**

**}**

**crear();**

**}**

**void Meteoro::colision(class Personaje &P1){**

**if( x>=P1.X() && x<P1.X()+5 && y>=P1.Y() && y<=P1.Y()+3){**

**P1.COR();**

**P1.borrar();**

**P1.pintar();**

**P1.vida();**

**x = rand()%110 + 4;**

**y = 4;**

**}**

**}**

**void Bala::trayectoria(){**

**gotoxy(x,y);printf(" ");**

**y--;**

**gotoxy(x,y);printf("%c",207);**

**}**

**bool Bala::choque(){**

**if(y==3)**

**return true;**

**return false;**

**}**

**int main(){**

**system("Title SolidWorld32");**

**system("COLOR 0D");**

**inicio();**

**system("cls");**

**limites();**

**gotoxy(16,9);cout<<"Gracias por probar mi proyecto, espero puedas ganara mas puntos la siguiente ronda :D\n";**

**gotoxy(37,10);cout<<"pulsa enter para continuar...";**

**getch();**

**return 0;**

**}**

**void inicio(){**

**bool repite = true;**

**int opcion;**

**Personaje P1(59,30,5,3);**

**cout<<endl<<endl<<endl<<endl<<endl<<endl<<endl<<endl;**

**cout<<" \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ "<<endl;**

**cout<<" | |"<<endl;**

**cout<<" | INSTITUTO POLITECNICO NACIONAL |"<<endl;**

**cout<<" | Escuela Superior de Ingeieria Mecanica y electronica |"<<endl;**

**cout<<" | Unidad Culhuacan |"<<endl;**

**cout<<" | Monroy Cortines Jacobo Emiliano |"<<endl;**

**cout<<" | Programacion orientada a objetos |"<<endl;**

**cout<<" | Proyecto Final |"<<endl;**

**cout<<" | Grupo: 2CV35 |"<<endl;**

**cout<<" | Profesor Osorino Soto Roberto |"<<endl;**

**cout<<" |\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|"<<endl<<endl;**

**cout<<"MENU DE INICIO\n"<<endl;**

**cout<<"\t1. Jugar"<<endl;**

**cout<<"\t2. Salir"<<endl;**

**do{**

**int n = 2;**

**opcion = retorno(n);**

**switch(opcion){**

**case 1:**

**int op;**

**op = 1;**

**while(op == 1){**

**system("cls");**

**Portada();**

**ocultar();**

**limites();**

**menu();**

**limites();**

**P1.pintar();**

**P1.vida();**

**movimiento();**

**Beep(250, 1000);**

**Beep(100, 1200);**

**Sleep(200);**

**system("cls");**

**limites();**

**gotoxy(37,10);cout<<"Deseas Reiniciar el juego?";**

**gotoxy(37,11);cout<<"1.si";**

**gotoxy(37,12);cout<<"2.no"<<endl;**

**gotoxy(37,13);cout<<"Digita dos veces el numero de la opcion que deseas: "<<endl;**

**getch();**

**gotoxy(37,14);cin>>op;**

**}**

**repite = false;**

**break;**

**case 2:**

**system("cls");**

**limites();**

**gotoxy(37,9);cout<<"Gracias por probar mi proyecto \n";**

**gotoxy(37,10);cout<<"pulsa enter para continuar...";**

**getch();**

**system("cls");**

**repite = false;**

**break;**

**}**

**}while(repite);**

**}**

**int retorno(int){**

**cout<<"\n\t\t\t\t Escoja la opcion que desea tomar";**

**cout<<"\n\t\t\t muevase entre las opciones con las teclas de arriba y abajo";**

**cout<<"\n\t\t\t\t para seleccionar una opcion, presione ENTER";**

**int n = 2;**

**bool repite2 = true;**

**int opcionseleccionada = 1;**

**int tecla;**

**do{**

**gotoxy(5, 20 + opcionseleccionada);cout<<" ";**

**gotoxy(5, 21 + opcionseleccionada);cout<<"==>";**

**gotoxy(5, 22 + opcionseleccionada);cout<<" ";**

**do{**

**tecla = getch();**

**}while(tecla != TECLA\_AR && tecla != TECLA\_AB && tecla != ENTER);**

**switch(tecla){**

**case TECLA\_AR:**

**opcionseleccionada--;**

**if(opcionseleccionada < 1){**

**opcionseleccionada = n;**

**}**

**break;**

**case TECLA\_AB:**

**if(opcionseleccionada > n-1){**

**opcionseleccionada = 0;**

**}**

**opcionseleccionada++;**

**break;**

**case ENTER:**

**repite2 = false;**

**break;**

**}**

**}while(repite2);**

**return opcionseleccionada;**

**}**

**void Portada(){**

**HANDLE consola=GetStdHandle(STD\_OUTPUT\_HANDLE);**

**char progreso=char(219);**

**char barra[17]={' ',' ',' ',' ',' ',' ',' ',' ',' ',' '**

**,' ',' ',' ',' ',' ',' ',0};**

**int tiempoFaltante=10;**

**for(int contador=0;contador<10;contador++){**

**limites();**

**barra[contador]=progreso;**

**#ifdef \_WIN32**

**Sleep(1000);**

**system("cls");**

**#else**

**sleep(1000);**

**clrscr();**

**#endif**

**tiempoFaltante--;**

**cout<<endl<<endl<<endl<<endl<<endl<<endl<<endl<<endl;**

**cout<<" \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ "<<endl;**

**cout<<" | |"<<endl;**

**cout<<" | Cargando videojuego |"<<endl;**

**cout<<" | Espere Porfavor |"<<endl;**

**cout<<" | El autor de este programa es: |"<<endl;**

**cout<<" | Monroy Cortines Jacobo Emiliano |"<<endl;**

**cout<<" |\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|"<<endl<<endl;**

**cout<<setw(40)<<"Faltan "<<tiempoFaltante<<" segundos para que desaparesca la portada"<<endl<<endl;**

**cout<<setw(70)<<barra<<endl;**

**if(tiempoFaltante==5){**

**#ifdef \_WIN32**

**system("cls");**

**#else**

**clrscr();**

**#endif**

**cout<<endl<<endl<<endl<<endl<<endl<<endl<<endl<<endl;**

**cout<<" \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ "<<endl;**

**cout<<" | |"<<endl;**

**cout<<" | Teclas de movimiento |"<<endl;**

**cout<<" | W = hacia arriba S = hacia abajo |"<<endl;**

**cout<<" | A = hacia la izquierda D = hacia la derecha |"<<endl;**

**cout<<" | Teclas Adicionales |"<<endl;**

**cout<<" | E = disparar |"<<endl;**

**cout<<" |\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|"<<endl<<endl;**

**cout<<" SOLIDWORK EL VIDEOJUEGO DE NAVES RETRO"<<endl;**

**cout<<setw(40)<<"Faltan "<<tiempoFaltante<<" segundos para que desaparesca la portada"<<endl<<endl;**

**cout<<setw(70)<<barra<<endl;**

**}**

**if(tiempoFaltante==4){**

**#ifdef \_WIN32**

**system("cls");**

**#else**

**clrscr();**

**#endif**

**cout<<endl<<endl<<endl<<endl<<endl<<endl<<endl<<endl;**

**cout<<" \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ "<<endl;**

**cout<<" | |"<<endl;**

**cout<<" | Teclas de movimiento |"<<endl;**

**cout<<" | W = hacia arriba S = hacia abajo |"<<endl;**

**cout<<" | A = hacia la izquierda D = hacia la derecha |"<<endl;**

**cout<<" | Teclas Adicionales |"<<endl;**

**cout<<" | E = disparar |"<<endl;**

**cout<<" |\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|"<<endl<<endl;**

**cout<<" SOLIDWORK EL VIDEOJUEGO DE NAVES RETRO"<<endl;**

**cout<<setw(40)<<"Faltan "<<tiempoFaltante<<" segundos para que desaparesca la portada"<<endl<<endl;**

**cout<<setw(70)<<barra<<endl;**

**}**

**if(tiempoFaltante==3){**

**#ifdef \_WIN32**

**system("cls");**

**#else**

**clrscr();**

**#endif**

**cout<<endl<<endl<<endl<<endl<<endl<<endl<<endl<<endl;**

**cout<<" \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ "<<endl;**

**cout<<" | |"<<endl;**

**cout<<" | Teclas de movimiento |"<<endl;**

**cout<<" | W = hacia arriba S = hacia abajo |"<<endl;**

**cout<<" | A = hacia la izquierda D = hacia la derecha |"<<endl;**

**cout<<" | Teclas Adicionales |"<<endl;**

**cout<<" | E = disparar |"<<endl;**

**cout<<" |\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|"<<endl<<endl;**

**cout<<" SOLIDWORK EL VIDEOJUEGO DE NAVES RETRO"<<endl;**

**cout<<setw(40)<<"Faltan "<<tiempoFaltante<<" segundos para que desaparesca la portada"<<endl<<endl;**

**cout<<setw(70)<<barra<<endl;**

**}**

**if(tiempoFaltante==2){**

**#ifdef \_WIN32**

**system("cls");**

**#else**

**clrscr();**

**#endif**

**cout<<endl<<endl<<endl<<endl<<endl<<endl<<endl<<endl;**

**cout<<" \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ "<<endl;**

**cout<<" | |"<<endl;**

**cout<<" | Los asteroides son mas |"<<endl;**

**cout<<" | faciles de esquivar si |"<<endl;**

**cout<<" | te mueves de izquiera a derecha |"<<endl;**

**cout<<" |\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|"<<endl<<endl;**

**cout<<" SOLIDWORK EL VIDEOJUEGO DE NAVES RETRO"<<endl;**

**cout<<setw(40)<<"Faltan "<<tiempoFaltante<<" segundos para que desaparesca la portada"<<endl<<endl;**

**cout<<setw(70)<<barra<<endl;**

**}**

**if(tiempoFaltante==1){**

**#ifdef \_WIN32**

**system("cls");**

**#else**

**clrscr();**

**#endif**

**cout<<endl<<endl<<endl<<endl<<endl<<endl<<endl<<endl;**

**cout<<" \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ "<<endl;**

**cout<<" | |"<<endl;**

**cout<<" | Los asteroides son mas |"<<endl;**

**cout<<" | faciles de esquivar si |"<<endl;**

**cout<<" | te mueves de izquiera a derecha |"<<endl;**

**cout<<" |\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|"<<endl<<endl;**

**cout<<" SOLIDWORK EL VIDEOJUEGO DE NAVES RETRO"<<endl;**

**cout<<setw(40)<<"Faltan "<<tiempoFaltante<<" segundos para que desaparesca la portada"<<endl<<endl;**

**cout<<setw(70)<<barra<<endl;**

**}**

**if(tiempoFaltante==0){**

**#ifdef \_WIN32**

**system("cls");**

**#else**

**clrscr();**

**#endif**

**cout<<endl<<endl<<endl<<endl<<endl<<endl<<endl<<endl;**

**cout<<" \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ "<<endl;**

**cout<<" | |"<<endl;**

**cout<<" | Preparate para triunfar |"<<endl;**

**cout<<" |\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|"<<endl<<endl;**

**cout<<" SOLIDWORK EL VIDEOJUEGO DE NAVES RETRO"<<endl;**

**cout<<setw(40)<<"Faltan "<<tiempoFaltante<<" segundos para que desaparesca la portada"<<endl<<endl;**

**cout<<setw(70)<<barra<<endl;**

**}**

**}**

**#ifdef \_WIN32**

**Sleep(1000);**

**system("cls");**

**#else**

**sleep(1000);**

**clrscr();**

**#endif**

**}**

**void menu(){**

**limites();**

**gotoxy(37,9);cout<<"En un futuro lejano, durante 6 millones de lustros\n";**

**gotoxy(37,10);cout<<"pulsa enter para continuar\n";**

**getch();**

**gotoxy(34,10);cout<<"el planeta tierra fue raptado hacia otro sistema solar\n";**

**gotoxy(35,11);cout<<"hacia otro sistema que esta a punto de ser colapsado\n";**

**gotoxy(39,12);cout<<"pulsa enter para continuar\n";**

**getch();**

**gotoxy(38,12);cout<<"un mundo hacia el futuro, la paz queda incierta\n";**

**gotoxy(37,13);cout<<"pulsa enter para continuar\n";**

**getch();**

**gotoxy(32,13);cout<<"los mejores soldados son reunidos para protejer un legado\n";**

**gotoxy(32,14);cout<<"donde los humanos luchamos para permanecer en el universo\n";**

**gotoxy(37,15);cout<<"pulsa enter para continuar\n";**

**getch();**

**gotoxy(32,15);cout<<"un equipo de 4 personas intentan salvar a la raza humana\n";**

**gotoxy(39,16);cout<<"su unico objetivo es cazar a los tetradimons\n";**

**gotoxy(37,17);cout<<"pulsa enter para continuar\n";**

**getch();**

**gotoxy(36,17);cout<<"para asi poder conseguir el nucleo de dimensiones\n";**

**gotoxy(35,18);cout<<"y llevar a la raza humana de nuevo al sistema solar\n";**

**getch();**

**system("cls");**

**limites();**

**gotoxy(50,10);cout<<"Teclas de movimiento\n";**

**gotoxy(25,11);cout<<"W = hacia arriba S = hacia abajo A = hacia la izquierda D = hacia la derecha\n";**

**gotoxy(51,12);cout<<"Teclas Adicionales\n";**

**gotoxy(54,13);cout<<"E = disparar\n";**

**gotoxy(44,14);cout<<"pulsa enter para continuar...\n";**

**getch();**

**system("cls");**

**}**

**void movimiento(){**

**Personaje P1(59,30,5,3);**

**Meteoro M1(10,5);**

**Meteoro M2(30,20);**

**Meteoro M3(50,9);**

**Meteoro M4(65,8);**

**Meteoro M5(70,14);**

**list<Meteoro\*> A;**

**list<Meteoro\*>::iterator itA;**

**list<Bala\*> B;**

**list<Bala\*>::iterator it;**

**for(int i=0; i<5; i++){**

**A.push\_back(new Meteoro(rand()%75+3, rand()%5+4));**

**}**

**bool game\_over = false;**

**int puntos = 0;**

**while(!game\_over){**

**gotoxy(4,1);printf("PUNTOS: %d", puntos);**

**if(kbhit()){**

**char tecla = getch();**

**if(tecla == 'e')**

**B.push\_back(new Bala(P1.X()+2, P1.Y()-1));**

**}**

**for(it = B.begin(); it != B.end(); it++){**

**(\*it)->trayectoria();**

**if((\*it)->choque()){**

**gotoxy((\*it)->X(), (\*it)->Y()); printf(" ");**

**delete(\*it);**

**it = B.erase(it);**

**}**

**}**

**M1.dispersion(); M1.colision(P1);**

**M2.dispersion(); M2.colision(P1);**

**M3.dispersion(); M3.colision(P1);**

**M4.dispersion(); M4.colision(P1);**

**M5.dispersion(); M5.colision(P1);**

**for(itA=A.begin(); itA!=A.end(); itA++){**

**(\*itA)->dispersion();**

**(\*itA)->colision(P1);**

**}**

**for(itA=A.begin(); itA!=A.end(); itA++){**

**for(it=B.begin(); it!=B.end(); it++){**

**if((\*itA)->X() == (\*it)->X() && ((\*itA)->Y()+1==(\*it)->Y() || (\*itA)->Y()==(\*it)->Y() )){**

**gotoxy((\*it)->X(),(\*it)->Y());printf(" ");**

**delete(\*it);**

**it = B.erase(it);**

**A.push\_back(new Meteoro(rand()%75+3, 4));**

**gotoxy((\*itA)->X(),(\*itA)->Y());printf(" ");**

**delete(\*itA);**

**itA = A.erase(itA);**

**puntos+=5;**

**}**

**}**

**}**

**if(P1.VID() == 0) game\_over = true;**

**Sleep(30);**

**P1.muerte();**

**P1.mover();**

**}**

**}**

**void ocultar(){**

**HANDLE hcon;**

**hcon = GetStdHandle(STD\_OUTPUT\_HANDLE);**

**CONSOLE\_CURSOR\_INFO cci;**

**cci.dwSize = 2;**

**cci.bVisible = FALSE;**

**SetConsoleCursorInfo(hcon,&cci);**

**}**

**void limites(){**

**for(int i = 2 ; i<118 ; i++){**

**gotoxy(i,2);printf("%c",205);**

**gotoxy(i,33);printf("%c",205);**

**}**

**for(int i = 3 ; i<33 ; i++){**

**gotoxy(2,i);printf("%c",186);**

**gotoxy(118,i);printf("%c",186);**

**}**

**gotoxy(2,2);printf("%c",201);**

**gotoxy(2,33);printf("%c",200);**

**gotoxy(118,2);printf("%c",187);**

**gotoxy(118,33);printf("%c",188);**

**}**

**void gotoxy(int x, int y){**

**HANDLE hcon = GetStdHandle(STD\_OUTPUT\_HANDLE);**

**COORD dwPos;**

**dwPos.X = x;**

**dwPos.Y = y;**

**SetConsoleCursorPosition(hcon, dwPos);}**

**Mantenimiento**para el mantenimiento de este programa simplemente no se debe intervenir en el código fuente.

En caso de actualización se requiere supervisión de una persona que maneje el AOO, DOO, LOO y POO.

El programa solo corre en sistemas operativos con soporte en lenguaje C y C++.

El autor de este programa no es removido de su cargo bajo ninguna circunstancia.

**Conclusión**

Puedo deducir que este trabajo fue tedioso, sin embargo, fue un trabajo que disfruté muchísimo en su elaboración, en el que me divertí y aprendí respecto a la elaboración de programas en C++. A pesar del poco conocimiento previo que tengo en lenguaje C y C#, uno se da cuenta que no deja de aprender cosas nuevas cada día, mis más grandes agradecimientos a el profesor por todo el aprendizaje que tuve sobre alrededor del semestre, a mis compañeros por todo lo que aprendí de ellos, auténticamente me agrado la materia y deseo tener seguimiento de este proyecto a posteriores grados.

Por último, vi como la POO fue evolucionando drásticamente en diferentes formas de implementación, puedo concluir que domino el tema un 85%, aunque nunca dejamos de aprender.

**Bibliografía**

<https://www.youtube.com/watch?v=dJzLmjSJc2c&list=PLWtYZ2ejMVJlUu1rEHLC0i_oibctkl0Vh>

<https://www.youtube.com/watch?v=X6lMTEIzjys&list=PLpOqH6AE0tNj8W0EGpoGG2CEMDJTt4ihZ>

<https://classroom.google.com/u/0/c/NTM4NTQ3Nzg1MzAx/m/NTI2NjEyMTA1MjYw/details?hl=es>