Estructuras de Datos

Proyecto Segunda Evaluación – 2023 PAO 2
"Tres en raya" contra el computador

LEA ESTE DOCUMENTO DETENIDAMENTE

Este es un proyecto **grupal**. Por tanto, todos los miembros del grupo deben involucrarse en el diseño e implementación de la solución que se solicita. La entrega del proyecto incluirá un reporte de auto y coevaluación, donde cada miembro del grupo calificará sus contribuciones al proyecto y las de sus compañeros.

Es su responsabilidad agruparse a tiempo para la entrega del proyecto. Usted debe indicar su grupo en el Aula Virtual hasta el martes 12 de diciembre. Los estudiantes que no se hayan unido a ningún grupo para entonces, serán asignados aleatoriamente por el profesor. A partir de entonces, no se aceptarán cambios en la conformación de los grupos. Usted es el único responsable de unirse a un grupo y asegurarse que dicho paso se haya ejecutado exitosamente. **EN NINGUNA CIRCUNSTANCIA SE ACEPTARÁN ENTREGAS INDIVIDUALES.**

Si algún grupo experimenta el incumplimiento sistemático de alguno de sus miembros, esta situación debe ser reportada al profesor **TAN PRONTO COMO SE PRESENTE**. Cualquier estudiante cuyo incumplimiento sea comprobado, será retirado del grupo y **DEBERÁ IMPLEMENTAR EL PROYECTO DE MANERA INDIVIDUAL SOBRE EL 50% DEL PUNTAJE POSIBLE.**

Si algún estudiante siente que otro (u otros) toma(n) el control del grupo y no permite(n) a los demás miembros hacer aportes o colaborar en el proyecto, debe reportar esta situación al profesor **TAN PRONTO COMO SE PRESENTE**. Cualquier estudiante que pretenda monopolizar el trabajo del proyecto o impida a los demás hacer aportes será separado del grupo y **DEBERÁ IMPLEMENTAR EL PROYECTO DE MANERA INDIVIDUAL Y SOBRE EL 50% DEL PUNTAJE POSIBLE.**

Si algún estudiante reporta de manera extemporánea cualquiera de las dos situaciones detalladas en los dos párrafos precedentes, será penalizado con su 25% de la nota total obtenida en el proyecto. Reporte novedades al profesor **TAN PRONTO COMO ÉSTAS SE PRESENTEN**.

Considere las políticas de buena conducta académica que se explicaron en la primera clase del curso respecto al plagio y demás violaciones del código de Ética de la ESPOL. Los autores del proyecto deben ser usted y sus compañeros de grupo (no otras personas, **ni motores de Inteligencia Artificial**). Cualquier indicio de lo contrario, será reportado a las unidades correspondientes para el tratamiento pertinente.

Finalmente, **siga las instrucciones de este documento** para evitar inconvenientes. Si tiene dudas, consulte al profesor en lugar de asumir cosas que pueden ser incorrectas.

El incumplimiento de instrucciones explícitas indicadas en este documento o indicadas en nuestras sesiones teóricas derivará, de manera inapelable, en la penalización y rebaja de puntos sobre su entrega.

Introducción

"Tres en Raya" es un juego en el que dos jugadores (representados por los símbolos **O** y **X**), marcan los espacios de un tablero de 3×3 de manera alternada. En cada turno, cada jugador debe colocar su símbolo una vez sobre una casilla libre. El primer jugador que logra llenar una fila, columna, o diagonal con tres de sus símbolos, gana el juego.

La siguiente secuencia de una partida de tres en raya es iniciada por el jugador **X**. Cada tablero mostrado corresponde a un turno distinto. En este ejemplo, **X** gana la partida:5

Ejemplo de una partida que termina en empate (**X** empieza el juego):

En este proyecto, usted implementará una aplicación gráfica de JavaFX que permitirá a una persona jugar el "Tres en Raya" **contra la computadora**. A lo largo de la implementación de su solución, usted y sus compañeros de grupo deberán decidir cuáles de las estructuras de datos revisadas en el curso son las más apropiadas para implementar distintas partes del juego. En combinación con otros elementos, algunas de estas estructuras se utilizarán para dotar a la computadora de "inteligencia" para que pueda jugar el "Tres en Raya" en contra de un jugador humano.

Utilidad de un Tablero

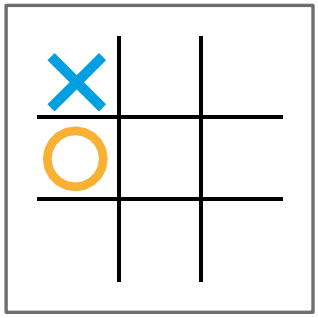
Al igual que los jugadores humanos, en un juego, la computadora debe "decidir" qué movimiento realizar cada vez que sea su turno. Como no es factible calcular de antemano todos los posibles movimientos que podrían derivar en una jugada ganadora, la computadora debe tomar una decisión óptima **en cada turno**. Dicha decisión debe considerar el estado actual del tablero y una función de utilidad **u** que indica lo cerca que se está de una jugada ganadora (o perdedora). Al considerar los valores retornados por esta función de utilidad, la computadora puede decidir qué movimiento es más conveniente dado un estado específico del tablero.

Para el "Tres en Raya", dato un tablero **t**, la función de utilidad se define como:

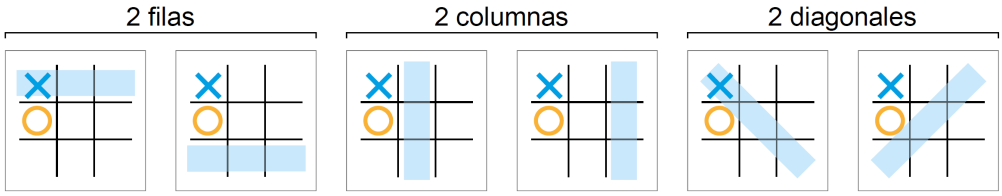
$$u_{jugador}(t) = P_{jugador} - P_{oponente}$$

donde P_j es el número total de filas, columnas y diagonales disponibles en el tablero **t** para el jugador **j**.

Considere el siguiente tablero:

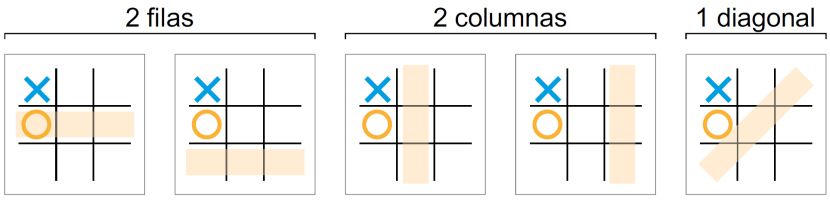


El cálculo de los valores de **P** para cada jugador se ilustra en las figuras mostradas a continuación:



P(X) = 2 + 2 + 2

P(X) = 6



P(O) = 2 + 2 + 1

P(O) = 5

Por tanto, la utilidad del tablero mostrado para el jugador que juega con el símbolo **X** sería:

$\mathbf{u_X} = P_X - P_O$
$\mathbf{u_X} = 6 - 5$
$\mathbf{u_X} = 1$

Decidiendo Jugadas

Utilizando la función de utilidad descrita en la sección anterior, su proyecto dotará a la computadora de una estrategia de decisión para jugar el "Tres en Raya". Dicha estrategia puede resumirse en la siguiente frase:

> "*Computadora, elije el mejor movimiento para ti misma, asumiendo que el humano escogerá el peor para ti*".

Esta filosofía, aplicada ampliamente en el mundo de la inteligencia artificial y los juegos, se conoce con el nombre de *minimax*. En términos matemáticos, *minimax* consiste en **mini**mizar la pérdida **máx**ima esperada.

En el proyecto que usted implementará, cada vez que le toque jugar, la computadora aplicará la estrategia minimax para decidir qué movimiento realizar.

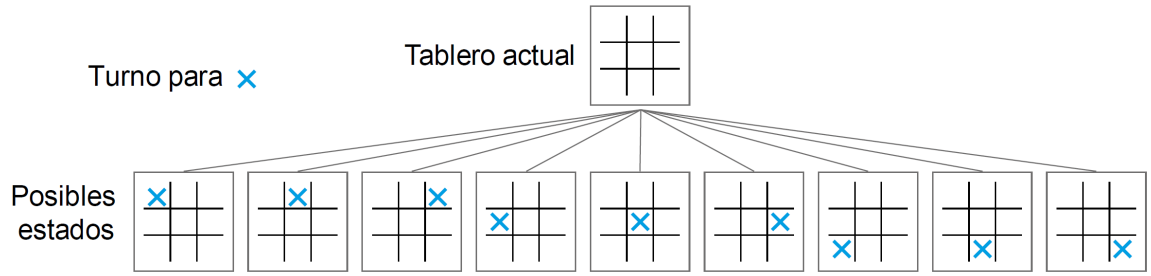
Para el "Tres en Raya", el *algoritmo* de minimax consiste en:

1. Generar los posibles estados que un tablero podría tomar después de dos turnos (el propio y el del oponente).
2. Calcular la utilidad de cada uno de los tableros que podrían ser generados por el oponente.
3. Encontrar la utilidad mínima de cada familia de tableros que podría generar el oponente y asociar esa utilidad mínima al padre respectivo.
4. Elegir el tablero propio (es decir, el tablero del jugador que tiene el turno) con la máxima de todas las utilidades mínimas y efectuar el movimiento que lleve a este tablero.

En las páginas siguientes, se describe con más detalle cada uno de estos pasos.

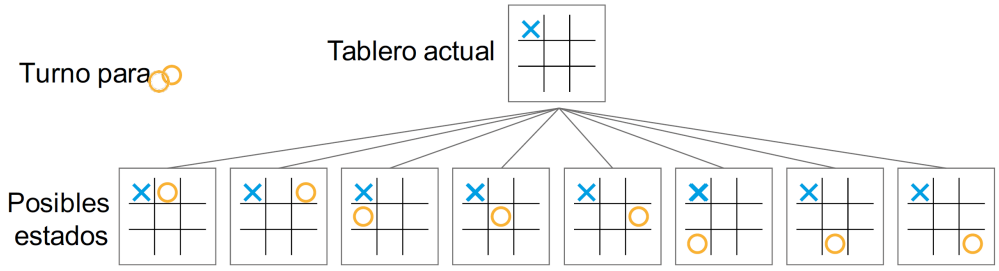
Paso 1: Generar Posibles Estados

Generar los posibles estados de un tablero significa producir todos los tableros que podrían resultar a partir del tablero actual, dado un turno específico. Por ejemplo, si el tablero actual está vacío y le toca el turno al jugador con el símbolo **X**, hay nueve posibles tableros (es decir, estados) que podrían generarse. Estos se muestran en la siguiente figura:



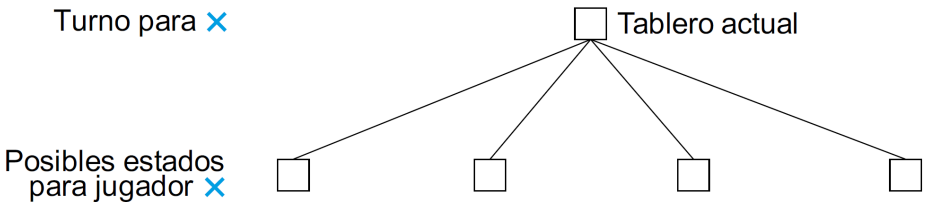
Lo de arriba ocurre, porque en un tablero vacío, hay nueve posibles lugares donde el jugador con el símbolo X podría jugar y, cada uno de estos lugares, derivaría en un tablero distinto.

Para el tablero mostrado a continuación, si el turno es para el jugador con el símbolo **O**, habría únicamente ocho posibles estados:

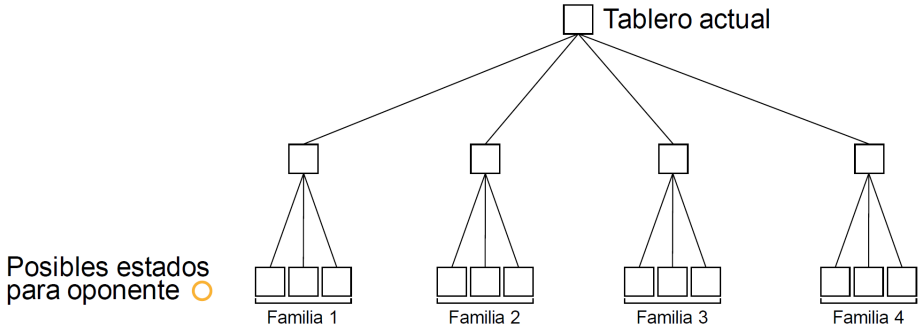


El algoritmo de minimax indica que se deben generar los estados del tablero considerando **dos turnos**. Esto significa que se deben generar los posibles estados que resulten del turno actual y, por cada uno de estos, los tableros que podrían resultar por el turno del oponente. Las siguientes figuras ilustran esto:

Primeros estados generados:



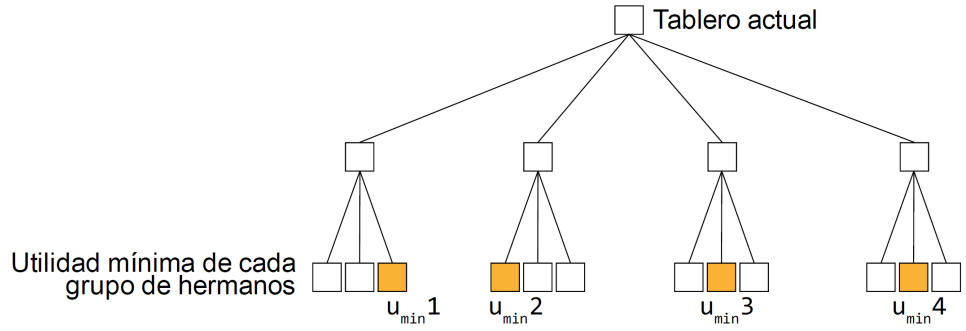
Segunda generación de estados:



Como se puede apreciar, el Paso 1 genera un árbol n-ario. Es decir, un árbol en el que cada uno de sus nodos puede tener cero o más hijos.

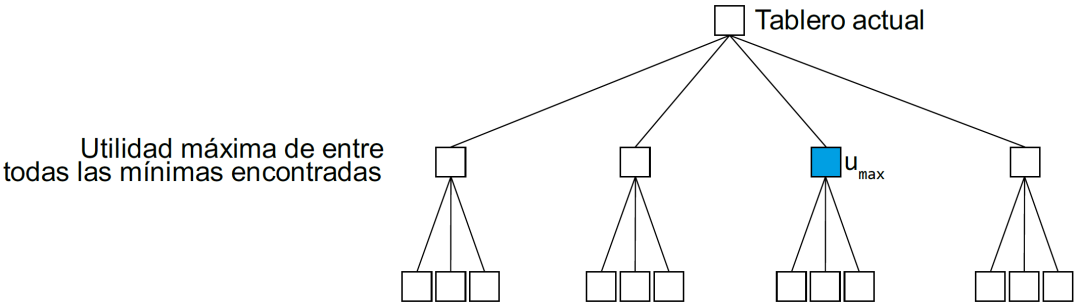
Pasos 2 y 3: Utilidades Mínimas en Tableros que Podría Generar el Oponente

El siguiente paso consiste en calcular las utilidades de cada uno de los tableros que podrían ser generados por el oponente (las hojas del árbol mostrado en el paso anterior). Considerando estos valores, se encontrará el tablero que garantice la **utilidad mínima** de cada grupo de hermanos (es decir, de cada familia). El valor mínimo de cada familia se ilustra en la figura de abajo mediante los tableros pintados en naranja, cada uno de los cuales tiene una utilidad u_{min}.



Paso 4: Utilidad Máxima en Tableros que Podría Generar el Jugador

Una vez que se conoce la utilidad mínima de cada familia de tableros, este valor se asocia al padre correspondiente. Finalmente, de todos los tableros que podrían ser generados por el jugador con el turno actual, se escoge aquel tablero que genera la máxima utilidad.

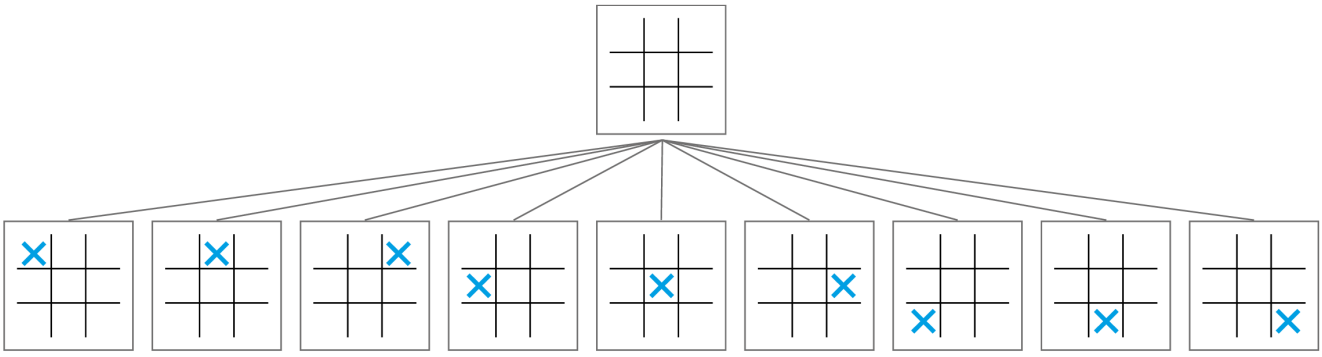


El tablero encontrado (mostrado en celeste en la imagen de arriba) indica el movimiento que debe realizar el jugador en su turno.

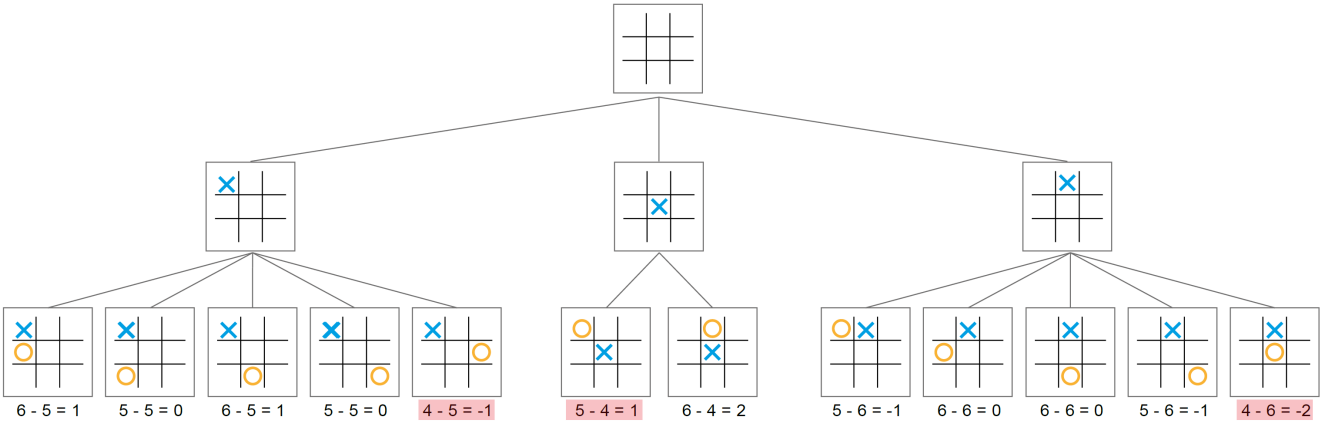
Ejemplo Ilustrativo

El algoritmo descrito en la sección anterior se ilustra aquí con un ejemplo concreto, que inicia con un tablero vacío. El turno inicial corresponde a la computadora, quien juega con el símbolo **X**. Para decidir en qué posición le conviene marcar su símbolo, la computadora aplicará el algoritmo minimax, de acuerdo con lo descrito anteriormente.

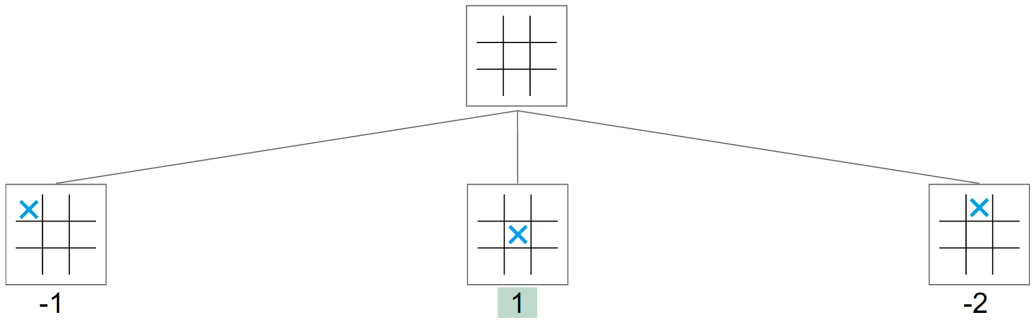
En primer lugar, se generan los posibles tableros que podría producir la jugada de la computadora:



Por **cada uno** de estos tableros, se generan aquellos que podrían resultar del movimiento hecho por el oponente (es decir, por el jugador humano). **Una muestra** de estos tableros se muestra en la siguiente figura. Por restricciones de espacio, no se muestran todos los posibles 72 tableros del segundo nivel ni los 9 del primero (que sí se muestran en la figura anterior).



La imagen anterior incluye también los valores de la función de utilidad de cada uno de los tableros del segundo nivel del árbol. Los valores resaltados con rojo indican el mínimo dentro de cada familia. Este valor mínimo es luego asociado a cada uno de los padres, como se muestra en la siguiente figura:



En este punto, la computadora debe escoger al tablero que garantiza la mayor utilidad. En el ejemplo dado, es el tablero con utilidad 1, que se encuentra en el centro de la figura anterior y cuya utilidad aparece resaltada con color verde. Esto significa que, dado un tablero vacío, a la computadora le conviene colocar su símbolo en la celda central del tablero, ya que este movimiento maximiza su ganancia potencial y, al mismo tiempo, minimiza las posibles de perder.

Modelamiento del Problema y Requerimientos Mínimos

Como probablemente habrá notado, el escenario descrito arriba puede ser modelado con árboles n-arios en combinación con otros TDAs lineales. Dados los objetivos que se persiguen en este curso, usted y sus compañeros de grupo serán responsables de implementar los TDAs que consideren pertinentes para la solución de este problema. Sin embargo, usted puede utilizar todas las colecciones lineales del Java Collection Framework (listas, pilas, colas, conjuntos, y mapas), pero es responsable de implementar toda estructura no lineal que vaya a utilizar. Un requisito obligatorio de este proyecto es utilizar árboles. **Una solución que no utilice árboles invalidaría su proyecto y le otorgaría una nota automática de cero (0) a todos los miembros del grupo. Este es un curso de Estructuras de Datos. Por tanto, usted debe usar estructuras (las más apropiadas) para resolver el problema propuesto.**

Su juego debe consistir en una interfaz gráfica de JavaFX que permita al usuario jugar contra la computadora indicando inicialmente:

1. Los símbolos que cada jugador tendrá (es decir, quién será X y quién será O)
2. Quién inicia el juego (el humano o la computadora)

Estos parámetros deben ser indicados por el usuario utilizando controles gráficos. Usted debe decidir cómo su interfaz le permitirá al usuario efectuar estás operaciones. Asimismo, deberá decidir cómo el usuario humano indicará dónde colocar un símbolo en el tablero. Asegúrese de diseñar interacciones que sean fáciles y amigables (por ejemplo, dando click directamente sobre la casilla que se desea rellenar).

Su programa debe indicar, al final, quién ganó el juego o si hubo un empate.

Funcionalidad Opcional

Las funcionalidades descritas arriba son obligatorias y constituyen los **REQUERIMIENTOS MÍNIMOS PARA QUE SU PROYECTO SEA ACEPTADO Y ADMITIDO A SER CALIFICADO**. No cumplir con los requerimientos mínimos otorga al grupo una nota automática de cero (0). Esto aplica, incluso, si su proyecto aplica funcionalidad extra.

Tenga en cuenta, sin embargo, que cumplir los requerimientos significa apuntar a **la nota mínima 80%**.

Ventajosamente, este proyecto es también una oportunidad para ser creativo. Así, usted y su grupo pueden implementar funcionalidad extra, que contribuya a una mejor versión del juego. A continuación, se muestra una lista de algunas de funcionalidades opcionales que usted y sus compañeros de grupo podrían implementar para aplicar a puntos más allá de la nota mínima:

- Hacer recomendaciones al usuario humano sobre qué movimiento le conviene hacer.
- Mostrar los tableros intermedios generados cada vez que la computadora debe tomar una decisión junto con los valores de la función de utilidad.
- Mostrar el árbol general que se genera desde que el juego comienza hasta que éste termina.
- Permitir un modo de juego en el que la computadora juega contra sí misma.
- Permitir un modo de juego en el que dos humanos juegan uno contra otro.
- Guardar y reabrir partidas a medio jugar
- Etcétera

Usted y sus compañeros de grupo son libres de implementar cualquier otra funcionalidad extra que consideren pertinente para los objetivos del curso de Estructuras de Datos. Recuerde, esta es una oportunidad para ser creativos.

La reproducción de sonidos durante el juego **NO SERÁ CONSIDERADA COMO FUNCIONALIDAD** que merezca puntos extra. Esto no significa que su juego no puede incluir sonidos. Sin embargo, reproducir sonidos en Java es trivial y no aporta mucho al contenido del curso de Estructuras.

Entregables

El proyecto de NetBeans que implemente la interfaz gráfica final de su proyecto, con -al menos- las funcionalidades mínimas detalladas anteriormente, debe ser entregado hasta las 8:00 pm del día **martes 16 de enero** de 2023.

Su entrega deberá incluir además un reporte utilizando la misma plantilla del**.docx**. que fue enviada para la entrega del proyecto de la primera evaluación. Dicho reporte debe incluir:

- *Screenshots* de su interfaz explicando su funcionalidad,
- Una imagen de su clase principal, y
- Tabla de auto- y co-evaluación de los miembros del grupo

Su proyecto de Netbeans y el archivo **.docx** deben ser entregados a través del Aula Virtual en un único archivo comprimido **.zip**.

No se aceptarán entregas atrasadas, sin reporte, o que no implementen los requerimientos mínimos.

Recuerde que este **es un proyecto grupal**. Esto significa que usted debe trabajar **con** sus compañeros de grupo para sacar el proyecto adelante. Su grupo deberá reunirse periódicamente para coordinar y discutir acciones relacionadas al proyecto. Dividir funcionalidad y repartir responsabilidades podría no ser la mejor estrategia, sobre todo en las fases tempranas del proyecto.

Presentaciones

Usted y sus compañeros de grupo deberán presentar su proyecto (el código subido al Aula Virtual) al profesor en una reunión presencial que tendrá lugar el **jueves 18 de enero, en nuestro horario y lugar habitual de clases**. Esta reunión tendrá una duración de **10 MINUTOS** por grupo y se efectuará únicamente entre el profesor y **TODOS** los miembros del grupo. Allí, las observaciones sobre su entrega serán dadas de manera verbal.

Solo los grupos que hayan entregado un proyecto funcional deberán presentarse a esta reunión. Los grupos o estudiantes que no cumplan con este requisito DEBEN ABSTERSE DE ASISTIR ESE DÍA.

La reunión iniciará con uno de los miembros del grupo mostrando su proyecto en funcionamiento. Esta demostración debe evidenciar que el grupo ha cumplido con los requerimientos mínimos del proyecto y debe mencionar explícitamente toda funcionalidad extra que hayan implementado. **Su grupo debe definir con anticipación qué miembro del grupo estará a cargo de esta presentación inicial.**

La presentación inicial debe enfocarse en lo que su grupo ha implementado. **NO** debe ser sobre los requerimientos del proyecto --- el profesor tiene claro lo que se ha solicitado. Asimismo, el profesor conoce los nombres de los integrantes del grupo, por tanto, no es necesario presentar a sus compañeros. **Enfóquese en mostrar su proyecto en ejecución y en comunicar al profesor lo que su grupo ha logrado.**

Durante la reunión, cada integrante del grupo deberá responder, al menos, una pregunta sobre el proyecto. Para esto, todos los miembros del grupo deberán tener el código subido al Aula Virtual abierto en las computadoras del laboratorio donde se realizará la presentación. Sus respuestas serán calificadas y la calificación individual recibida será uno de los dos factores multiplicativos que contribuirán a su nota en el proyecto.

TODOS los archivos detallados arriba (.docx, .zip y su proyecto de NetBeans) deben ser nombrados con el número de su grupo precedido del sufijo "Grupo_". Por ejemplo: Grupo_03 o Grupo_12.

Recuerde lo que se explicó en la primera clase:

> Toda instrucción de este curso está diseñada para no hacer perder tiempo a nadie. Ni al profesor, ni al estudiante. Todo incumplimiento de una instrucción que incurra en una pérdida de tiempo para el profesor **SERÁ PENALIZADO CON PUNTOS** en la actividad correspondiente.

De nuevo: **NO** se aceptarán entregas atrasadas o incompletas.

Instrucciones Finales

Recuerde: **Este es un proyecto grupal.**

Sea oportuno(a) en cuanto a sus preguntas: **No se atenderán consultas sobre el proyecto después del 21 de diciembre de 2023**. Tampoco se responderán preguntas que consulten sobre instrucciones indicadas explícitamente en este documento. Antes de hacer preguntas, **LEA ESTE DOCUMENTO DETENIDAMENTE**.

Es **OBLIGATORIO** que considere las políticas de buena conducta académica que se explicaron en la primera clase del curso respecto al plagio académico y demás violaciones del código de Ética de la ESPOL. Los autores del proyecto deben ser usted y sus compañeros de grupo. Cualquier indicio de lo contrario, será reportado a las unidades correspondientes dentro de la universidad para el tratamiento pertinente.

Finalmente, **siga las instrucciones de este documento** para evitar inconvenientes. Si tiene dudas, consulte al profesor (en lugar de asumir cosas que pueden ser incorrectas). Los ayudantes de la materia podrían no ser la fuente de información más confiable cuando se trata de responder preguntas sobre el proyecto.

¡Muchos éxitos a todos!

RECUERDE

Un proyecto que no implementa la funcionalidad mínima solicitada en este documento es calificado automáticamente con **CERO** y **NO PUEDE SER PRESENTADO NI DEFENDIDO**. Esto aplica, incluso, si su proyecto implementa funcionalidad adicional.

Por tanto, si su proyecto no implementa los requerimientos mínimos descritos en este documento, **NO SE PRESENTE A LA DEFENSA DEL PROYECTO** esperando convencer al profesor que le califique un proyecto incompleto. Sea respetuoso(a) del tiempo del profesor y de sus compañeros.