Centro de Estudios de Postgrado

Máster del profesorado

UNIVERSIDAD DE JAÉN
Centro de Estudios de Postgrado

Trabajo Fin de Máster

PROGRAMACIÓN ESTRUCTURADA APLICADA AL AJEDREZ

Alumno/a: **Díaz Catena, Antonio**

Tutor/a: Rafael Jesús Segura Sánchez y Francisco Feito Higueruela
Dpto: Informática

Julio, 2015

ÍNDICE

DEDICADO A MI PADRE, POR SER MI EJEMPLO EN LA DOCENCIA Y ENSEÑARME A JUGAR AL AJEDREZ

A MI MADRE, POR DEJARME EL COCHE PARA IR A CURSAR EL MÁSTER A JAÉN

A MI HERMANO, POR LAS PELEAS A LA HORA DE JUEGAR AL DOOM II EN UN VIEJO 133 MHZ

Y A ISABEL

RESUMEN

El presente trabajo, está desarrollado en torno a la programación informática estructurada, dentro del módulo programación del ciclo formativo "Grado superior Técnico de Desarrollo de Aplicaciones Multiplataforma". Se compone de:

- Fundamentación teórica y epistemológica: desarrollo de la fundamentación de los motores de videojuegos, sus librerías, entornos de desarrollo específicos, funciones así como componentes. Además de los aspectos técnicos del motor de juego del ajedrez.
- La segunda parte estará compuesta por: el tratamiento de una unidad didáctica; en la que presentaremos como abordar de una manera clara y efectiva la programación estructurada a través de unos ejercicios planteados entorno al ajedrez.

En el desarrollo de la unidad didáctica podremos encontrar objetivos, actividades y resultados de aprendizaje que el alumno superará a lo largo de ella.

Palabras Clave:

AJEDREZ, PROGRAMACIÓN ESTRUCTURADA, ENSEÑANZA, FORMACIÓN PROFESIONAL.

SUMMARY

This work is developed around computer programming module structured training cycle Technical Development platform applications and it contains:

- Theoretical and epistemological foundation. Throughout this section I will develop the foundation of the game engines and their libraries, development environments, specific functions and components. Besides the technical aspects of the game of chess engine .
- The second part will consist of the development of a teaching unit in which present and clearly and effectively structured programming unity. In which the exercises are presented around chess.

In the development of the teaching unit we can find all the objectives, activities and learning outcomes the student surpass along it .

***Key Words*:** CHESS, STRUCTURED PROGRAMMING, PROFESIOANL FORMATION.

Introducción.

El presente trabajo, nace como consecuencia del tiempo dedicado al Máster Universitario en Profesorado de Educación Secundaria Obligatoria y Bachillerato, Formación Profesional y Enseñanza de Idiomas.

A lo largo de este documento, voy a poner en práctica todos los conceptos absorbidos a lo largo del curso. Dando un toque de frescor con la presente unidad didáctica.

A través de esta, introduciré una serie de ejercicios prácticos para que el alumnado pueda adentrarse en el mundo de la programación sin dificultad alguna. El ajedrez, como disciplina, requiere capacidad de análisis y cálculo. Este nos servirá de guía para visualizar programas en los que está presente la programación estructurada.

FUNDAMENTACIÓN ESPITEMÓLÓGICA

Motor de juego

Es un entorno de programación específico que ofrece una serie de rutinas, las cuales, permiten el diseño de un videojuego[1]. La función principal de esta plataforma es la de proveer al usuario de la misma, de unas herramientas con las que pueda controlar el audio, animación, detectar posibles colisiones físicas, scripting. Así como cada una de las características importantes del juego.

A día de hoy, existen una gran cantidad de motores. Siendo la mayoría de ellos "open-source". A través de un lenguaje de programación (como pueden ser C#, C++ o Java) el usuario del motor puede crear aplicaciones en la misma.

Librerías

Como en la mayoría de proyectos de desarrollo software, la industria de los videojuegos se suele apoyar en una serie de librerías o bibliotecas con funciones ya existentes. Si bien, muchos desarrolladores prefieren adaptarlas a sus circunstancias específicas. Las más importantes que están presentes en este campo son[1]:

- STL (Standar Template Library). Se utiliza en el lenguaje C++ debido a su fácil portabilidad y eficiencia.
- APIs(Application Programming Interface) Gráficas. OpenGL y Direct 3D, ideales para moldear gráficos en 3D. Aportan soluciones comunes como puede ser el renderizado

de modelos tridimensionales. Además ocultan ciertos puntos de la tarjeta gráfica presentando al desarrollador una API común.

- Librerías para detección y proceso de las colisiones de los entes que forman parte del juego, ideales para la gestión física del juego. Las más importantes: Havok Physics, CryEGINE o Unreal.
- Algoritmos de Inteligencia artificial. Para calcular de manera rápida y efectiva el camino optimo entre puntos, por ejemplo.

Componentes y funciones de un motor de juego

Los más importantes son[1]:

- *Gestor de recursos.* Permite al usuario del motor llegar a las distintas entidades software que lo compone. Además, suministra una interfaz única para acceder a cada una de ellas.
- *Motor de rendering.* Es uno de los componentes más complejos, puesto que utiliza una estructura multicapa. Además se encargada de generar una imagen a partir de un modelo 3D. Las capas que lo componen son las siguientes:

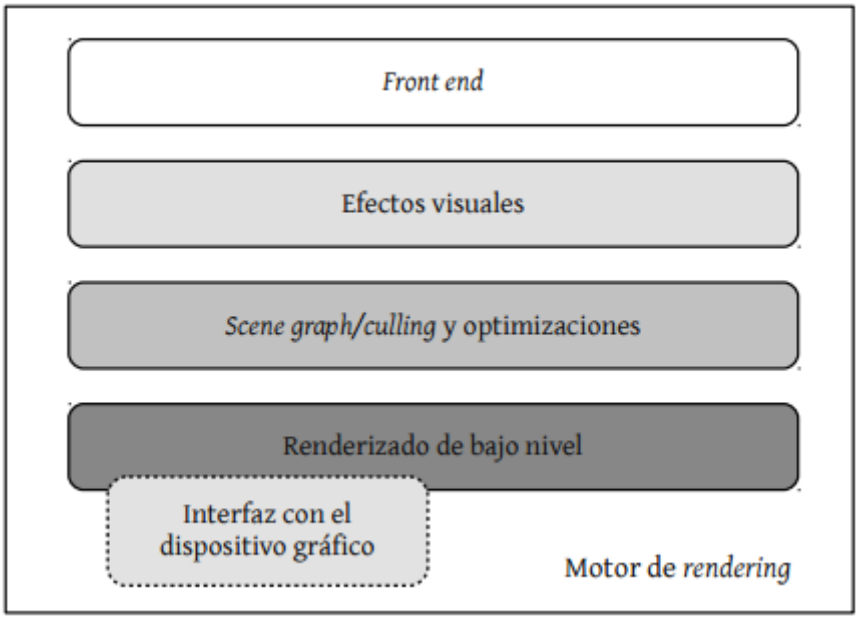


Figura 1. Capas Motor Rendering. Fuente: Desarrollo de Videojuegos: Arquitectura del Motor de Videojuegos

1. *Renderizado de bajo nivel:* Es la encargada de renderizar, lo más rápido posible, las distintas primitivas geométricas de una capa. Además es la encargada de interactuar con las diferentes APIs, como puede ser el OpenGl o Direct3D.

2. *Scene grap/cullin y optimizaciones.* Determina que partes de la escena se mandan a la capa de rendering. Esta distinción es necesaria para obtener una optimización del rendimiento del motor rendering.
3. *Efectos visuales.* Constituye una serie de módulos que son los responsables de crearlos en los juegos.
4. *Front End.* Sin ella no se podría superposicionar el contenido 2D con el 3D.

- *Herramientas de depuración.* Resultan indispensables para contralar todos y cada uno de los errores que pueda tener el juego. Puesto que la mayoría de ellos funcionan en tiempo real, además tenemos que sacar el mayor rendimiento posible. Entre las más importantes caben destacar:

1. Mecanismos de control para medir el tiempo empleado en la ejecución del código.
2. Medición en tiempo real y de manera gráfica del rendimiento del juego mientras se ejecuta.
3. Volcado de logs (registros) en ficheros de texto.
4. Vistas gráficas para controlar el consumo de memoria de cada parte del motor del juego.
5. Herramientas para la depuración de la información generada.
6. Herramientas para grabar el juego en ejecución. Indispensables para depurar bugs.

- *Motor de Física.* Durante el desarrollo normal del videojuego es necesario que los objetos no choquen entre sí. Además, si se traspasan uno con otros no se podría desarrollar el juego normalmente. Este sistema está compuesto por:

1. *Detección de colisión.* Devuelve una variable lógica y dice si ha habido.
2. *Determinación de la colisión.* Calcula el punto exacto donde se ha producido.
3. *Respuesta a la colisión.* Determinar cómo actuar ante una.

- *Interfaces de usuario.* Encargadas de interactuar y de procesar dicha interacción con el usuario final del videojuego, asumen la entrada de todos los eventos. También, le suministra todos los eventos de salida, por lo que se produce una interacción entre

máquina y jugador. Por tanto, se le suele denominar comúnmente componente de entrada/salida del jugador.

- *Módulo multijugador.* El motor de juego debe de dar soporte a esta posibilidad de juego, ya que la mayoría de los motores tratan la opción single player como una opción del módulo multijugador.

- *Módulo Networking.* Es el encargado de informar al jugador del desarrollo del juego. Para ello se utilizan los sockets (paquetes de información). Un ejemplo sería la posición donde se encuentra el jugador en cada instante.

- *Subsistema de juego o game play.* Contiene todas las funcionalidades principales del videojuego, así como las leyes internas que lo gobiernan. Permite la interacción de los diferentes personajes que actúan en él, así como definir las metas que tiene cada uno a lo largo de la partida. La importancia de este subsistema reside en que separa la lógica normal del juego del código subyacente, por tanto, está presente un sistema scripting para definir la conducta de los actores del juego. Entre los componentes del mismo, se encuentra el sistema de eventos que permite la comunicación entre objetos de diferentes clases. Está formado por:

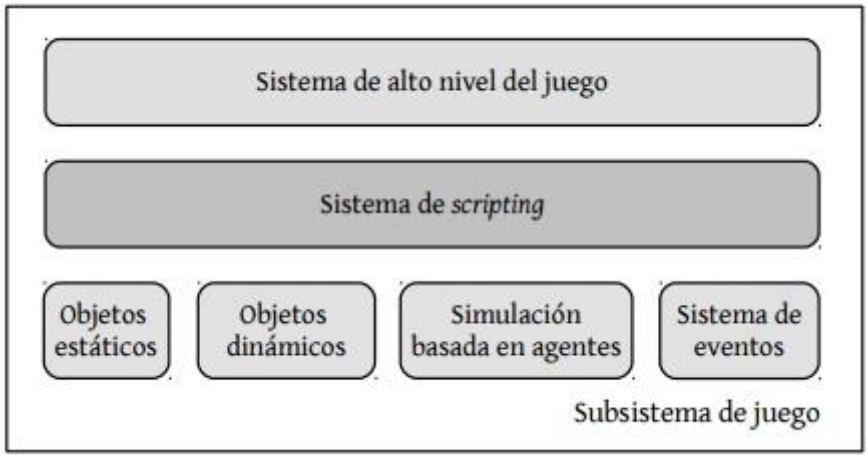


Figura 2. Componentes del subsistema de juego. Fuente: Desarrollo de Videojuegos: Arquitectura del Motor de Videojuegos

- *Audio.* Los motores de audio han ido cobrando más fuerza dentro de esta industria. Sin duda alguna ha venido de la mano del desarrollo de los nuevos formatos.

- *Subsistemas específicos del juego.* Se encuentra encima de la capa subsistema del juego y del resto, debido a que es la encargada de ofrecer las características propias del juego. Dependiendo de las características del juego tendrá un mayor número de capas.

Entornos de desarrollo específicos de videojuegos.

Desde Open Source, hasta plataformas de pago, el catálogo de Entornos de desarrollo específicos de videojuegos es muy extenso. Para este trabajo fin de máster, me he decantado por analizar los motores expuestos a continuación, puesto que son los que más repercusión tienen en la industria.

Doom Engine

La revolución en este campo de la informática la trajo consigo la liberación del mítico juego DOOM que fue el primero en sacar bajo licencia GNU su motor. Es aquí donde nace este concepto[1]. El creador de esta mítica saga de videojuegos, John Carmack, pensó que la mejor forma de seguir trabajando en esta saga era la de descomponer el juego en diferentes módulos. Creando así una arquitectura orientada a la reutilización, dividida por un lado en el sistema de renderizado gráfico y por otras el sistema de detección de colisiones y el de audio.

Su interfaz gráfica sirvió para poner en marcha cientos de versiones del juego. Sin duda alguna Doom Engine (nombre del motor) sentó las bases para el desarrollo de esta industria. La mayoría de los juegos son reciclados a partir de versiones anteriores, puesto que ahorramos coste depurando y mejorando la versión anterior existente[1]. Cabe destacar que no es un motor 3D puesto que solo tiene la opción de movernos de izquierda a derecha, en ningún momento ni hacia arriba o abajo, debido al mapeado por texturas.

Lo objetos son reproducidos en este motor a través de su unidad básica el vertex (vértices), que presentan un punto del mapa 2D. Estos vértices se van confluyendo uno con otros dando lugar a líneas (linedefs)[2]. Entre los objetos que se imprimen por pantalla podemos encontrar[2]:

- *Sidedefs.* Lados de los linedefs. En cada línea podemos encontrar uno o dos lados, dando lugar a estos. Se utilizan para situar texturas en las paredes del juego.
- *Sectors (Secotres).* Agrupación de sidedefs. Generan polígonos y son los encargados de producir la altura del techo, la longitud del suelo o la iluminación.
- *Things (Cosas).* Al final encontramos una serie de librerías que contiene una lista de cosas como monstruos o personajes y sirven para situarlos en el mapa, dentro de una coordenada 2D.

En cuanto al renderizado, debemos destacar los siguientes aspectos[2]:

- *Trazado de paredes.* Se nutre del sistema BSP (particionado binario del espacio) por el cual se plasman las paredes en función de la distancia en la que se encuentra. Primero se trazan los segs (línea que separa a los "linedefs") y se guardan en una lista de enlazado para unirlos más tarde, conforme se va avanzado.
- *Suelo y techo.* Son dibujados como visplanos, es decir, la textura del suelo y techo se introducen horizontalmente (coordenada x) y le corresponde una coordenada y que lleva una textura para que sea dibujada.
- *Sprites.* Conforme se desarrolla un sector, en él podemos descubrir cosas que se encuentran guardadas en una lista. Si en nuestro campo de visión se encuentran alguna de estos sprites se dibujarán, sino, se ignoran.

Unreal Engine

Programado por Epic Games en el lenguaje C++. Unreal Engine, es uno de los motores fetiche para desarrollar shooter (juego de disparos en primera persona). Está pensado para utilizar tecnología OpengL y crear juegos en diferentes plataformas. Ha sido responsable de los juegos shooter más vendidos del mercado como Unreal Tournament, Deus Ex, Turok, Tom Clancy's Rainbow Six: Vegas, America's Army, Red Steel, Gears of War, BioShock, BioShock 2, BioShock Infinite, Star Wars Republic Commando, Batman: Arkham Asylum o Mass Effect.

La primera versión de este motor ve la luz en el año 1998. Entre sus características principales integradas estaban la de renderizado, inteligencia artificial, detección de colisiones, visibilidad y opciones para redes.

En cuanto a la segunda versión del motor hace aparición de la mano de America's Army en el año 2002. Este juego fue desarrollado bajo la segunda versión de él. La regeneración del mismo pasó por la reescritura del motor del núcleo y del renderizado, además de otras mejoras, que hicieron el motor de juego compatible con Play Station 2, Xbox y Game Cube[3].

La tercera generación (2006) trae consigo el soporte para el normal mapping y sombras dinámicas. Es diseñado para PC y da soporte DirectX 9/10, Xbox 360 y PlayStation 3. Entre las mejoras que aportó esta versión destacan renderizado para un mayor número de objetos simultáneos, físicas más realistas para efectos de agua y de texturas corporales, mayor destructibilidad para los entornos, IA mejorada y las luces, sombras con rutinas avanzadas para los shaders. Todas estas adelantos hacen que el juego no sea concebido exclusivamente para la industria de los videojuegos, también, se empieza a aplicar como simulador de conducción, construcción, interiores o generación de construcción[3].

La última versión de Unreal, la cuarta, sale en el año 2011 y empieza a ser un motor gratuito. Entre sus características más importantes podemos destacar[4]:

- DirectX 11 y 12. Este motor de renderizado, permite la creación de escenas en full HDR, miles de luces dinámicas por escena, además de sombreado base de elementos físicos y materiales.
- Efectos Visuales de cascada. Permitiendo la representación de fuego, aire, agua o humo. Su unidad de procesamiento gráfico permite la simulación de partículas y un sistema de colisión que interactúa con el buffer en profundidad.
- C++. El código fuente es muy accesible permitiendo navegar por las funciones de cada personaje con total facilidad. Acceso total.
- Unreal Engine 4's Sound Cue. Motor de audio.
- Vista previa del juego instantánea.
- Navegador de contenido. Permitiendo un acceso instantáneo a cada una de las funcionalidades del juego.

Unity3D

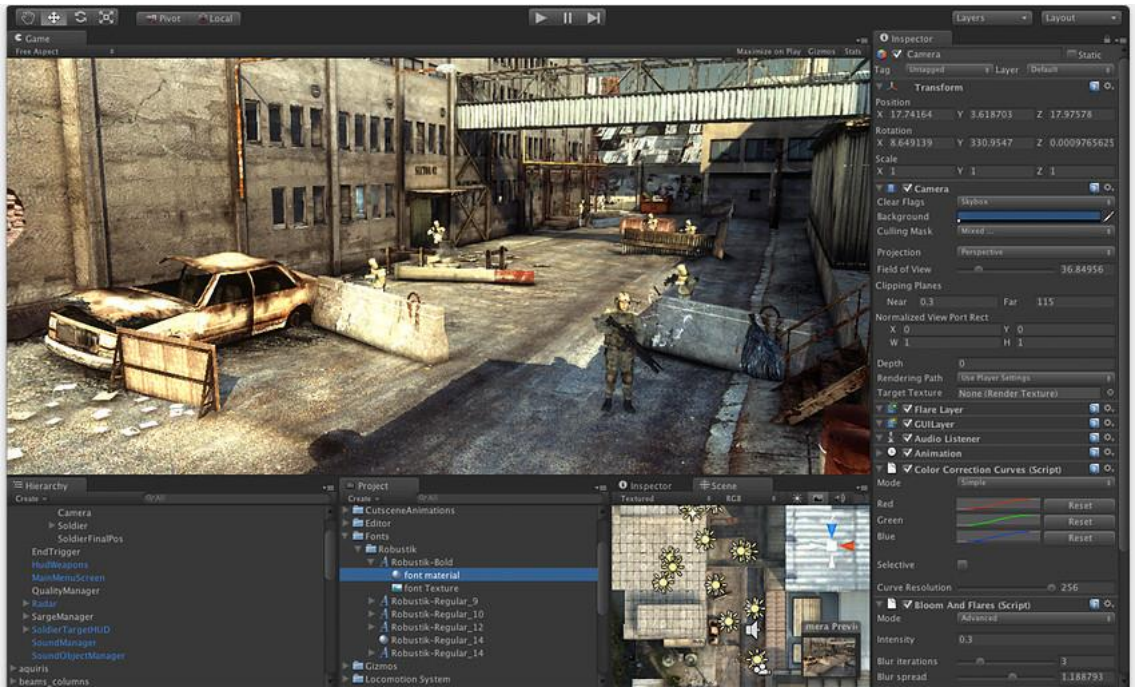


Figura 3. Captura Pantalla Unity3D . Fuente : alittlebigof.wordpress.com

Es uno de los motores más potentes del mercado. Primero porque permite crear juegos para una cran cantidad de plataformas como Windows, OS X, Linux, Xbox 360, PlayStation 3, Playstation Vita, Wii, Wii U, iPad, iPhone, Android y Windows Phone. Y además está construido sobre el lenguaje C++. Nace de la idea de crear un motor gráfico específico para sistemas MAC aunque al final se ha adaptado a todos los sistemas operativos.

La primera versión importante de este motor de videojuegos es la 3.5 y entre las características más importantes destacan[5]:

- Sistema de partículas Shuriken
- Navmesh para *pathfinding* y evasión de obstáculos
- Iluminación del espacio líneal (gamma correcta)
- Renderización HDR
- Renderización multihilo
- Sondas de luz
- Despliegue Google Native Client.
- Reescrito desecho oclusión
- Una función de nivel de detalle apoyo.
- *Sddon* de Adobe Flash Player para vista previa
- Perfilador de GPU
- L*ightmaps* direccionales.

En cuanto a la versión 4 se presenta en el año 2012 y presenta las siguientes mejoras[6]:

- Sistema de partículas Shuriken soporta fuerzas externas, normales de Bent y eliminación automática.
- Soporte texturas 3D
- Navegación: obstáculos dinámicos y prioridad de evasión.
- Optimizaciones importantes en el rendimiento y uso de memoria de UnityGUI.
- Fuentes dinámicas en todas las plataformas con HTML como marcado.
- Depuración remota de Unity Web Player.
- Nuevos flujos de trabajo en la ventana de proyecto.
- Mapa iterativo de lightmap.
- Componentes basados en refinados de flujos de trabajo.
- Inspectores extensible para clases personalizadas.
- Mejorado el pipeline de importación de Cubemap.

- Mejoras en datos geométricos para una memoria enorme y ahorro en rendimiento.
- Las mallas se pueden construir a partir de figuras geométricas no-triangulares para hacer puntos y líneas eficientemente.
- Búsqueda, vista previa en vivo y compra de Assets del Asset Store desde la ventana del proyecto.
- Unity ad-on para adobe flash player
- Mejoras gráficas para móviles.
- Direcx 11.

La última versión existente de Unity, la cinco, ha sido lanzada en este mismo año. La gran ventaja que es permite integrar múltiples plataformas desde el mismo entorno de desarrollo integrado.

Entre las características más importantes que tiene esta nueva versión el motor de juego Unity 5 se encuentran[6]:

- Genrelaes
    - Scripting con C#, JavaScrip o Boo.dsfDespliegue con un clic.
    - Editor de 64 bits.
    - Física cargada de acción.
    - Gráficos optimizados.
    - Editor totalmente extensible.
    - Animación que parece real.

- Gráficos
    - Shandig físico.
    - Acceso a rendering de bajo nivel.
    - Efectos render a textura.
    - Sistema de partículas Shurken
    - Fuentes dimáicas con Markup
    - Efectos de procesamiento de pantalla completa
    - Realtime Global Illumination de Engluhten
    - Static Banching

- 2D
    - Sprite slpicing automatic
    - Empaquetador de sprites.
    - Animación automáticas de srpites.
    - Física 2D
    - Alpha cutout

- Física 3D
    - Simulación multihebra
    - Detección de choques séper precisa.
    - Componente de ropa
    - Soporte para MeshCollider a escala sin bakeado
    - Física avanzada para vechículos.

- Optimización
    - Occlusion Culling
    - Deferred rendering
    - Streaming en todo el sentido de la palabra Asset Bundles
    - Profiler
    - Stencil buffer Access
    - Dynamic Batching
    - Nivel de detalle
    - GPU Skinning para directX 11 y Open Gl ES 3.0

- Scripting
    - Depuración del reproductor web.
    - Acceso a los datos de la web a través de las funciones WWW
    - Acceso de scripts al pipeline de activos
    - Soporte .NET Socket
    - Abre una URL en el Navegador del usuario
    - Inspector GUI para clases personalizadas
    - Soporte de plugin de código nátivo.

- Audio
    - Logro de la transición del estado de ánima de su paisaje sonoro.
    - Invoque scripts desde dentro de la reproducción de animación
    - Plugins de audio nativo
    - Jerarquías de mezcladores.

- Otras características
    - Soportes externo de control de versiones
    - Networking con RakNet para multiples jugadores
    - Generadores de informes de error para iOS
    - Terrenos
    - NavMeshes y Path-Finding
    - Reproductor Linux sin procesador
    - Soporte integrado para SpeedTree
    - Net Streaming (No soportado en iOS)
    - Prueba de juegos de manera instantánea.

CryEngine

Este motor es uno de los más potentes del mercado. Programado en C++ y Lua es capaz de adaptarse a diferentes plataformas como Xbox, Wii, iOS, Windows o Play Station. Consta de tres versiones.

CryENGINE 1 fue desarrollado entre 2001 y 2004 por la empresa alemana Crytek, en paralelo a primer juego de la misma: Far Cry. Fue el primer motor en utilizar "píxel por sombreado" (luces, sombras y materiales) siendo por esto considerado el primer motor de "alto rango dinámico" del mercado. Los logros más importantes realizados con CryENGINE de Crytek fueron el editor Sandbox, la iluminación en tiempo real, las largas distancias de vista (así como muy detalladas y de alta calidad), los primeros planos, las vegetaciones realistas, la física de alta fidelidad y la inteligencia artificial[7].

Características del motor[7]:

- El editor Sandbox es un editor del juego en tiempo real. “Lo que ves es lo que juegas"

- CryENGINE 1 utiliza un sistema de escritura para los shaders, combinando texturas de diferentes formas lo que produce gran variedad de efectos visuales. Soporta en tiempo real: la iluminación de píxeles, reflexiones desiguales, refracciones, efectos de brillo volumétricos, texturas animadas, y superficies brillantes.

- Para visualizar terreno, CryENGINE 1 utiliza un sistema de alturas avanzado además de reducción de polígonos para crear entornos masivos y realistas con vistas distancias de hasta 2 km.

- El Sistema de la física del motor soporta cinemáticas inversas de personajes, vehículos, cuerpos rígidos, líquidos, telas y efectos suaves del cuerpo. La combinación de sombras pre calculadas en tiempo real, sombras plantilla y lightmaps se utiliza para producir un entorno dinámico. Incluye alta resolución, perspectiva correcta y las implementaciones de sombras suaves volumétricas para sombreado de cubierta dramática y realista. También es compatible con tecnología de partículas avanzada y todo tipo de efectos de iluminación volumétrica de partículas.

- El sistema de inteligencia artificial del CryENGINE ofrece la posibilidad de crear amigos, enemigos y comportamientos personalizados sin tocar el código C ++.

La segunda versión de este motor de juego también fue creada por Crytek. Este entorno de desarrollo permitía la edición en tiempo real, bump mapping, creación luces dinámicas, sistema de red, sistema motor de física o shaders Además de su gran arma, Sandbox[8].

Aquí es donde entra en juego Polybump 2, puesto que crea una descripción superficial de alta calidad que permite la extracción rápida de las características superficiales, ideal para los mapas normales, mapas de desplazamiento y la no inclusión en la dirección de área. La información extraída puede ser utilizada para hacer modelos de detalle de la superficie. Después, los datos se almacenan en un formato de archivo intermedio que se pueden exportar de diferentes maneras. Esto trajo consigo que se procesaran millones de polígonos a gran velocidad[8].

En cuanto a la representación en tiempo real CryENGINE 2 proporciona ambientes de interior y al aire libre en DirectX9 y 10, así como el apoyo a las consolas como la Xbox 360 y PlayStation 3.

El sistema de sonido en el CryEngine 2 introdujo nuevas características y mejoras. Cada sonido lleva su propia especificación con él, por lo que los diseñadores de sonido tienen el control total de la calidad final y lo utilizan constantemente a lo largo del juego. Entre sus características más importantes[8]:

- Mezcla durante el juego
- Data Drive Sound System
- Sistema Interactivo Dinámico de Música
- Audio Ambiental
- Dinámica de Sonidos , dependiendo del mundo
- Sistema AI / modular avanzado AI

CryENGINE 2 posee un sistema de inteligencia artificial flexible y fácilmente personalizable que se ocupa de los comportamientos de personajes y vehículos. Es totalmente compatible con los complejos requisitos del sistema de locomoción de personajes bípedos y está totalmente integrado en el editor de CryENGINE Sandbox 2. Dentro del sistema de inteligencia artificial debemos destacar[8]:

- Lenguaje de programación LUA.
- Búsqueda dinámica de ruta.
- Objetos inteligentes

CryENGINE 3 fue el primer editor del mundo con tecnología YSIWYP. Crytek 3 consiguió mejorar e integrar por completo el editor Sandbox´. Además CryENGINE 3 trae consigo la funcionalidad YSIWYP a un nivel completamente nuevo, lo que supuso su integración por completo en PlayStation 3 y Xbox 360.

El nuevo motor gráfico de la tecnología Crytek, desarrolla sin fisuras algunos ambientes de interior y al aire libre, debido a su adaptación multi núcleo. CryENGINE 3 es el procesador más rápido del mundo de gama alta. Sus características hacen de él el motor ideal para el desarrollo de juegos en video consola. Las más importantes son las siguientes[8]:

- Flujo gráfico. El entorno de desarrollo integrado provee a los diseñadores una intuitiva interfaz gráfica para controlar hilo, triggers y lógica de juego.
- Vegetación integrada y sistema de generación de terreno. Este mismo se comporta de acuerdo a las leyes de la naturaleza.
- Sistema de partículas en tiempo real. Simplificando la creación de explosiones, fuego, humo o lluvia pudiendo interactuar estas con otros elementos del entorno.

- Herramientas para integrar carreteras, ríos o caminos, permitiendo el nivelado del terreno con toda facilidad.
- Creación de vehículos. Incluyendo daño de vehículo, posición de los pasajeros o armas. Además se le pueden integrar parámetros de física o efectos.
- Soporte Multicore. Permitiendo exprimir al máximo la tecnología multinucleo de los nuevos ordenadores.
- Iluminación global en tiempo real. Sin tener que hacer dichosos cálculos geométricos puesto que está unificado para todos los objetos dinámicos y estáticos.
- Iluminación en diferido. Permite la prestación de una gran cantidad de fuentes de luz con sombreado de píxeles de manera eficiente.
- Iluminación natural. Creando sombras naturales que se corresponden con el movimiento en tiempo real.
- Capa volumétrica de nebulización. Permitiendo la creación de nubes, humo, gases o bancos de niebla. Interactuando de manera realista con la visibilidad.
- Pristine Motion Blur. Quizás la tecnología más innovadora. La profundidad de los efectos de campo se puede definir fácilmente con el editor WYSIWYP multiplataforma. CryENGINE 3 ahora implementa a la más alta calidad estos efectos a un costo bajo rendimiento.

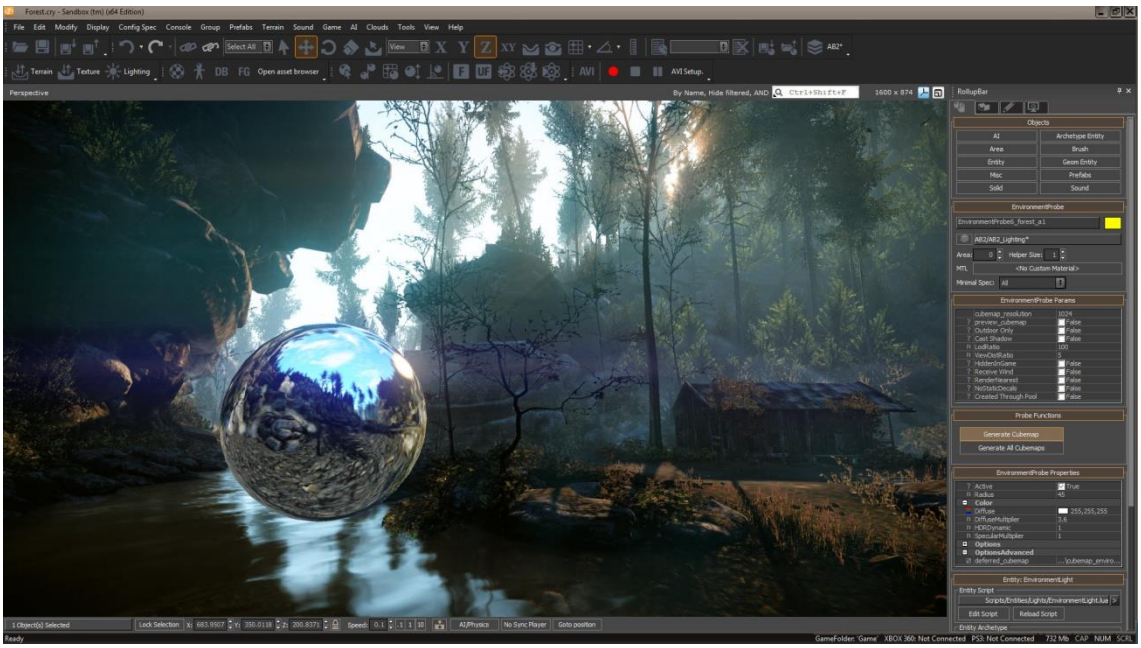


Figura 4. Motor CryENGINE 3. Fuente: http://www.pivotstudio.co/

Otra de las apuestas de Crytek ha sido la representación de los personajes del juego. Durante la partida nos tenemos que sentir identificado con el personaje. Así el motor CryENGINE 3 apuesta por el desarrollo personalizado de cada uno de ellos. Las herramientas utilizadas para ello son las siguientes[8]:

- Sistema de animación de personajes. Este mismo avanza considerablemente en el estado del arte en la animación de personajes en tiempo real.
- Sistema de individualización del personaje. Con este sistema es posible incluso reemplazar partes del cuerpo entero como cabezas o manos. Además, podemos cambiar del personaje a nuestro gusto. El sistema de deformación también permite la variación del personaje. Otra propiedad, basado en shaders, permite ensuciar manos y ropa.
- Animación paramétrica esquelética. CryENGINE 3 ofrece control interactivo de respuesta sobre un personaje con credibilidad. Personajes de CryENGINE 3 cambian la dirección y la velocidad suavemente o de repente, dependiendo si se mueve cuesta arriba o cuesta abajo, reaccionando a los cambios de escenarios sobre la marcha. Lo que permite una representación creíble y realista.
- Animación facial. Esta potente herramienta se encuentra dentro CryENGINE 3 Sandbox. Utiliza tecnología de análisis de forma de onda de audio para extraer fonemas (y otras características clave del habla) para animar rasgos faciales y proporcionar la sincronización de labios. La herramienta de seguimiento de vídeo capta los movimientos de la cara de un actor y estos se transfieren directamente al modelo facial en el editor, donde las expresiones se pueden combinar con la sincronización de labios a la vez que se editan.

Siendo los personajes tan realistas y animados las técnicas de Inteligencia artificial debe ir en concordancia con el personaje. CryENGINE 3 Sandbox permite manejar la conducta y los movimientos sensoriales del personaje a través de[8]:

- Designer-Friendly AI Editing System. Permite a los diseñadores controlar los comportamientos básicos mediante el sistema de scripting visual Flow-Graph, . Mientras que el lenguaje de programación LUA contrala variabilidad de estas conductas, creando respuestas más complejas. Esta combinación de sistemas permite la creación de cualquier comportamiento de inteligencia artificial a través de una tubería WYSIWYP extremadamente simple.
- Buscador de marcas dinámico. Los algoritmos avanzados en 2D y 3D permiten que las búsquedas en tiempo real sean agiles. Por tanto, la respuesta a eventos que pueden crear, modificar o destruir caminos es muchísimo más rápida. Lo que hace del entorno del personaje un mundo interactivo.
- Generación automática de malla de navegación. Contará con una generación de la malla de navegación automatizada en la búsqueda de caminos a diferencia de los sistemas basados en waypoint. El sistema está unificado para las zonas de interior y al aire libre y genera automáticamente, dentro de los volúmenes colocados por los diseñadores, una malla de navegación que edificios, movimientos especiales (saltos, escalada, etc.) y se adapta en tiempo real a los obstáculos dinámicos.

- Control de la hora y el día. Pues esta cambia dinámicamente durante una misión del juego. Así que se refleja los movimientos del sol, la posición de la luna, la iluminación y las condiciones atmosféricas durante cualquier ciclo día o de la noche predefinido.
- Alta Calidad de Agua 3D. CryEngine 3 permite que la reproducción de superficies de agua, como los océanos, pueden ser modificados debido al viento, generando efectos suaves automáticos cuando el agua se encuentra a la orilla ,o, según el contorno de la costa.

Otras de las características de CryEngine 3 es la gran variedad de herramientas que permite que mientras estamos desarrollando, realizar múltiples tareas para medir el rendimiento del motor en tiempo real, crear informes detallados de uso de memoria o ejecutar tutoriales de cada nivel para la adaptación a este motor de juego. Además CryENGINE 3 Sandbox dispone de un sistema de capas que permite la separación controlada de un nivel de juego. Al bloquear los elementos (de hasta 256 capas), se puede trabajar cada una de forma paralela, hasta que al final se fusionan. Este sistema permite a todo un equipo de desarrolladores hacer cambios de forma simultánea en un solo nivel sin la preocupación de impactar el trabajo desarrollado por sus compañeros. Este separación permite una estructura de trabajo claramente estratificada y si está bien organizada aumenta significativamente el progreso del desarrollo.[8]

El offline renderin, permite la creación de vídeos en streaming o imágenes fijas desde dentro del juego y la reproducción de una escena en cualquier resolución de pantalla, incluyendo la generación de vistas panorámicas para su visualización en pantallas de 360 grados. Estos videos son los que se utilizan como tráiler de los juegos.

El motor de sonido CryENGINE 3 ofrece muchas características y mejoras. Con su concepto de datos impulsados, garantiza un desarrollo de audio impecable y de bajo riesgo. Cada sonido lleva su propia especificación con ella, por lo que los diseñadores de sonido tienen pleno control del comportamiento dinámico además de la calidad final del sonido. Los sonidos se pueden crear con calidad de estudio, a la vez que aguanta cualquier configuración de los altavoces de sonido envolvente, disponibles hasta 7. Compatibilidad multi-plataforma, mejora de la transmisión de datos y optimización del rendimiento individual para Xbox 360 o PlayStation 3 están garantizados por la biblioteca de sonidos incluidos FMOD Ex.

Con un mínimo esfuerzo de codificación de sonidos, se puede reaccionar automáticamente y de manera compleja a parámetros como la distancia, la hora del día, o el ruido de la batalla utilizando efectos DSP en tiempo real. Esta técnica proporciona retroalimentación de audio no repetitiva y sensible en un mundo de juego interactivo y real en parámetros de la física. El sistema de música personalizable reacciona a cualquier evento juego, con el fin de dar al jugador una experiencia emocional e interactiva, como en las películas.

CryENGINE 3 permite a los diseñadores de sonido crear una impresión de densidad y se reproducen con precisión en la naturaleza, la mezcla perfecta entre diferentes entornos. La nueva tecnología permite un sonido ambiente que sea direccional en la distancia. En CryENGINE 3 los sonidos pueden ser provocados por la implantación de los mismos sobre animaciones combinadas para mejorar la aplicación de todos los efectos Foley. Añadir recursos de audio en archivos adjuntos de animación hace al personaje y al entorno mucho más realistas.

Las mezclas de audio se pueden definir para adaptarse a cualquier situación de juego. No sólo el volumen y el tono son controlables, sino también efectos DSP como filtros y ecualizadores. En el juego los sonidos se integran en los estados de ánimo predefinidos en tiempo real e imita la corriente efecto ambiental, por ejemplo, cuando se conduce dentro de un tanque.

AJEDREZ EN LA COMPUTACIÓN

En el ajedrez no existen motores de juegos específicos para su desarrollo, si bien, se ha creado una cultura en torno al ajedrez y la computación. Bases de datos con librerías de aperturas y finales, software interactivo para enseñar a jugar, visores de partidas (ideales para el juego medio) o software para resolver problemas; están haciendo del ajedrez y la programación dos disciplinas complementarias.

Rank	Name	Rating			Score	Average Opponent	Draws	Games	LOS
		ELO	+	−					
1	Stockfish 6 64-bit	3474	+14	−14	74.6%	−190.4	31.2%	2100	100.0%
2	Komodo 9 64-bit	3425	+15	−15	69.8%	−153.2	30.2%	1800	96.8%
3	Houdini 4 64-bit	3408	+10	−10	70.5%	−171.8	25.9%	4200	100.0%
4	Critter 1.6a 64-bit	3292	+8	−8	60.7%	−88.3	31.5%	6600	100.0%
5	Fire 3.0 64-bit	3178	+10	−11	46.3%	+29.6	24.7%	3900	96.6%
6	Rybka 4.1 64-bit	3167	+8	−8	41.2%	+64.1	29.5%	6400	100.0%
7	Chiron 2 64-bit	3112	+11	−11	37.8%	+101.7	21.0%	4200	100.0%
8	Naum 4.6 64-bit	3079	+12	−12	52.0%	−12.6	24.1%	3100	100.0%
9	Shredder 12	3018	+9	−9	36.1%	+113.2	21.8%	6500	100.0%
10	Spike 1.4 Leiden	2942	+10	−10	40.4%	+76.3	21.7%	4500	99.4%
11	Hiarcs 13.2	2924	+10	−10	40.1%	+82.2	20.0%	4700	100.0%
12	Deep Sjeng WC2008 64-bit	2877	+9	−9	45.1%	+42.1	21.0%	6300	100.0%
13	Cheng4 0.38 64-bit	2833	+17	−17	45.2%	+39.4	20.4%	1300	80.3%
14	Cyclone xTreme Fear	2824	+10	−10	47.9%	+15.1	20.7%	4900	81.4%
15	EXchess 7.18b 64-bit	2816	+15	−15	46.7%	+28.6	21.8%	1800	100.0%
16	GNU Chess 5.60 64-bit	2778	+16	−16	48.0%	+15.2	22.8%	1500	99.5%
17-18	Loop for Chess960	2755	+8	−8	53.8%	−31.7	22.8%	7500	53.0%
17-18	Tornado 4.88 64-bit	2755	+12	−12	53.9%	−31.5	17.9%	3000	93.6%
19	Octochess 5190 64-bit	2740	+15	−15	49.7%	+0.4	22.8%	1600	98.4%
20	Bright 0.4a	2721	+10	−10	55.7%	−46.9	19.0%	4600	91.5%
21	Frenzee 3.5.19 64-bit	2710	+12	−12	54.1%	−33.4	15.8%	2800	100.0%
22	Nebula 2.0 64-bit	2607	+12	−12	45.1%	+40.3	19.7%	2900	59.9%
23	Daydreamer 1.75 64-bit	2605	+11	−11	47.3%	+22.0	21.4%	3700	100.0%
24	Movei 00.8.438	2568	+9	−9	46.0%	+32.3	19.1%	6400	100.0%
25	The Baron 2.23	2546	+10	−10	48.3%	+11.9	17.0%	4400	56.4%
26	Pharaon 3.5.1	2545	+8	−8	45.4%	+36.5	18.5%	7200	91.2%
27	Hermann 2.8 64-bit	2535	+12	−12	50.4%	−4.7	16.3%	3200	87.0%
28-29	Amyan 1.72	2527	+11	−11	49.1%	+6.4	17.2%	3700	46.4%
28-29	Jonny 2.83	2527	+10	−10	49.0%	+8.2	15.8%	4300	96.1%
30	Hamsters 0.7.1	2515	+10	−10	47.9%	+16.3	20.2%	4400	100.0%
31	Ufim 8.02	2486	+8	−8	44.0%	+48.9	18.5%	7000	91.9%
32	DanaSah 4.66	2475	+13	−13	47.2%	+20.8	19.4%	2500	100.0%
33	LittleThought 1.052 64-bit	2405	+12	−12	45.0%	+38.4	16.6%	3300	100.0%
34	SlowChess 2.960d	2373	+13	−13	44.1%	+45.6	18.5%	2300	99.0%
35	Maverick 0.60 64-bit	2345	+20	−20	50.4%	−3.5	15.6%	1000	76.5%
36	Hussar 0.4	2337	+13	−13	39.1%	+86.0	18.2%	2500	99.8%
37	Patzer 3.80	2311	+13	−13	37.0%	+105.4	14.4%	2700	99.2%
38	TJchess 1.01	2289	+13	−13	35.1%	+122.4	15.7%	2700	100.0%
39	Aice 0.99.2	2258	+10	−10	33.3%	+134.2	14.7%	4500	80.8%
40	Ayito 0.2.994	2252	+11	−11	33.0%	+139.8	13.0%	4400	99.7%
41	Betsy Fischer	2230	+13	−13	32.6%	+142.4	15.9%	2900	100.0%
42	Chispa 4.0.3	2175	+14	−14	34.7%	+121.1	24.0%	2100	100.0%
43	Homer 1.02	2123	+17	−17	36.0%	+108.4	16.8%	1500	100.0%
44	Taktix 2.23x	1887	+25	−25	33.3%	+139.8	12.4%	1000	67.1%
45	BigLion 2.23x	1879	+25	−26	32.6%	+148.6	10.2%	1000	100.0%
46	ArcBishop80 2.24a	1485	+43	−45	24.7%	+253.0	11.3%	300	91.9%
47	BigLion80 2.23x	1448	+44	−47	19.7%	+302.3	11.3%	300	

Figura 5. Ranking ELO 2015 Motor Ajedrez. Fuente: http://www.computerchess.org.uk/

Es una herramienta para jugadores aficionados como para profesionales. Para el aficionado porque le permite conectarse con jugadores alrededor del mundo y mejorar su nivel jugando contra la máquina. Sin embargo, el profesional las usa como un instrumento estudio, valiéndose de la capacidad de síntesis y visión de juego que tienen las máquinas, para analizar partidas y resolver problemas del juego medio, que es cuando existe una mayor combinación de movimientos diferentes, haciendo de ellas el medio de revolución en el estudio del ajedrez.

Además permiten compilar y almacenar toda y cada una de las aperturas del juego. La evolución y creación de máquinas más potentes, está procediendo al estudio de diferentes líneas de juego estratégico, ya que la capacidad analítica de las máquinas es fortísima.

Cabe destacar que existe un ranking internacional con el ELO (sistema de puntación del ajedrez) con las máquinas más poderosas que existen.

Aunque, como se ha dicho antes, no existan motores de juego específicos para desarrollar plataformas de ajedrez, quiero hacer hincapié en las líneas de estudio que existen para poder desarrollar máquinas. Resulta interesante puesto que existen varios frentes de estudio abiertos.

Implementación de juegos de ajedrez. Líneas de estudio.

***Origen*.**

El padre de la programación ajedrecística es el científico estadounidense Claude E. Shannon. Fue el encargado de trazar y presentar las líneas maestras en una convención en Nueva York bajo el título "Programming a Computer for Playing Chess". Una de las mentes más brillantes de los laboratorios Bell, predijo que habría dos maneras diferentes de buscar la mejor jugada durante una partida, la Tipo A y la Tipo B[5].

La tipo A sería la más simple de todas. Un algoritmo de fuerza bruta se encargaría de examinar todas las ramas de un árbol en el que estarían presentes todos los movimientos, usando un algoritmo minimax. Aunque llegó a la conclusión que sería muy poco práctico porque:

1. En el juego medio existe una media de 30 movimientos posible en una jugada, por tanto, examinar cada nodo del árbol sería una tarea dificultosa y laboriosa. Pero Shannon no contaba con la evolución de la tecnología, con máquinas potentes este problema quedó solucionado.
2. La demora de la latencia. El retardo provocado por examinar cada uno de los nodos finales del árbol.

Por estas razones introdujo en la estrategia de búsqueda de tipo B el concepto de búsqueda selectiva. Durante una partida de ajedrez, el jugador, no se para a examinar todos los movimientos posibles, sino aquellos son más productivos para su posición, ataque o defensa. Así los movimientos “malos” no son analizados.

Además en este estudio introdujo una serie de consideraciones que deben de ser evaluadas en función de la posición que se encuentra cada jugada, entre las cabe destacar:

- Ventaja Material
- Estructura de Peones
    - Peones aislados y retrasados.
    - Control relativo del centro.
    - Peones en color opuesto al alfil propio.
    - Peones pasados.
- Posición de las piezas.
    - Caballos avanzados, protegidos o avanzados.
    - Torres en columnas abiertas o semiabiertas.
    - Torres en séptima fila.

    - Torres dobladas.
- Posibilidades de ataque.
    - Piezas que protegen a otras piezas.
    - Ataques sobre otras piezas.
    - Ataques sobre casillas adyacentes al rey enemigo.
    - Clavadas.
- Movilidad, medida por el número de movimientos legales posibles.

Si nos paramos a pensar detenidamente en los programas de Tipo B, ¿es la máquina la que tiene la responsabilidad de tomar las decisiones importantes?; ¿de qué movimientos vamos a prescindir? Es por eso que con la evolución y desarrollo de la velocidad del hardware los programas Tipo A se hicieron mucho más famosos.

Offset Board Representation. Representación del trablero

Se genera un array bidimensional con ocho elementos, tablero[8][8] . En el estarán representada todas las fichas del tablero y tendrán los siguientes valores:

- 0Vacío
- 1Peón blanco
- 2Caballo blanco
- 3Alfil blanco
- 4Torre blanca
- 5Dama blanca
- 6Rey blanco
- -1Peón negro
- -2Caballo negro
- -3Alfil negro
- -4Torre negra
- -5Dama negra
- -6Rey negro

Técnicas de búsqueda.

Debemos tener presente que la inteligencia artificial y el ajedrez van cogidos de la mano. Cada movimiento de pieza es el nodo de un árbol de búsqueda y ella tiene un movimiento de respuesta, por tanto, nace otra rama diferente que estamos en la obligación de evaluar.

Entre los algoritmos de búsqueda utilizados y heurísticas para mejorar tiempos de búsqueda en el ajedrez podemos encontrar:

- *Algoritmo Minimax.* Nuestra computadora intentará buscar la posición que le dé un mayor beneficio ante el próximo movimiento.
- *Poda Alfa-Beta.* Presente en el estudio de Shannon. Cuando estamos jugando al ajedrez no nos paramos a analizar todos los movimientos del rival. Por tanto, nos limitamos a buscar en aquellas ramas que tengan ventaja sobre el resto, así descartamos movimientos que no sean buenos y mejoramos los tiempos de búsqueda.
- *Heurística del movimiento nulo.* Durante los finales nos encontraremos con la posibilidad de no realizar movimientos que cambien nuestra posición de ventaja en el tablero. Aquí es donde entra en juego esta heurística. Si el lado donde tenemos que movernos es un jaque o si se encuentra un rey con muchos peones eliminaremos por completo está opción de movimiento en el árbol de búsqueda.
- *Reducción del Movimiento.* Esta heurística pretender ordenar los movimientos en una lista de mayor a menor puntación.

Evaluación de las hojas finales.

Una vez exploradas todas las hojas nos encontramos con la tesitura de evaluar el resultado de la búsqueda final. Quizás sea una de las problemáticas más importantes que nos encontramos en el ajedrez por computación, debido, en gran medida, a la información que tenemos que evaluar. En esta función multivariable se verán representado:

- Valor material. Cada pieza tendrá una puntación. El rey tiene un valor muy alto para que el jaque mate sobrepase al resto de opciones.
- Estado y control del tablero. Fundamental para determinar quien tiene ganada la posición.
- Seguridad del Rey. No es lo mismo que el rey se encuentre enrocado a que este desprotegido.

- Estructura de los peones. Hay que tener en cuenta si un peón se encuentre pasado (a punto de coronar), bloqueados o doblados. Aunque su valor material sea el más bajo hay que tener en cuenta su posición puede resolver partidas

Bases de datos de ajedrez.

Ideales para jugadores que se están iniciando en el ajedrez. La cantidad ingente de aperturas hace de ellas el método de estudios más extendido a lo largo del mundo. Sin ellas, los principiantes no tendrían la oportunidad de analizar cada movimiento que se produce en ellas, conociendo todas sus variables posibles.

Lo mismo pasa con finales de partida pero ellos dependen del número de piezas que se encuentran en el tablero. Existen bases de datos que analizan finales en los que hay hasta siete piezas en juego. En este apartado va a tener un peso importante la capacidad de almacenamiento de la máquina, puesto que cuanto más fichas tengamos en juego, más espacio vamos a necesitar.

En cuanto a las bases de datos en la que se encuentran los finales de partida, debemos decir que se encuentran almacenadas de manera retrospectiva, es decir, se empieza desde el jaque mate hasta llegar al punto en el que nos encontramos.

PROYECCIÓN DIDÁCTICA

Introducción

El módulo de programación de de este ciclo queda definido, a nivel estatal, en el Real Decreto 450/2010, de 16 de abril, por el que se establece el título de Técnico Superior en Desarrollo de Aplicaciones Multiplataforma y se fijan sus enseñanzas mínimas y en la Orden EDU/2000/2010, de 13 de julio, por la que se establece el currículo del ciclo formativo de Grado Superior correspondiente al título de Técnico Superior en Desarrollo de Aplicaciones Multiplataforma; y, a nivel autonómico**,** en la Orden de 16 de junio de 2011, por la que se desarrolla el currículo correspondiente al título de Técnico Superior en Desarrollo de Aplicaciones Multiplataforma.

Perfil del alunando fatal tú, míralo y compara con el original.

Nuestro centro se encuentra en Úbeda, una ciudad de 35.000 habitantes. La capital de La Loma tiene una población flotante de unas 120.000 personas y es el centro comercial de toda la comarca. El tejido productivo gira en torno al comercio, sector servicios y la agricultura.

El turismo es otra fuente de riqueza de la ciudad, puesto que Úbeda, junto con Baeza, tiene la denominación Patrimonio de la humanidad de la Unesco.

El centro es referencia en la Comarca, es uno de los pocos de la provincia donde existe el ciclo formativo de aplicaciones multiplataforma. Además, numerosos estudiantes se desplazan a diario al mismo para poder cursar sus estudios. Los planes de estudios desarrollados en él son: educación secundaria obligatoria, tres modalidades de bachillerato además de la formación profesional. Es por esto que nos encontramos en un centro una gran variedad de alumnado, motivo este, hace que los grupos ¿no se conozcan? lo que propicia que el alumnado se integre y se conozco mucho más rápidamente.

Justificación.

Este proyecto nace de la idea de asociar dos disciplinas diferentes y muy próximas entre sí: el ajedrez y la programación informática.

En el último tramo de la asignatura Tecnología de la Información y Comunicación (estudiada en segundo de bachillerato), se nos presenta la unidad didáctica de programación. En ella están presentes los lenguajes de programación, la programación estructurada orientada a objetos. Introducir a los alumnos el concepto de algoritmo, bucle, secuencia o clase resulta una tarea tediosa y compleja puesto que es la primera vez que tienen que afrontarlo, teniendo en cuenta además la dificultad añadida que tiene la representación mental de un algoritmo, su desarrollo y las búsqueda de una solución.

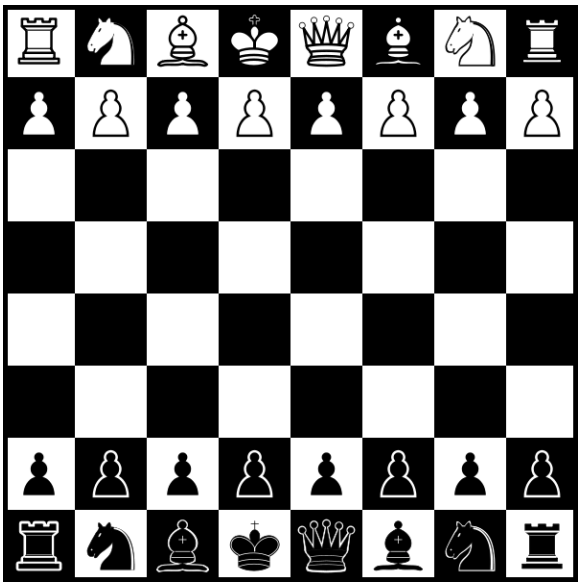


Figura 6. Tablero de Ajedrez. Matriz 8 x 8 . Fuente docs.kde.org

¿Por qué no utilizar un tablero de ajedrez y sus piezas? Es un recurso barato, los tableros y piezas son fáciles de costear. También, ayuda a la realización de este objetivo la existencia del compromiso de todos los partidos de políticos de España para introducir la práctica del ajedrez como asignatura en la escuela, por lo que este material estará presente en todos los centros educativos[11].

El ajedrez es programación en sí mismo; el tablero es una matriz, un algoritmo una apertura, el jaque es una condicional o la coronación de un peón es una estructura de control. El final de partida con dos torres y un rey contra un rey se puede resolver de manera iterativa con la programación estructurada.

Además, el ajedrez tiene las siguientes propiedades intelectuales[12]:

- Atención y concentración. Con un tablero y unas piezas un niño o niña es capaz de concentrase y pensar.
- Análisis y síntesis. Durante el desarrollo de una partida el jugador se ve obligado a observar cada uno de los movimientos de su contrincante, por tanto, debe dar una respuesta rápida y eficaz
- Razonamiento lógico-matemático: El mismo razonamiento que se presenta en el ajedrez está presente en las matemáticas.
- Creatividad e imaginación: Durante la partida estamos obligados a buscar nuevas vías de acción para derrotar al enemigo, siempre hay que buscar nuevas alternativas posibles.

Cambiar distribución con el otro.

Todas ellas se encuentran embebidas en el Pensamiento Computacional, que tiene por objeto introducir la programación informática en las aulas y en todas las edades. Fur impulsado por la International Society for Technology in Education (ISTE) y presenta los siguientes beneficios[13]:

- Organizar y analizar datos de manera lógica.
- Representar datos mediante abstracciones, como modelos y simulaciones.
- Automatizar soluciones mediante pensamiento algorítmico (una serie de pasos ordenados).
- Identificar, analizar e implementar posibles soluciones con el objeto de encontrar la combinación de pasos y recursos más eficiente y efectiva.
- Generalizar y transferir ese proceso de solución de problemas a una gran diversidad de problemas.

Como vemos ajedrez y programación presentan nexos de unión, entonces; ¿Por qué no lo llevamos a la práctica?

Competencia General

La competencia general de este título consiste en desarrollar, implantar, documentar y mantener aplicaciones informáticas multiplataforma utilizando tecnologías y entornos de desarrollo específicos, garantizando el acceso a los datos de forma segura y cumpliendo los criterios de «usabilidad» y calidad exigidas en los estándares establecidos.

Objetivos específicos de esta unidad.

Art. 1

d) Instalar y configurar módulos y complementos, evaluando su funcionalidad, para gestionar entornos de desarrollo.

e) Seleccionar y emplear lenguajes, herramientas y librerías, interpretando las especificaciones para desarrollar aplicaciones multiplataforma con acceso a bases de datos

Competencias profesionales, personales y sociales de esta unidad.

t) Establecer vías eficaces de relación profesional y comunicación con sus superiores, compañeros y subordinados, respetando la autonomía y competencias de las distintas personas.

d) Gestionar entornos de desarrollo adaptando su configuración en cada caso para permitir el desarrollo y despliegue de aplicaciones.

Contenidos básicos de esta unidad

1. Identificación de los elementos de un programa informático:
    - Estructura y bloques fundamentales.
    - Variables.
    - Tipos de datos.
    - Literales.
    - Constantes.
    - Operadores y expresiones.
    - Conversiones de tipo.
    - Comentarios.
    - Entornos integrados de desarrollo.
        - ✓ Definición y tipos. Entornos comerciales y de Software libre.
        - ✓ Instalación y descripción de entornos integrados de desarrollo.
        - ✓ Creación de proyectos. Estructura y componentes.

2. Uso de estructuras de control:
    - Estructuras de selección.
    - Estructuras de repetición.

- Estructuras de salto.
- Control de excepciones.
- Depuración de programas.
- El depurador como herramienta de control de errores.
- Documentación de programas.
    - ✓ Documentación interna, comentarios.
    - ✓ Documentación externa, diagramas de clases, requisitos, guías, etc.

Resultados de aprendizaje y criterios de evaluación de esta unidad programación

1. Reconoce la estructura de un programa informático, identificando y relacionando los elementos propios del lenguaje de programación utilizado.

   Criterios de evaluación:

   a. Se han identificado los bloques que componen la estructura de un programa informático.
   b. Se han creado proyectos de desarrollo de aplicaciones.
   c. Se han utilizado entornos integrados de desarrollo.
   d. Se han identificado los distintos tipos de variables y la utilidad específica de cada uno.
   e. Se ha modificado el código de un programa para crear y utilizar variables.
   f. Se han creado y utilizado constantes y literales.
   g. Se han clasificado, reconocido y utilizado en expresiones los operadores del lenguaje.
   h. Se ha comprobado el funcionamiento de las conversiones de tipos explícitas e implícitas.
   i. Se han introducido comentarios en el código.

2. Escribe y depura código, analizando y utilizando las estructuras de control del lenguaje.

   Criterios de evaluación:

   a. Se ha escrito y probado código que haga uso de estructuras de selección.
   b. Se han utilizado estructuras de repetición.

c. Se han utilizado estructuras de repetición.

d. Se han reconocido las posibilidades de las sentencias de salto.

e. Se ha escrito código utilizando control de excepciones.

f. Se han creado programas ejecutables utilizando diferentes estructuras de control.

g. Se han probado y depurado los programas.

h. Se ha comentado y documentado el código.

Indicadores de evaluación

- Identificada los bloques que componen la estructura de un programa informático.
- Crea proyectos de desarrollo de aplicaciones.
- Utiliza entornos integrados de desarrollo.
- Identificada los distintos tipos de variables y la utilidad específica de cada uno.
- Modificada el código de un programa para crear y utilizar variables.
- Crea y utiliza constantes y literales.
- Clasificada, reconoce y utiliza expresiones de los operadores del lenguaje.
- Comprueba el funcionamiento de las conversiones de tipos explícitas e implícitas.
- Introduce comentarios en el código.
- Escribe y prueba el código usando de estructuras de selección.
- Utiliza estructuras de repetición.
- Reconoce las posibilidades de las sentencias de salto.
- Escribe código utilizando control de excepciones.
- Crea programas ejecutables utilizando diferentes estructuras de control.
- Prueba y depura los programas.
- Comenta y documenta el código.

Instrumentos de evaluación.

1. Prácticas: 40%. Es la parte más significativa y representativa de la programación, por tanto, va a tener un peso elevado dentro de la misma.

2. Prueba Escrita: 35%. Se realizará dentro de la plataforma MOODLE para ahorrar tiempo.
3. Tareas Clase: 35%. En el momento que termine la explicación se pondrán unos ejercicios prácticos para que el alumno los realice en clase.

Metodología.

Al estar dentro de un ciclo formativo nos vamos a encontrar con que la duración de las clases es de dos horas. Por tanto, invertiremos media hora en dar una explicación teórica y el resto se invertirá en desarrollar ejercicios prácticos. Para representar los elementos de la programación estructurada tomaré de referencia un tablero de ajedrez sirviéndome de la estructura de su tablero (que es una matriz ocho por ocho).

La programación informática requiere mucha más práctica que teoría, vislumbrar el resultado de un programa informático se observa con mucha más facilidad a través del ordenador que de un papel.

En el centro contamos con MOODLE, por tanto, el alumnado no tendrá obligación de invertir en un manual para la asignatura. A través de paquetes SCROM iré colgando los apuntes de la asignatura en esta plataforma. Además de todo el material necesario para el desarrollo de la unidad (así como el enlace al entorno de desarrollo Netbeans), prácticas, ejercicios, incluyendo el examen que se realizará en la misma plataforma.

Me he decantado por el entorno de desarrollo integrado Netbeans porque es muy fácil de instalar y porque cuenta con licencia gratuita. Otra de las propiedades de esta plataforma es la facilidad con la genera la documentación de código.

En cuanto a las prácticas, tendrán un peso elevado dentro de nuestra unidad didáctica. Estas serán evaluadas en función de la presentación de las mimas. Se va a generar una dinámica de trabajo muy similar a la de una empresa. Dentro de la plataforma MOODLE se irán marcando los objetivos que han de cumplir las mismas (rúbrica) además del tiempo que tendrá cada alumno para poder entregarlas.

Su duración será de 8 sesiones, incluyendo el examen. Estamos ante la primera toma de contacto del alumnado con un entorno de desarrollo integrado, incluyendo la programación, por tanto, trabajaremos con paciencia para que sea capaz de asimilar conceptos de una manera clara y efectiva.

Sesiones

Sesión 1

En la primera sesión la parte teórica va a tener la duración de dos horas, puesto que vamos a identificar los elementos de un programa informático, además de su estructura y bloques fundamentales. También vamos a presentar los entornos integrados de desarrollo su definición y tipos.

Agrupación: por parejas mixtas.

Recursos: ordenadores del aula de informática y proyector.

Espacio: Aula informática del centro.

Sesión 2

La parte teórica va a tener una duración de una hora. Preguntaremos dudas que hayan podido surgir y empezaremos a analizar entornos de desarrollo comerciales y de software libre, aprendiendo a instalarlos, y creando un proyecto de prueba para comprobar cómo funciona la herramienta de depuración de errores. Tendremos una hora para trabajar en clase.

Prácticas: Ejercicios

Agrupación: por parejas mixtas.

Recursos: ordenadores del aula de informática y proyector.

Espacio: Aula informática del centro.

Sesión 3

La parte teórica va a tener una duración de una hora. Preguntaremos dudas que hayan podido surgir y empezaremos a analizar las variables, los tipos de datos y los literales. Además aprenderemos a documentar código. Tendremos una hora para trabajar en clase.

Prácticas: Ejercicios

Agrupación: por parejas mixtas.

Recursos: ordenadores del aula de informática y proyector.

Espacio: Aula informática del centro.

Sesión 4

La parte teórica va a tener una duración de una hora. Preguntaremos dudas que hayan podido surgir y empezaremos a analizar constantes, operadores y expresiones, conversiones de tipo. Tendremos una hora para trabajar en clase.

Prácticas: Ejercicios

Agrupación: por parejas mixtas.

Recursos: ordenadores del aula de informática y proyector.

Espacio: Aula informática del centro.

Sesión 5

Seguir y empezaremos a analizar las estructuras de selección. Tendremos una hora para trabajar en clase.

Prácticas: Ejercicios

Agrupación: por parejas mixtas.

Recursos: ordenadores del aula de informática y proyector.

Espacio: Aula informática del centro.

Sesión 6

La parte teórica va a tener una duración de una hora. Preguntaremos dudas que hayan podido surgir y empezaremos a analizar las estructuras de repetición. Tendremos una hora para trabajar en clase.

Prácticas: Ejercicios

Agrupación: por parejas mixtas.

Recursos: ordenadores del aula de informática y proyector.

Espacio: Aula informática del centro.

Sesión 7

La parte teórica va a tener una duración de una hora. Preguntaremos dudas que hayan podido surgir y empezaremos a analizar las estructuras de salto y control de excepciones. Tendremos una hora para trabajar en clase.

Prácticas: Ejercicios

Agrupación: por parejas mixtas.

Recursos: ordenadores del aula de informática y proyector.

Espacio: Aula informática del centro.

Sesión 8

Examen. Constará de diez preguntas de tipo test, con opción de respuesta múltiple. Además de cuatro preguntas cortas para responder y dos ejercicios donde se planteará un programa para resolverlo.

Prácticas: Ejercicios

Agrupación: alumno en solitario.

Recursos: ordenadores del aula de informática.

Espacio: Aula informática del centro.

Cronograma

Sesión	Contenidos	Duración Teoría	Duración Ejercicios. Nº Relación	Prácticas	Recursos	Agrupación	Espacio
1	Identificar los elementos de un programa informático, además de su estructura y bloques fundamentales. También vamos a presentar los entornos integrados de desarrollo su definición y tipos..	2,0 H			Ordenadores del aula de informática y proyector.	Agrupación: por parejas mixtas	Aula informática del centro.
2	Identificaremos entornos comerciales y de Software libre. Además aprenderemos a instalarlos y creamos un proyecto de prueba para comprobar cómo funciona la herramienta de depuración de errores	1,00 H	1,00 H Nº1	Nº 2 y3	Ordenadores del aula de informática y proyector.	Agrupación: por parejas mixtas	Aula informática del centro.
3	Analizar las variables, los tipos de datos y los literales	1,00 H	1,00 H Nº4 y5	Nº 6 y 7	Ordenadores del aula de informática y proyector.	Agrupación: por parejas mixtas	Aula informática del centro.

4	Analizar constantes, operadores y expresiones, conversiones de tipo.	1,00 H	1,00 H Nº8 y9	Nº10 y 11	Ordenadores del aula de informática y proyector.	Agrupación: por parejas mixtas	Aula informática del centro.
5	Analizar las estructuras de selección	1,00 H	1,00 H Nº 12 y 13	Nº 14 y 15	Ordenadores del aula de informática y proyector.	Agrupación: por parejas mixtas	Aula informática del centro.
6	Analizar las estructuras de repetición	1,00 H	1,00 H Nº 16 y 17	Nº 18 y 19	Ordenadores del aula de informática y proyector.	Agrupación: por parejas mixtas	Aula informática del centro.
7	analizar las estructuras de salto y control de excepciones	1,00 H	1,00 H Nº 20 y 21	Nº 22 y 23	Ordenadores del aula de informática y proyector.	Agrupación: por parejas mixtas	Aula informática del centro.
8	EXAMEN	2,00			Ordenadores del aula de informática.	Agrupación: En solitario	Aula informática del centro.

Evaluación

A la hora de evaluar a los alumnos vamos a seguir tres instrumentos. Entre ellos sumarán un total de diez puntos y cada uno se valora sobre diez, teniendo un porcentaje específico de la notal final.

- Pruebas escrita: 30 %.
  - 10 preguntas tipo test verdadero y falso. Dos preguntas incorrectas resta una que este bien. Puntación total: 4 puntos.
  - 2 Preguntas de respuesta corta. Puntación total: 2 puntos. Cada pregunta vale un punto.
  - 2 Programas informáticos. Cada programa valdrá dos puntos. Se valorará que compile además de su eficiencia.

- Prácticas: 40%.
  - Cada práctica valdrá 1.6. De estos :
    - 0,55 Valdrá su entrega dentro de plazo.
    - 0,55 Valdrá la documentación.
    - 0,55 Valdrá el correcto funcionamiento.

- Tareas Clase: 30%.
  - Cada tarea valdrá 1.6. De estos :
    - 0,55 Valdrá su entrega dentro de plazo.
    - 0,55 Valdrá la documentación.
    - 0,55 Valdrá el correcto funcionamiento.

Atención a la diversidad.

La primera medida a tomar será la reserva de plazas para los alumnos con discapacidad. Reservaremos al menos, un 5% de las plazas para el alumnado que tenga reconocido un grado de discapacidad igual o superior al treinta y tres por ciento. Estamos en un centro que es referencia para toda la comarca de La Loma y la informática la pueden

estudiar muchos alumnos con discapacidad física puesto que existen múltiples adaptaciones de ordenador. Con esta medida pretendemos que los alumnos se acercan a nuestro instituto.

Jornadas mujeres e informática. Es uno de los puntos flacos de esta disciplina. En una ciencia dominada por el hombre estamos en la obligación de acercar la informática a la mujer. Para ello se llevarán a cabo unas jornadas impulsadas por el mismo departamento que traten de acercar esta materia a todas las alumnas.

Úbeda es una ciudad en la que agricultura es una de sus piedras angulares. Durante el mes de noviembre los inmigrantes llegan a la ciudad para conseguir trabajo en la aceituna, mucho de ellos terminan residiendo en la ciudad. Como centro tenemos que estar preparados para la incorporación del nuevo alumnado que provenga de cualquier parte del mundo. Para ello debemos crear las condiciones óptimas para que se integre. Entre las medidas más significativas destacaremos la adaptación curricular, aulas temporales de adaptación lingüística y medidas especiales de evaluación.

Necesidades educativas especiales

Las medidas de atención a la diversidad para los alumnos con necesidades educativas especiales también se encuentran presentes en los ciclos formativos de formación profesional. Aunque las enseñanzas serán las mismas para los alumnos con necesidades educativas especiales podrán disfrutar de adaptaciones circulares no significativas (adecuación de la presentación de los contenidos, metodología, de los procedimientos de evaluación o de acceso a currículo.). En cuanto a los módulos o unidades de competencia gozará de adaptaciones curriculares significativas.

Anexo. Relación de Ejercicios

1. Imprime por pantalla el nombre de los diez mejores ajedrecistas de todos los tiempos. No olvides documentar el código.
2. Imprime por pantalla el nombre de las diez mejores máquinas de ajedrez. No olvides documentar el código.
3. Imprime por pantalla el listado de países que han ganado el campeonato del mundo. Ordénalos en función del número de títulos.
4. Las piezas de ajedrez tienen el siguiente valor material:
    - Peón: 1
    - Caballo: 3
    - Alfil: 3,5
    - Torre: 5
    - Dama: 10

    Defínelas como litareles y muestra su valor por pantalla. No olvides documentar el código.
5. El tablero de ajedrez tiene la siguiente estructura de salida.

Crea un array bidimensional y define cada casilla de la siguiente manera:

- 0Vacío
- 1Peón blanco
- 2Caballo blanco
- 3Alfil blanco
- 4Torre blanca

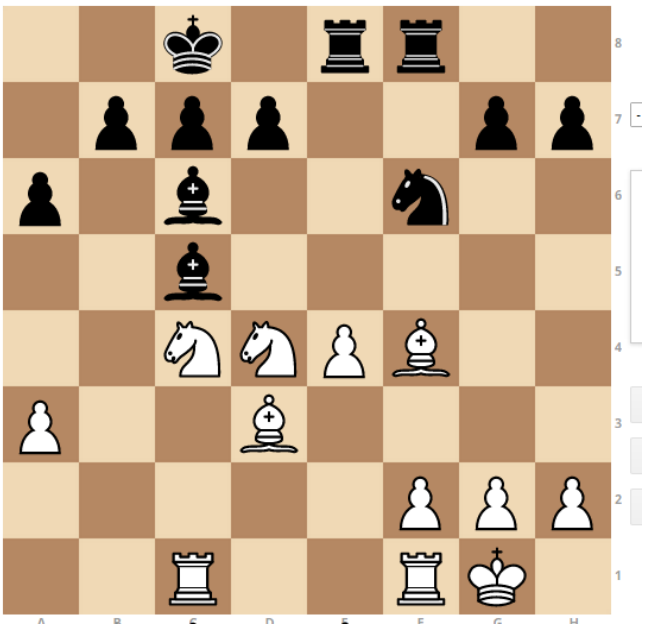
- 5Dama blanca
- 6Rey blanco
- -1Peón negro
- -2Caballo negro
- -3Alfil negro
- -4Torre negra
- -5Dama negra
- -6Rey negro

Muestras los resultados por pantalla. No olvides documentar el código.

6. Las piezas de ajedrez tienen el siguiente valor material:
    - Peón: 1
    - Caballo: 3
    - Alfil: 3,5
    - Torre: 5
    - Dama: 10

    Define un array de float, introdúcelas y muestra su valor por pantalla. No olvides documentar el código.

7. El tablero de ajedrez tiene la siguiente estructura.



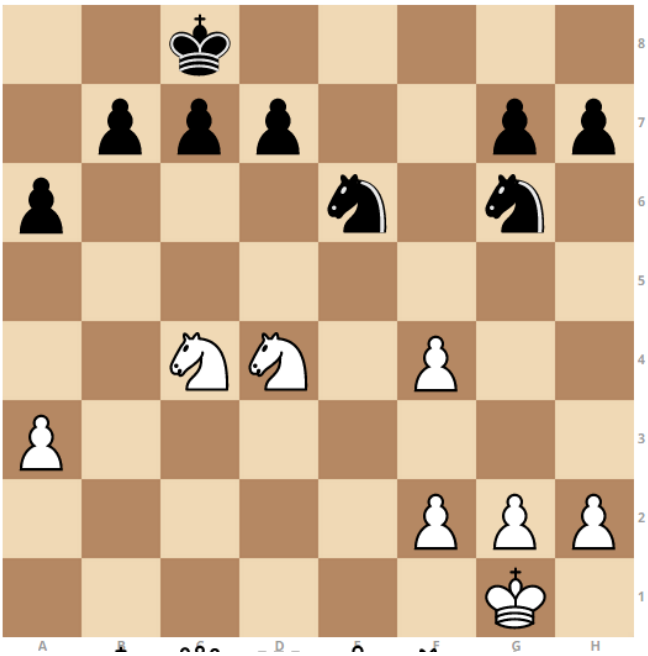
Crea un array bidimensional y define cada casilla de la siguiente manera:

- 0Vacío

- 1Peón blanco
- 2Caballo blanco
- 3Alfil blanco
- 4Torre blanca
- 5Dama blanca
- 6Rey blanco
- -1Peón negro
- -2Caballo negro
- -3Alfil negro
- -4Torre negra
- -5Dama negra
- -6Rey negro

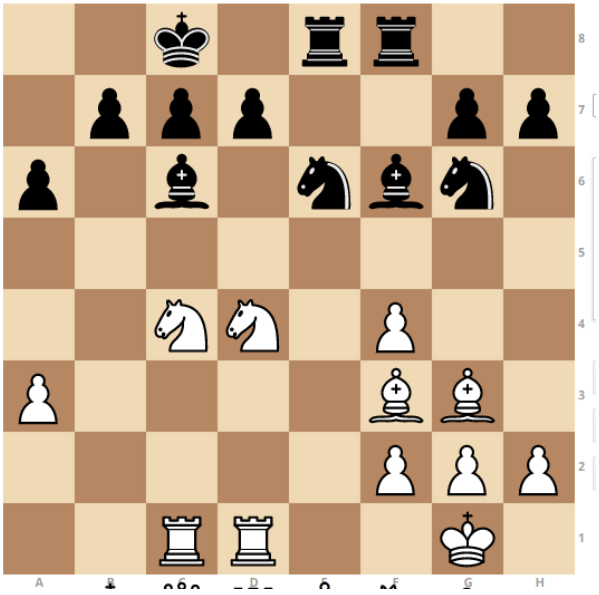
Muestras los resultados por pantalla. No olvides documentar el código.

8. Define el constante caballo negro que vale -2, blanco que vale 2. Peón blanco vale 1 y peón negro -1. Rey blanco vale 10 y rey negro vale -10. Después representa el siguiente tablero.



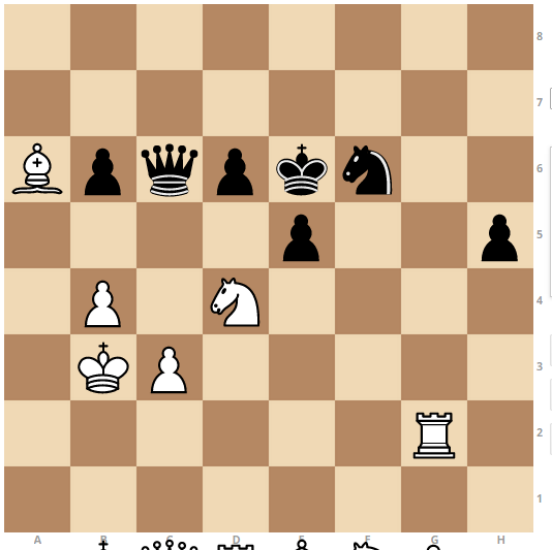
9. Utilizando las constantes del ejercicio anterior y alfil blanco y negro, 3 y -3 respectivamente, representa la siguiente estructura de tablero.

10. Utilizando las constantes del ejercicio anterior y torre blanca y negra,4 y -4 respectivamente, representa la siguiente estructura de tablero.

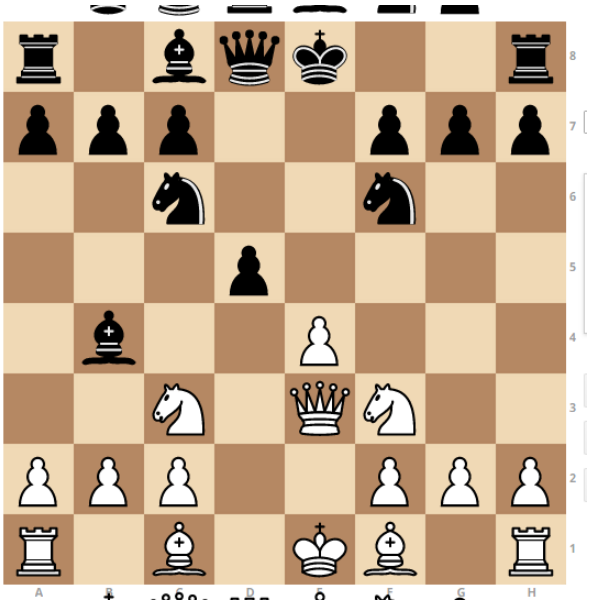


11. Utilizando las constantes del ejercicio anterior y dama blanca y negra,5 y -5 respectivamente, representa la siguiente estructura de tablero.

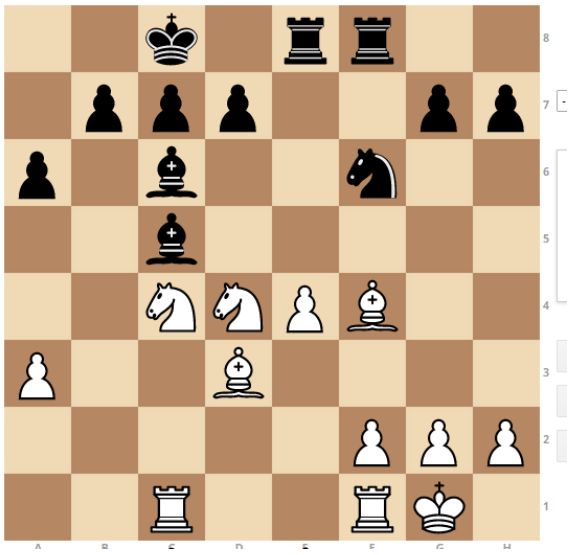
12. Pedro y Pablo acaban de terminar una partida de ajedrez. Deciden determinar el ganador de la partida por puntos. Sabiendo el valor de las piezas, ¿Quién habrá ganado?. Utiliza la estructura if, no olvides mostrar el resultado por pantalla.



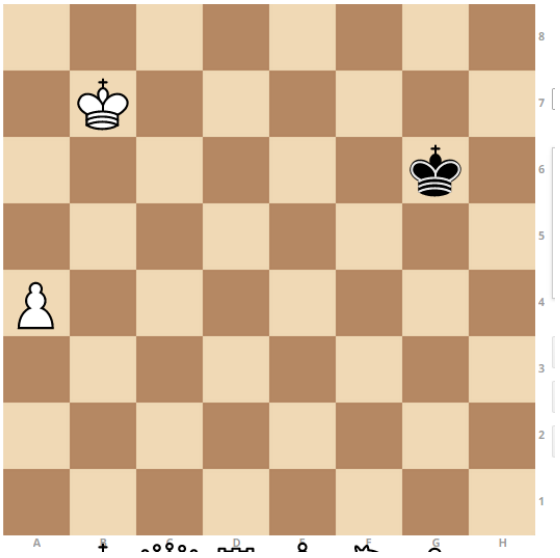
13. El tablero de ajedrez tiene la siguiente posición, utilizando un array bidimensional y representado las casillas de la misma forma que el ejercicio 5. ¿A qué posiciones se puede mover el caballo que hay en C3?. Utiliza la estructura if. No olvides documentar el código.



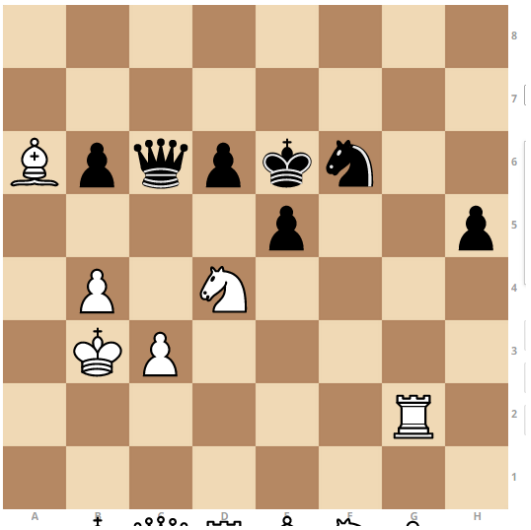
14. Pedro y Pablo acaban de terminar una partida de ajedrez. Deciden determinar el ganador de la partida por puntos. Sabiendo el valor de las piezas, ¿Quién habrá ganado?. Utiliza la estructura if, no olvides mostrar el resultado por pantalla.



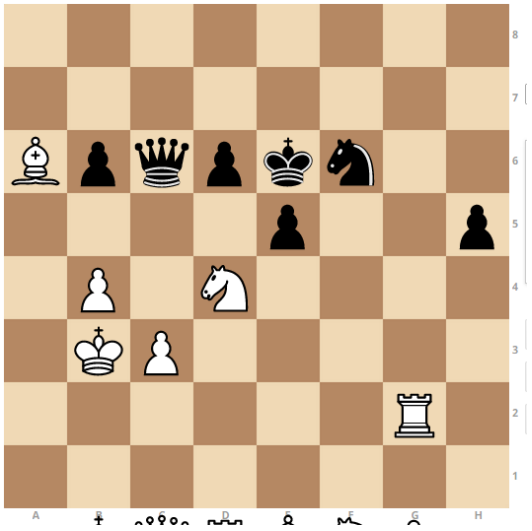
15. El tablero de ajedrez tiene la siguiente posición, utilizando un array bidimensional y representado las casillas del ejercicio anterior. ¿Por cuantas casillas tiene que pasar el peón para coronar?. Utiliza la estructura swtitch cuando llegue a la última, se debe imprimir por pantalla que pieza desea seleccionar. Utiliza una para variable para contar todas las posiciones .No olvides documentar el código.



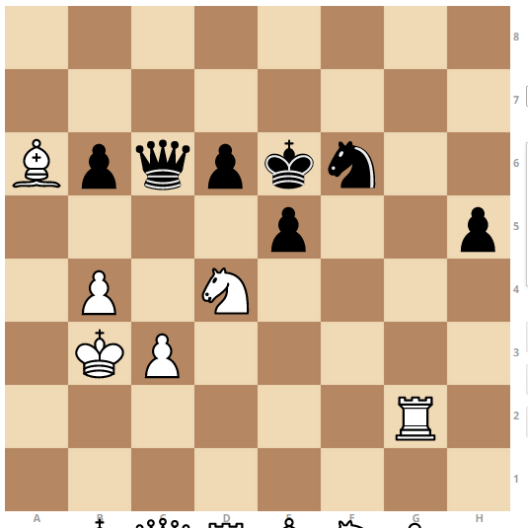
16. El tablero de ajedrez tiene la siguiente posición. Determina por a que casillas pude ir la torre situada en G2. Utiliza la estructura for e imprímelo por pantalla. No olvides documentar el código.



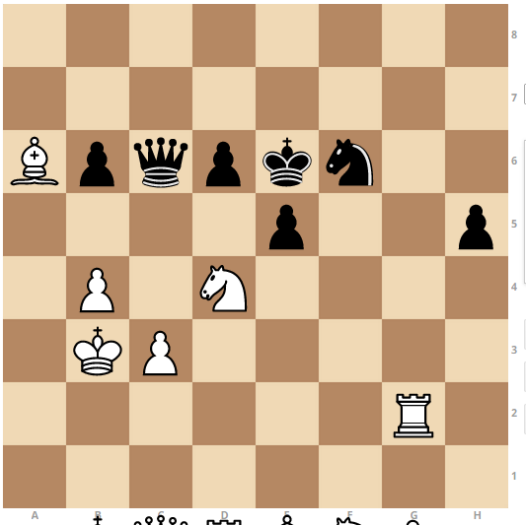
17. El tablero de ajedrez tiene la siguiente posición. Determina por a que casillas pude ir la dama situada en C6. Utiliza la estructura for e imprímelo por pantalla. No olvides documentar el código.



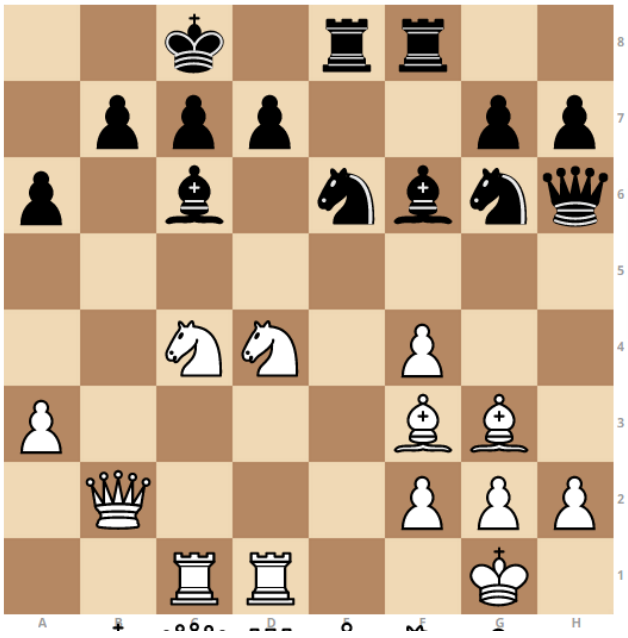
18. El tablero de ajedrez tiene la siguiente posición. Determina por a que casillas pude ir la torre situada en G2. Utiliza la estructura for e imprímelo por pantalla. No olvides documentar el código.



19. El tablero de ajedrez tiene la siguiente posición. Determina por a que casillas pude ir la dama situada en C6. Utiliza la estructura for e imprímelo por pantalla. No olvides documentar el código.



20. El tablero de ajedrez tiene la siguiente posición. Determina por a que casillas pude ir la dama situada en B6. Utiliza la estructura do while e imprímelo por pantalla. En el caso que se encuentre con un obstáculo utiliza una excepción. No olvides documentar el código.



21. El tablero de ajedrez tiene la siguiente posición. Determina por a que casillas pude ir la torre situada en E5. Utiliza la estructura while e imprímelo por pantalla. En el caso que se encuentre con un obstáculo utiliza una excepción. No olvides documentar el código.

22. El tablero de ajedrez tiene la siguiente posición. Determina por a que casillas pude ir el alfil situada en D6. Utiliza la estructura do while e imprímelo por pantalla. En el caso que se encuentre con un obstáculo utiliza una excepción. No olvides documentar el código.

23. El tablero de ajedrez tiene la siguiente posición. Determina por a que casillas pude ir la torre situada en D5. Utiliza la estructura while e imprímelo por pantalla. En el caso que se encuentre con un obstáculo utiliza una excepción. No olvides documentar el código.

BIBLIOGRAFÍA

[1].- David Vallejo Fernandez, Cleto Martin Angelina. (2011). Desarrollo de Videojuegos: Arquitectura del Motor de Videojuegos. España: EspaCursos.

[2].- DOOM ENGINE, (s. f). En Wikipedia. Recuperado el 18 de Julio de 2015 de https://es.wikipedia.org/wiki/Doom_Engine

[3].- UNREAL ENGINE, (s. f). En Wikipedia. Recuperado el 18 de Julio de 2015 de https://en.wikipedia.org/wiki/Unreal_Engine

[4].- UNREAL 4 ENGINE FEATURES (s.f). En Unreal Engine. Recuperado el 18 de Julio de 2015 de https://www.unrealengine.com/unreal-engine-4

[5].- UNITY 3.5 FEATURES (s.f). En Unity. Recuperado el 18 de Julio de 2015 de

http://unity3d.com/unity/whats-new/unity-3.5

[5].- UNITY 4 FEATURES (s.f). En Unity. Recuperado el 18 de Julio de 2015 de

http://unity3d.com/unity/whats-new/unity-4.0

[6].- UNITY 5 FEATURES (s.f). En Unity. Recuperado el 18 de Julio de 2015 de

http://unity3d.com/unity/whats-new/unity-5.0

[7].- CryENGINE 1 FEATURES (s.f). En Unity. Recuperado el 18 de Julio de 2015 de

http://www.crytek.com/cryengine/cryengine1/overview

[8].- CryENGINE 2 FEATURES (s.f). En Unity. Recuperado el 18 de Julio de 2015 de

http://www.crytek.com/cryengine/cryengine2/overview

[9].- Kristoffer Keipp. (2009). Cryengine 3 features in detail. 18 de Julio de 2015, de Pc games hardware Sitio web: http://www.pcgameshardware.com/aid,679921/Crytek-Features-of-the-Cryengine-3-explained/News/

[10].- Claude E. Shannon. (1950). Programming a Computer for Playing Chess. Philosophical Magazine, 41, 256-275. Disponible en: http://archive.computerhistory.org/projects/chess/related_materials/text/2-0%20and%202-1.Programming_a_computer_for_playing_chess.shannon/2-0%20and%202-1.Programming_a_computer_for_playing_chess.shannon.062303002.pdf

[11].- García, L. (2015, February 11). El Parlamento impulsa el ajedrez como asignatura en España. El País, p. 1. Retrieved from http://deportes.elpais.com/deportes/2015/02/11/actualidad/1423678750_203037.html

[12].- José María Olías Porras. (2006). Desarrollar la inteligencia a través del ajedrez. España: Palabra.

[13].- NSF, CSTA, ISTE: *Operational definition of computational thinking for K–12 education*. Disponible en: http://csta.acm.org/Curriculum/sub/CurrFiles/CompThinkingFlyer.pdf