



Proyecto Final

Somos el grupo 7

Y aquí está lo que hemos realizado para este proyecto de estructuras de datos



INTRODUCCIÓN

- › El proyecto consiste en un videojuego que implementa varios temas tratados en clases como son: listas, pilas, colas, árboles e incluso funciones hash.
- › Cada tema ha servido para solventar diferentes problemas que surgieron en el desarrollo.
- › El videojuego está ambientado en los clásicos RPG.

Objetivos

- › Reforzar los temas estudiados en clase, permitiéndoles trabajar con todos los algoritmos vistos en la solución de problemas del entorno.
- › Utilizar Git como herramienta para mejorar el trabajo en equipo.
- › Desarrollar una aplicación entretenida por medio de Java y sus herramientas nativas

Conocimientos aplicados

Conocimientos previos

Programación Orientada a Objetos

- › Herencia
- › Composición/Agregación
- › Hilos
- › Canvas/Jframe
- › Events Listeners

Photoshop/Paint

GIT

Programación Web

Edición de video y audio

Conocimientos de estructuras de datos

Listas (nodos doble enlace)

Pilas

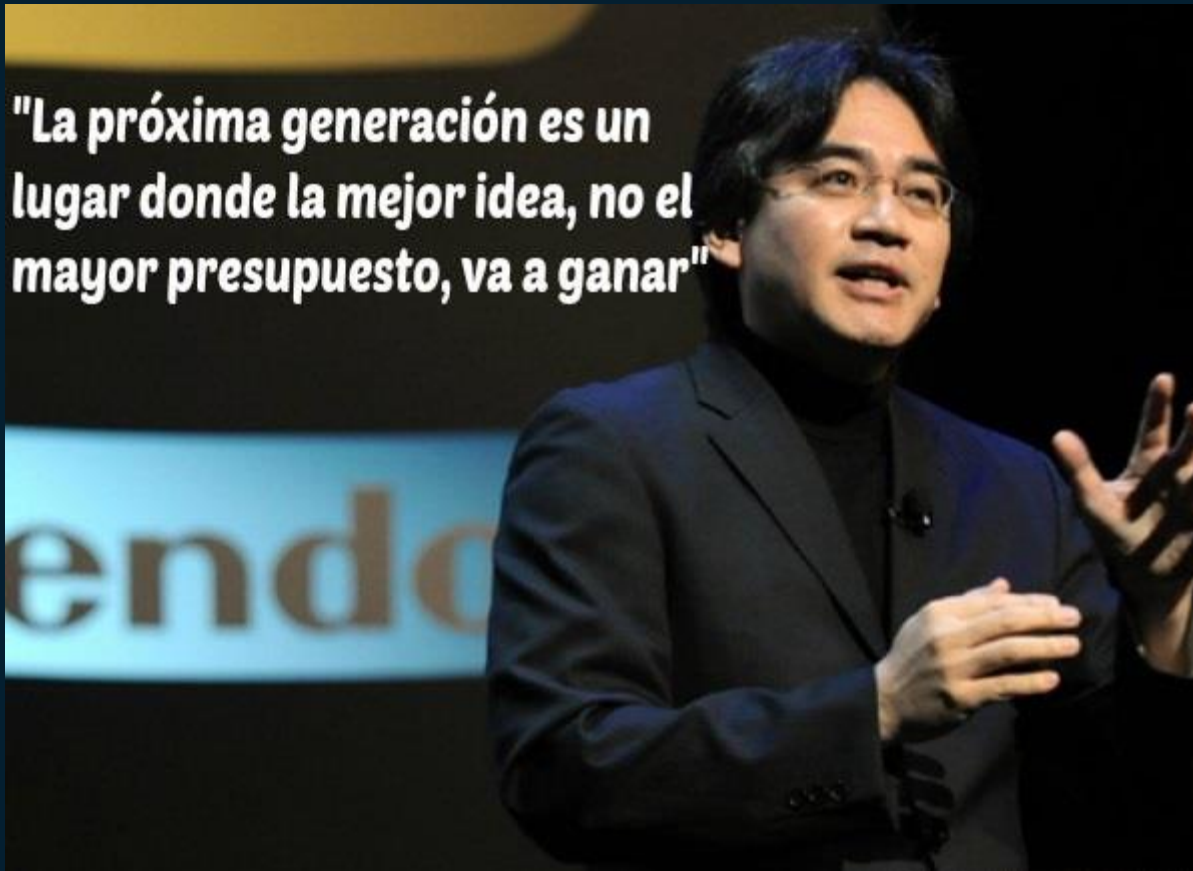
Colas

Árboles

Tablas hash

Grafos

**"La próxima generación es un
lugar donde la mejor idea, no el
mayor presupuesto, va a ganar"**



Satoru Iwada

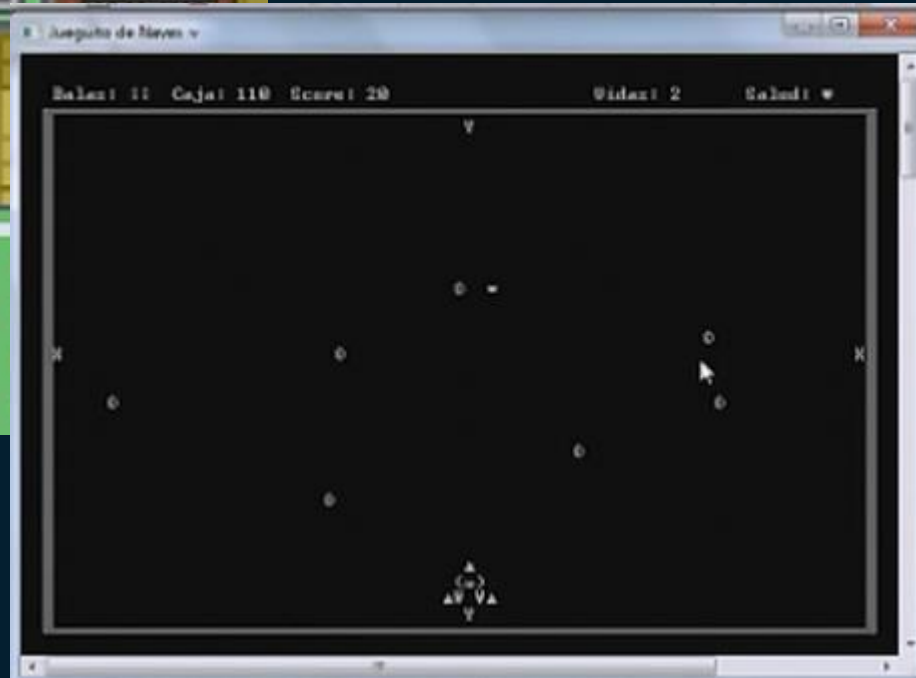
Puntos importantes del proyecto

- › Diseño de un videojuego en general
- › IA capaz de jugar Tres en raya
- › Protección de datos
- › Retroceder acciones un x tiempo
- › Toma de decisiones

Antecedentes

Contenido

1. Antecedentes
2. Términos
3. Videojuego
4. Tres en raya
5. Cheats
6. Seguridad
7. Retrocesos
8. Decisiones



Antecedentes

Contenido

1. Antecedentes
2. Términos
3. Videojuego
4. Tres en raya
5. Cheats
6. Seguridad
7. Retrocesos
8. Decisiones

Información

Juego del Tic-Tac-Toe

Símbolos:

Estudiante **X** Usted **O**

Tablero:

X	O	X
O	O	X
X	X	O

Estado actual de la partida:

EMPATE

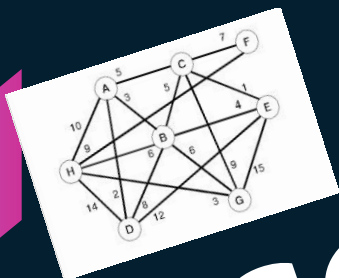
Score del juego:

Total	Estudiante	Usted	Empate
1	0	0	1

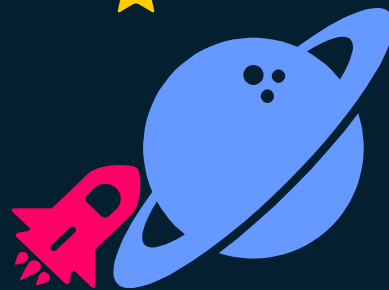
NUEVA PARTIDA



NPC COLISIÓN SPRITE



TRIGGER HITBOX



Contenido

1. Antecedentes
2. Términos
3. Videojuego
4. Tres en raya
5. Cheats
6. Seguridad
7. Retrocesos
8. Decisiones

VERSIONES

REBOOT



BUG

1.0.7

HUD

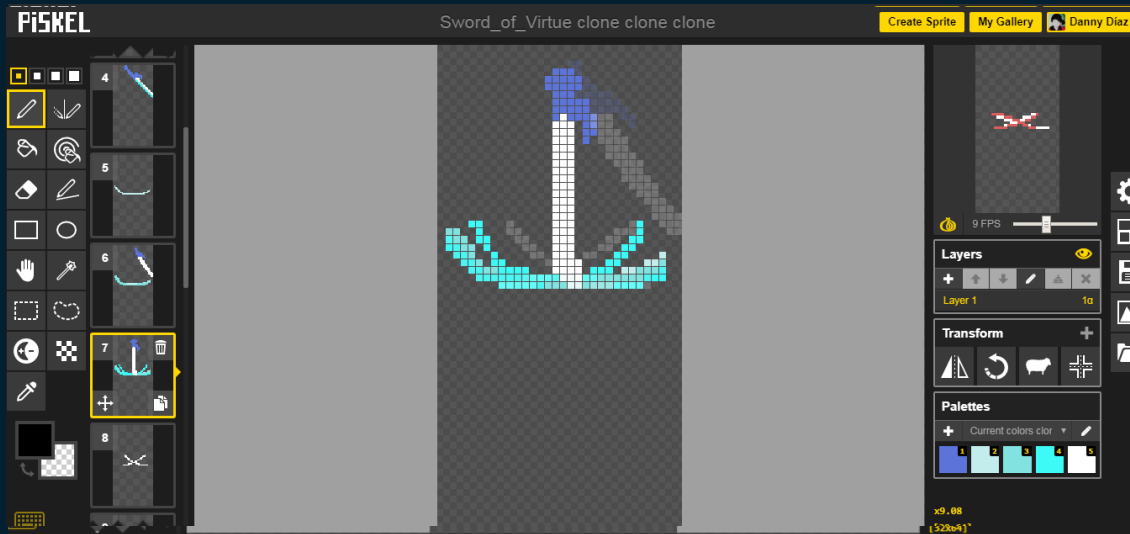
Contenido

1. Antecedentes
2. Términos
3. Videojuego
4. Tres en raya
5. Cheats
- Seguridad
6. Retrocesos
7. Decisiones

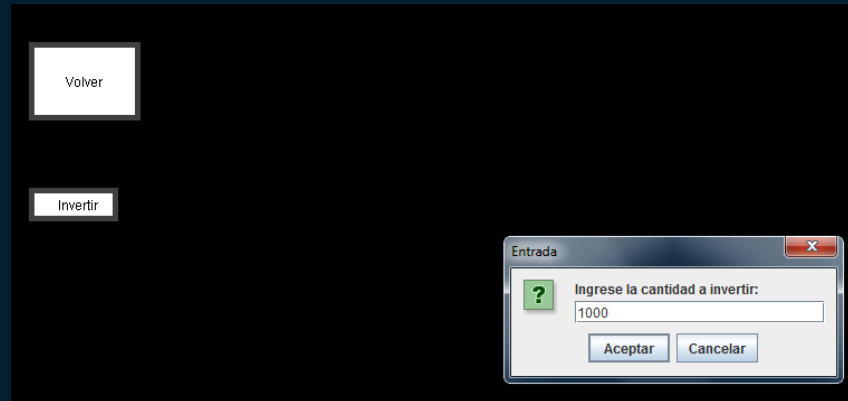
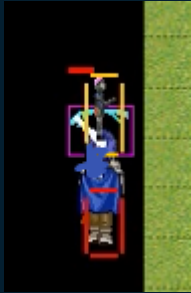
Diario de diseño

Contenido

1. Antecedentes
2. Términos
3. Videojuego
4. Tres en raya
5. Cheats
- Seguridad
6. Retrocesos
7. Decisiones



Mecánicas



Contenido

1. Antecedentes
2. Términos
3. Videojuego
4. Tres en raya
5. Cheats
- Seguridad
6. Retrocesos
7. Decisiones



1000\$

De inversion en 7 dias genera

1300\$

Tycoon

Estrategia **capitalista** para enganchar a los usuarios

Contenido

1. Antecedentes
2. Términos
- Videojuego**
3. Tres en raya
4. Cheats
5. Seguridad
6. Retrocesos
7. Decisiones

Bala



Se implemento un elemento pistola que permite disparar balas a los enemigos.

Boss final

Un jefe final que aparece al matar cierta cantidad de enemigos, posee mucha vida



Tienda

Un lugar donde el jugador puede comprar objetos útiles para su aventura.



Contenido

1. Antecedentes
2. Términos
3. Videojuego
4. Tres en raya
5. Cheats
6. Seguridad
7. Retrocesos
8. Decisiones

COLISIONES



Para implementar colisiones se usó hitbox, que fueron implementados con la clase Rectangle de java.

HUD

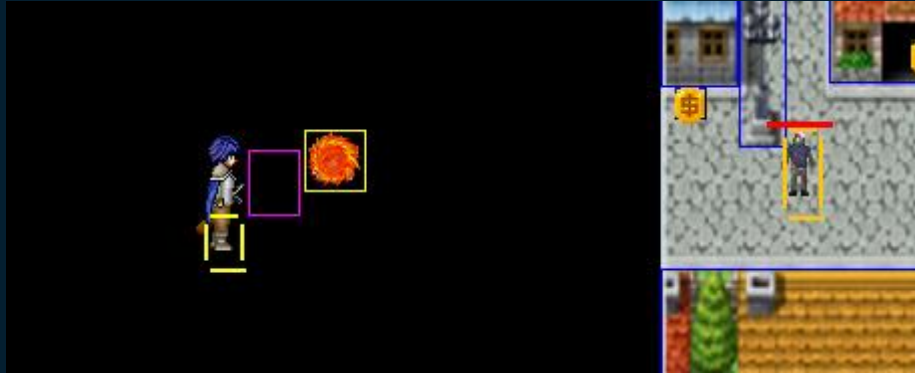


El HUD fue implementado con una imagen previa que sirve de fondo, y la clase Graphics con sus métodos nos ayudan a dibujar el resto de la interface.

Contenido

1. Antecedentes
2. Términos
3. Videojuego
4. Tres en raya
5. Cheats
6. Seguridad
7. Retrocesos
8. Decisiones

Bola de fuego



Implementación similar a la clase bala, la diferencia está en que esta habilidad es ilimitada, solo dependiendo del mana del jugador

Contenido

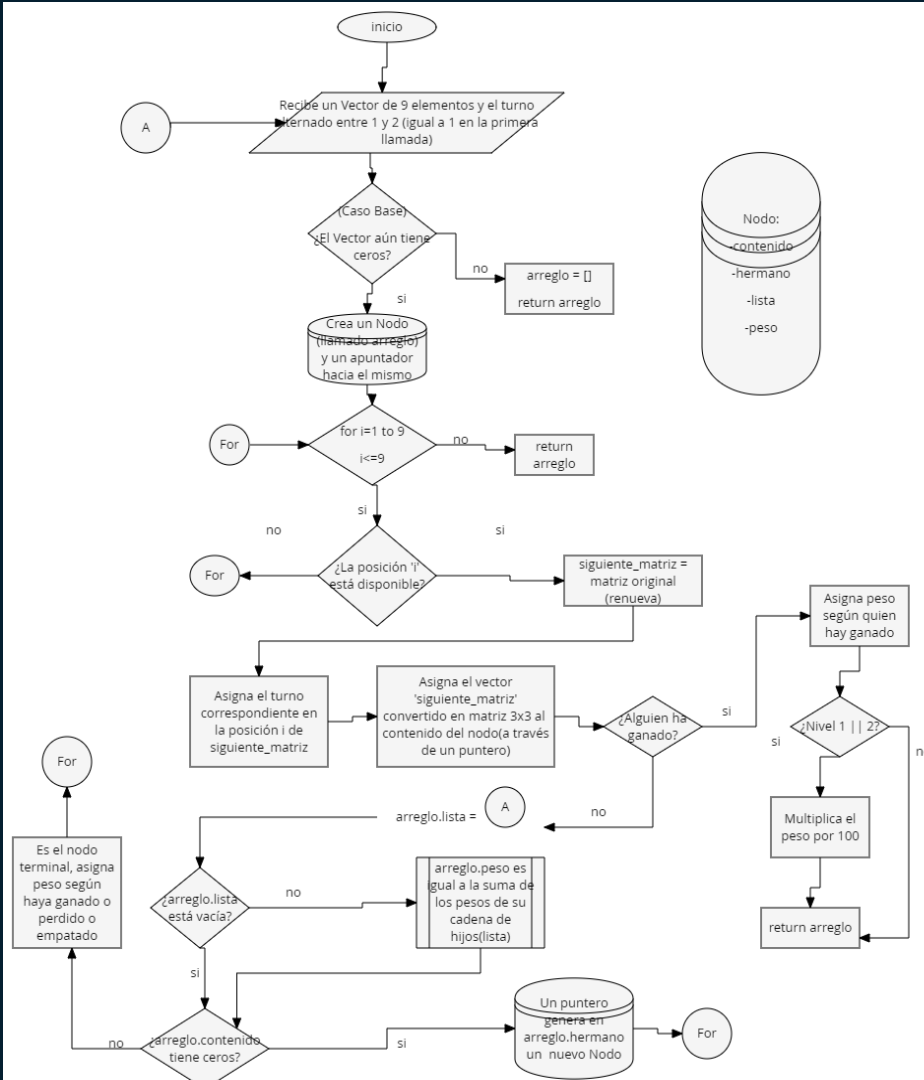
1. Antecedentes
2. Términos
3. Videojuego
4. Tres en raya
5. Cheats
6. Seguridad
7. Retrocesos
8. Decisiones

DEMOSTRACIÓN

Contenido

1. Antecedentes
2. Términos
3. Videojuego
4. Tres en raya
5. Cheats
6. Seguridad
7. Retrocesos
8. Decisiones

Diagrama de flujo del Tres en raya



Contenido

1. Antecedentes
2. Términos
3. Tres en raya
4. Cheats
5. Seguridad
6. Retrocesos
7. Decisiones

Estrategia mejorada

1. Antecedentes

2. Términos

Videojuego

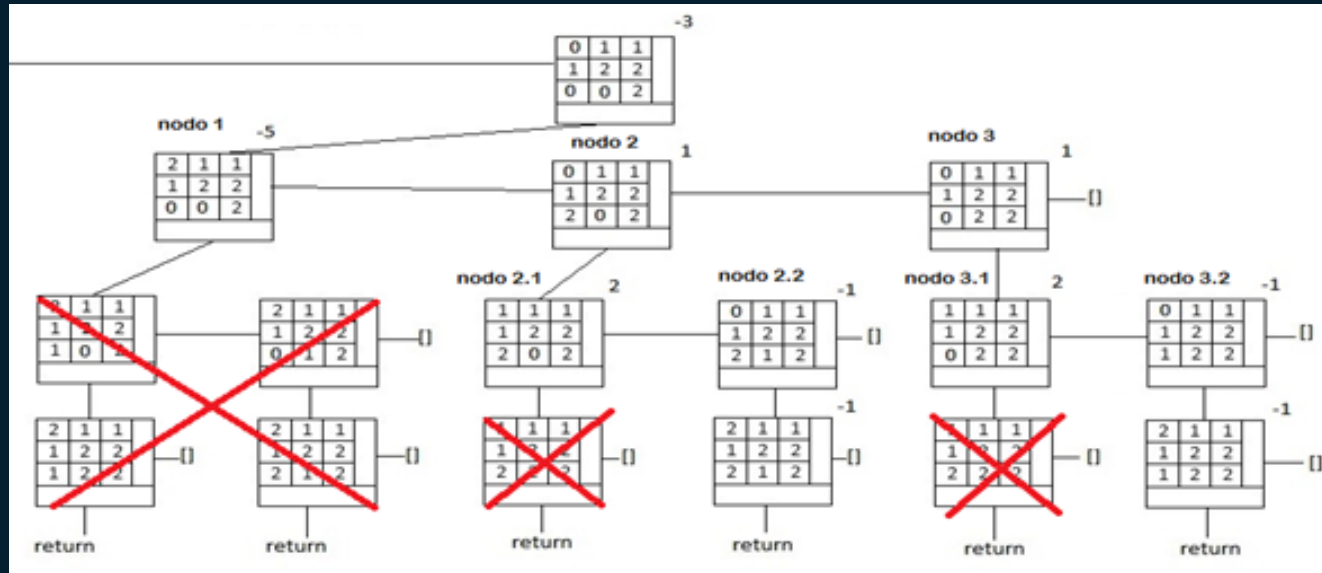
3. Tres en raya

4. Cheats

5. Seguridad

6. Retrocesos

7. Decisiones



Cantidad de nodos que genera la matriz con 'n' espacios disponibles (demostración)

Contenido

1. Antecedentes

2. Términos

Videojuego

3. Tres en raya

4. Cheats

5. Seguridad

6. Retrocesos

7. Decisiones

De manera analítica se obtuvo una fórmula para obtener la cantidad de nodos que cada padre generaba (esto es parte fundamental para la estrategia que usamos), para tres espacios disponibles (n=3) es:

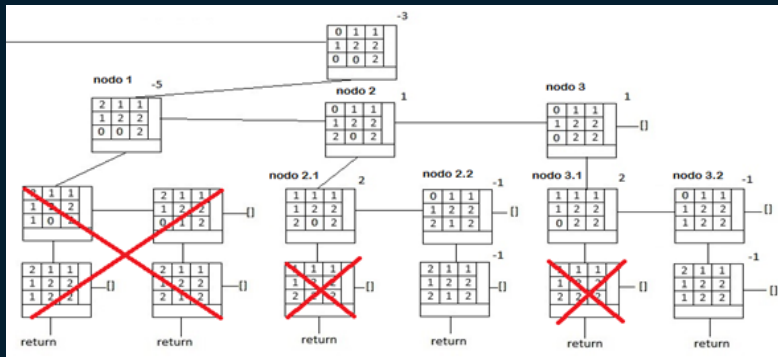
$$1+(n-1)+(n-1)(n-2)]*n$$

De manera general:

$$\begin{aligned} & [1+(1)(n-1)+\dots+(1)(n-1)(n-2)\dots(n-(n-1))] * n = n + n(n-1) + n(n-1)(n-2) \\ & = n(n-1)! / (n-1)! + n(n-1)(n-2)! / (n-2)! + n(n-1)(n-2)(n-3)! / (n-3)! \\ & = n! (1/(n-1)! + 1/(n-2)! + 1/(n-3)!) \end{aligned}$$

$$= n! * \sum_{k=1}^n \frac{1}{(n-k)!}$$

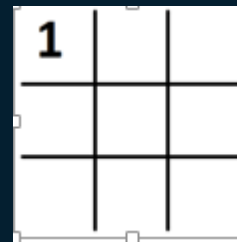
En definitiva mejora el procesamiento



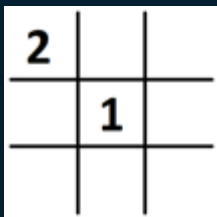
Estrategias para mejorar aún más el rendimiento: 1era y 2da

Primera Jugada:

Los jugadores más experimentados colocan la primera jugada en una esquina cada vez que empiezan primero. Esto le da al oponente mayor oportunidad para cometer un error. Si el oponente responde colocando en cualquier parte que no sea el centro, las probabilidades de victoria aumentan.



Segunda Jugada:



Si el oponente juega primero y comienza en una esquina, siempre se coloca la segunda jugada en el centro. La segunda jugada no debe ubicarse en una esquina contraria, porque perderíamos automáticamente, sin vuelta atrás. Con esta estrategia, se estima que el juego terminará seguramente en empate.

Contenido

1. Antecedentes

2. Términos

Videojuego

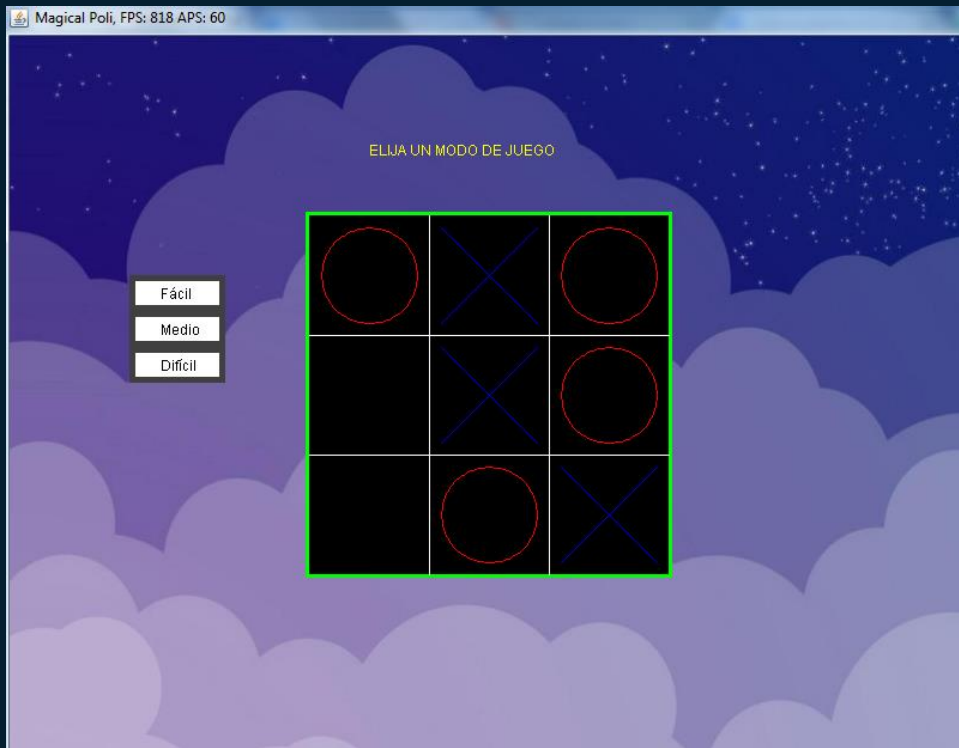
3. Tres en raya

4. Cheats

5. Seguridad

6. Retrocesos

7. Decisiones



Contenido

1. Antecedentes

2. Términos

Videojuego

3. Tres en raya

4. Cheats

5. Seguridad

6. Retrocesos

7. Decisiones

Contenido

1. Antecedentes

2. Términos

Videojuego

3. Tres en raya

4. Cheats

5. Seguridad

6. Retrocesos

7. Decisiones





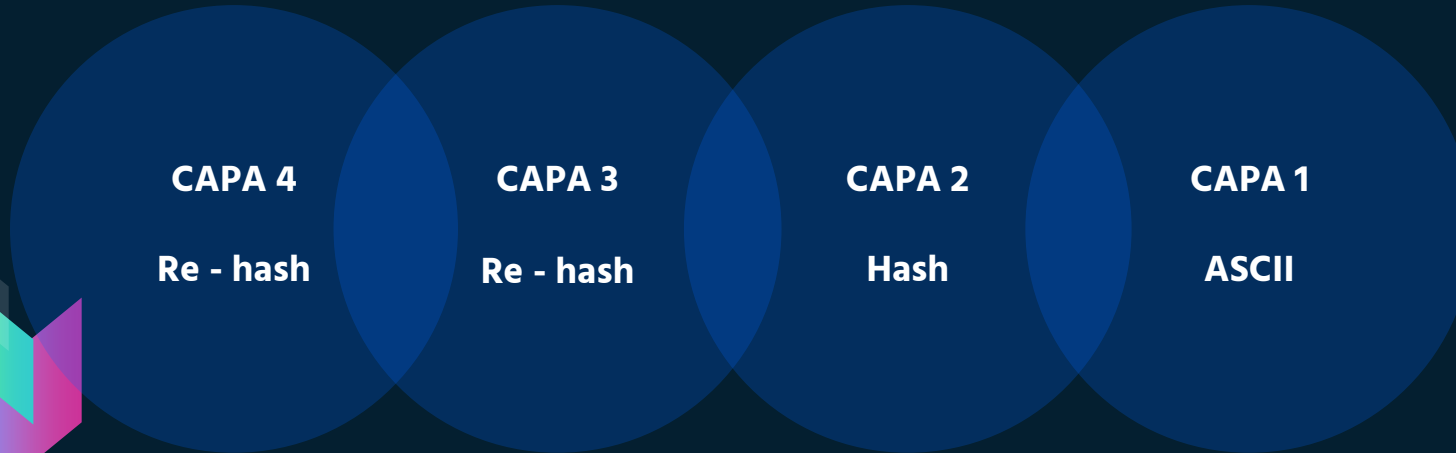
87->
87->
87->87->
87->87->87->
87->87->87->68->
87->87->87->68->68->
87->87->87->68->68->68->83->
87->87->87->68->68->68->83->83->
87->87->87->68->68->68->83->83->83->
87->87->87->68->68->68->83->83->83->65->
87->87->87->68->68->

Lista de Teclas, solo se guardan si coinciden parcialmente o totalmente con los cheats

Contenido

1. Antecedentes
2. Términos
3. Tres en raya
4. Cheats
5. Seguridad
6. Retrocesos
7. Decisiones

Encriptación mediante hashing



Contenido

1. Antecedentes
2. Términos
- Videojuego
3. Tres en raya
4. Cheats
5. Seguridad
6. Retrocesos
7. Decisiones

Contenido

1. Antecedentes
2. Términos
- Videojuego
3. Tres en raya
4. Cheats
5. Seguridad
6. Retrocesos
7. Decisiones

Ejemplos de encriptacion

	Usuario 1	Usuario 2	Usuario 3
Capa 1	495051	979899	100101102
Capa 2	2	10	3
Capa 3	3	1	4



Contenido

1. Antecedentes

2. Términos

Videojuego

3. Tres en raya

4. Cheats

5. Seguridad

6. Retrocesos

7. Decisiones

Viaje en el tiempo

ESTADO 1
Vida = 100
Sprite = frente



ESTADO 2
Vida = 50
Sprite = espalda



ESTADO 3
Vida = 10
Sprite = derecha



Contenido

1. Antecedentes
2. Términos
- Videojuego**
3. Tres en raya
4. Cheats
5. Seguridad
- 6. Retrocesos**
7. Decisiones

Contenido

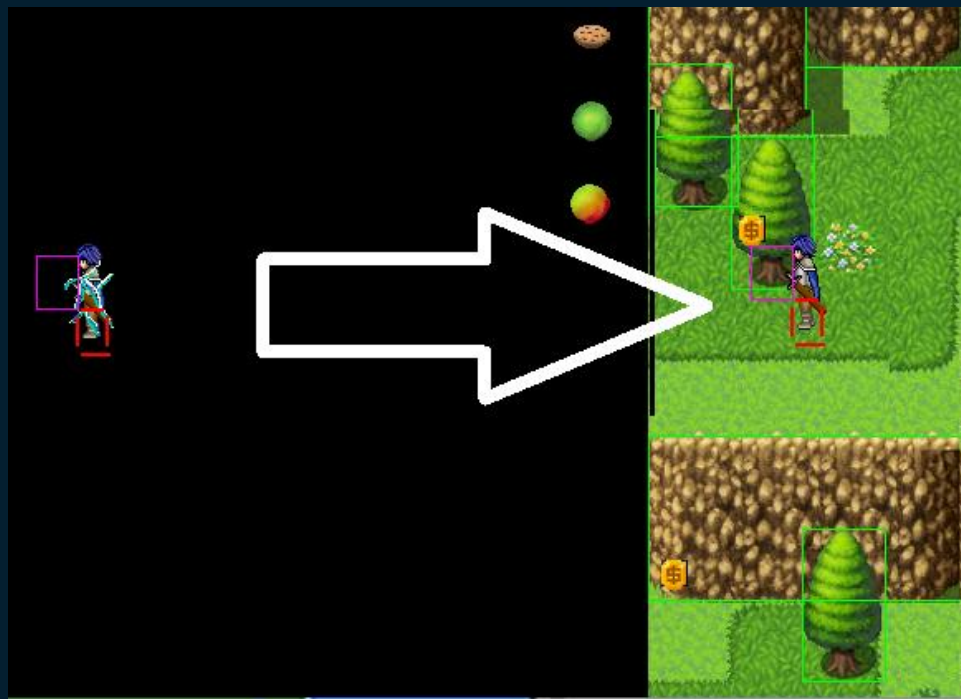
1. Antecedentes
2. Términos

Videojuego

3. Tres en raya
4. Cheats
5. Seguridad

6. Retrocesos

7. Decisiones



Contenido

1. Antecedentes

2. Términos

Videojuego

3. Tres en raya

4. Cheats

5. Seguridad

6. Retrocesos

7. Decisiones



Contenido

1. Antecedentes

2. Términos

Videojuego

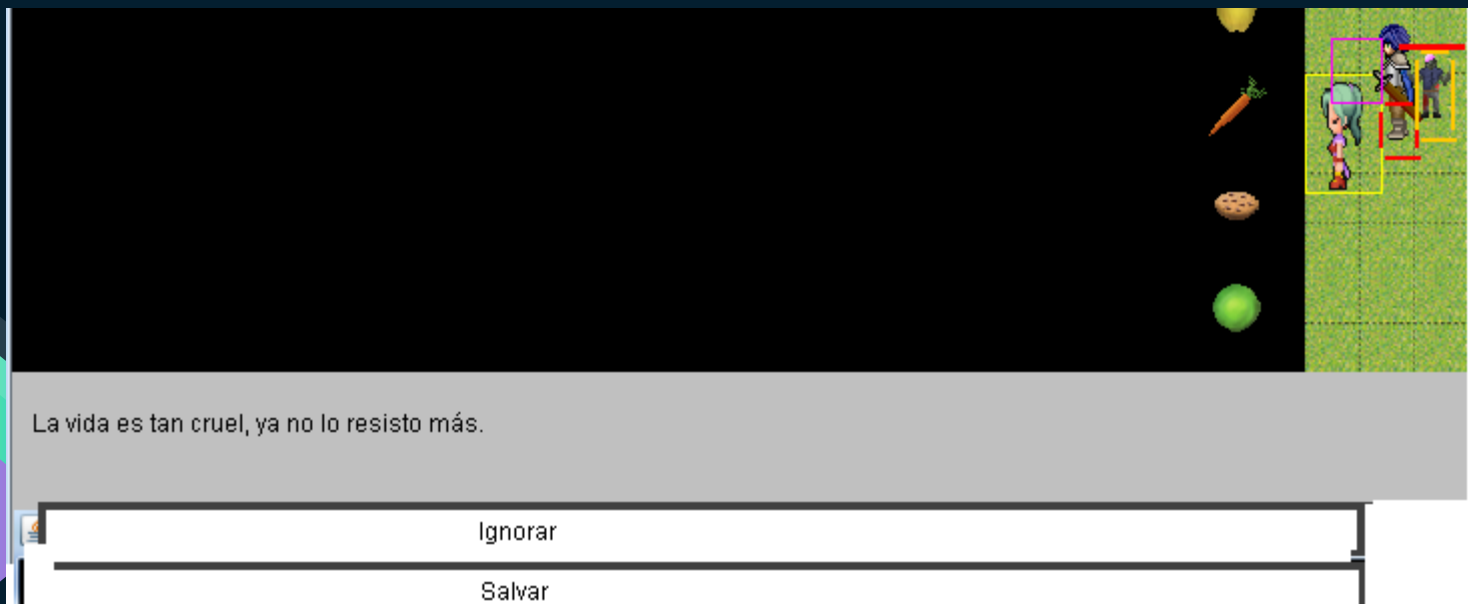
3. Tres en raya

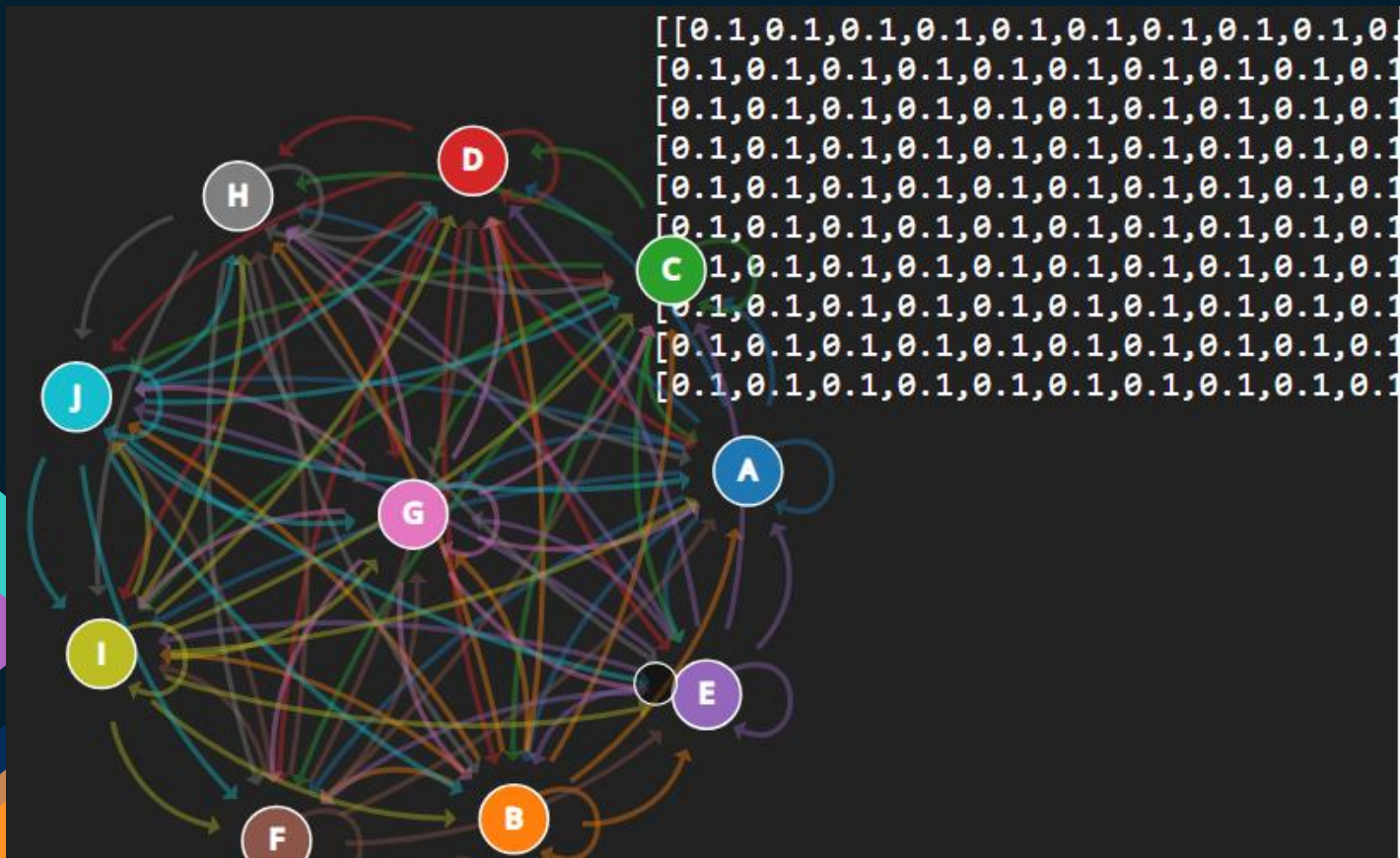
4. Cheats

5. Seguridad

6. Retrocesos

7. Decisiones





Contenido

1. Antecedentes
2. Términos
3. Tres en raya
4. Cheats
5. Seguridad
6. Retrocesos
7. Decisiones



KarmaB: 0 KarmaM: 0

BALAS: 5/30

BALAS: 10/250

Contenido

1. Antecedentes
2. Términos

Videojuego

3. Tres en raya
4. Cheats
5. Seguridad
6. Retrocesos
7. Decisiones

FINAL

1. Las pilas permiten almacenar varios estados en el tiempo de tal forma que podemos volver a un estado inicial. Esto permitió el "viaje en el tiempo".
2. Las listas guardan información de una manera casi limitada. Están acotadas por la cantidad de memoria RAM y los niveles de caché de un computador.
3. Los arboles pueden dar un vistazo al futuro en juegos "cerrados" es decir con un límite de jugadas posibles.
4. Para desarrollar una IA que pueda jugar ajedrez se requiere un cómputo bastante significativo es por eso que técnicas como el reinforcement learning son mucho más útiles que crear árboles de decisión.
5. Tomar decisiones que afectan el futuro del juego afectan personalmente al usuario de forma que se "sumerge" mucho más a la aventura

Conclusiones

Recomendaciones

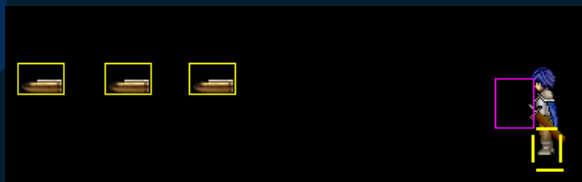
Logramos aplicar los conocimientos de
Estructuras de datos y programación 1 y 2

FINAL

Retorno en el tiempo



Objetos arrojadizos



Encriptacion Hash

A screenshot of a login window titled "Logueo". It has two input fields: "Nickname:" with the text "Rafa17" and "Contraseña:" with masked characters ".....". Below the fields are two buttons: "Ingresar" and "Salir".

Conclusiones

Recomendaciones


GIT

FINAL



Conclusiones

Recomendaciones

- 
1. Se recomienda el uso de matrices para la representación del mapa mediante cuadrados definidos, al cargar simplemente el mapa si modelarlo resulta difícil la implementación de algoritmos para el movimiento e inteligencia de enemigos y NPCs.
 2. Se recomienda el uso de la función System para usar la tarjeta gráfica de la computadora y mejorar el rendimiento.
 3. Para un mejor desarrollo de videojuegos se recomienda el uso de una IDE de desarrollo diseñada específicamente para este objetivo como GameStudio ya que esto facilita la implementación de interacciones y la creación de elementos dentro de un videojuego

FINAL

Conclusiones

Recomendaciones

Enlaces útiles

1. Dibujar sprites 2d: <https://www.piskelapp.com>
2. Aumentar y reducir el volumen de las canciones: <http://www.mp3louder.com/es/>
3. Algoritmo Floyd Warshall: <https://www.youtube.com/watch?v=h-nmexY9gtA>, Algoritmo de Floyd Warshall para la ruta más corta
4. Recortador de audio online: <https://audiotrimmer.com/es/>
5. Java-Eventos de mouse: <https://www.youtube.com/watch?v=9Zflg2FnVjo>
6. CURSO Juego de rol 2D en java:
<https://www.youtube.com/watch?v=qa6GA5p9nQ0&list=PLN9W6BC54TJJr3erMptodGOQFX7gWfKTM>
7. Curso de programación web básica: <https://www.youtube.com/watch?v=5xgPUZU1EYk>
8. Recursos 2d gratis: <https://opengameart.org/content/tiny-16-basic>
9. Curso GIT:
<https://www.youtube.com/watch?v=zH3I1DZNovk&list=PL9xYXqvLX2kMUrXTvDY6GI2hgacfY0rId>
10. Desarrollo de videojuegos: <https://www.youtube.com/watch?v=eMMnaUmWtnw>

Referencias

1. Johnsonbaugh, R. (2005). Árboles de juegos. En R. Johnsonbaugh, *Matemática Discreta, sexta edición* (págs. 429-434). México: PEARSON EDUCACIÓN.
2. Tomás Blanco. *PARA JUGAR COMO JUGÁBAMOS*. 2003. ISBN 84-87339-44-1 Ed. John Wiley & sons.
3. Cairo Osvaldo, Guardati Silvia, Estructura de datos, ISBN: 970105908-5, Tercera Edición 2006. [Online]. Disponible en: https://drive.google.com/file/d/0B_XimPSyUDLcM2ZtU3VCVHhLUUk/edit?pref=2&pli=1
4. Estructura de Datos en Java, JOYANES Luis, ZAHONERO Ignacio. ISBN: 9788448173937, Edición 2008. Disponible en: ftp://soporte.uson.mx/PUBLICO/02_ING.SISTEMAS.DE.INFORMACION/estructura_datos/Estructura%20de%20datos%20en%20java%20Joyanes%201ed.pdf
5. Estructura de Datos, GARCIA Iván, GARCIA Magariño, ISBN: 8445419358 ISBN-13, Edición 2011.
6. Curso de creación de un juego de rol 2d en java: <https://www.youtube.com/watch?v=qa6GA5p9nQ0>
7. Curso de GIT de CodigoFacilito: <https://www.youtube.com/watch?v=zH3l1DZNovk>



1. carlos.chicaiza02@epn.edu.ec
2. danny.diaz@epn.edu.ec/
3. rafael.vinueza@epn.edu.ec
4. Página web: <https://magody.github.io/>