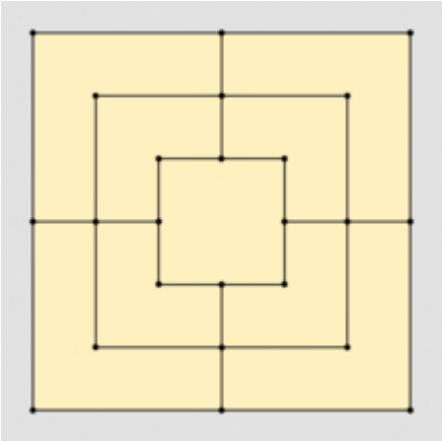
Departamento de Computación
FCEFQyN, Universidad Nacional de Río Cuarto
Asignaturas: Diseño de Algoritmos - Algoritmos II
Primer Cuatrimestre de 2020

Trabajo Práctico No. 2

La resolución de este trabajo práctico debe ser enviada a través de GitHub Classroom, antes de las **23:59 del lunes 15 de Junio de 2020**.
El código provisto como parte de la solución a los ejercicios deberá estar documentado apropiadamente (por ejemplo, con comentarios en el código). Aquellas soluciones que no requieran programación, como así también la documentación adicional de código que se desee proveer, debe entregarse en **archivos de texto convencionales**, en el mismo repositorio de entrega de las soluciones, con nombres que permitan identificar facilmente su contenido.
Tanto la calidad de la solución, como el código y su documentación serán considerados en la calificación. Recuerde además que los trabajos prácticos **tienen defensa**, y que el trabajo en la resolución de los ejercicios debe realizarse en grupos de 3 personas.

Ej. 1. El juego conocido como *Lasker Morris* es una variante del juego *Nine Men Morris*, inventada por el campeón mundial de ajedrez Emanuel Lasker. En esta variante de juego, dos jugadores alternan la ubicación de 10 piezas (blancas y negras) en los vértices de un tablero como el siguiente:



Las piezas blancas comienzan, y en cada turno tienen la oportunidad de agregar una nueva pieza, o mover una pieza existente en el tablero, a una de las posiciones libres adyacentes a la misma. Cuando un jugador puede conectar tres de sus piezas en línea, elige una de las piezas de su contrincante, para quitar del tablero. Finalmente, cuando a un jugador sólo le quedan 3 piezas, puede mover cualquiera de sus piezas a cualquier lugar libre del tablero.

En este caso se requiere implementar un programa que juegue a Lasker Morris, tanto con blancas como con negras, utilizando como motor de juego a *Minimax con poda alfa-beta.* La implementación debe respetar las interfaces y clases base provistas como parte del ejercicio, y debe garantizar que computa un movimiento correcto en a lo sumo 1 minuto (el límite de tiempo impuesto para que cada motor realice su movimiento). Si el motor produce un movimiento inválido o no es capaz de producir un movimiento dentro del tiempo establecido, se considerará que el motor *perdió* el juego.

Los motores producidos por los equipos jugarán un torneo en el que todos competirán contra todos, como locales (blancas) y visitantes (negras). Se asignarán los puntajes usuales por partido jugado: 3 puntos por partido ganado, 1 punto por partido empatado (si ninguno consigue ganar dentro de los 60 movimientos por jugador), 0 puntos por partido perdido. Adicionalmente, cuando un jugador pierda debido a movimiento inválido o imposibilidad de realizar un movimiento, se considerará una penalidad adicional, a ser definida.

Nota: Los primeros puestos del torneo recibirán puntajes adicionales.

Ej. 2. El *Gran DT* es una competencia en la que los participantes arman equipos utilizando jugadores del torneo de primera división del fútbol argentino. Luego, cada fecha, todos los jugadores del torneo reciben puntajes basados en sus respectivos desemeños en la fecha, y así los participantes cuyos equipos recibieron el mayor puntaje global (suma de los puntajes de los jugadores), reciben premios. Para armar los equipos, existen una cantidad de reglas, que involucran, entre otras cosas, un *presupuesto máximo* (cada jugador tiene un valor monetario, y la suma del equipo no puede superar el presupuesto), y una *formación* (de acuerdo a la formación elegida se puede tener un máximo de jugadores por cada puesto).

En este problema, se requiere desarrollar un programa Java que, dada una nómina de jugadores del torneo argentino de fútbol de primera división, con sus respectivos clubes, precio y valor estimado, y un presupuesto global, encuentre el equipo óptimo para una formación específica, que no supere el presupuesto. El presupuesto utilizado es el vigente para Gran DT, $65.000.000, y la formación será 4-3-3; esto último implica armar un equipo de 15 jugadores, conformado por: 2 arqueros, 5 defensores, 4 volantes y 4 delanteros (esto es dado que se incluye un suplente por posición de juego). Para resolver el problema, utilizaremos algoritmos genéticos y la biblioteca JGAP. El proyecto asociado a este ejercicio ya cuenta con clases base, información de las estadísticas requeridas de los jugadores[1], clases para procesar esta información, etc. Dado que se utilizan algoritmos genéticos, no hay garantía de construcción de soluciones óptimas. Sin embargo, se espera que los tests que acompañan el proyecto, que ejecutan el algoritmo múltiples veces para tener en cuenta la aleatoriedad del comportamiento de la aplicación, construyan equipos que satisfagan las restricciones del juego.

Nota: Las implementaciones que consigan equipos cercanos al óptimo, eficientemente, recibirán puntajes adicionales.

Ej. 3. Alloy es un lenguaje formal que puede utilizarse para capturar modelos de software con precisión, y realizar diferentes análisis sobre los mismos. En este problema, se requerirá capturar diferentes propiedades (15 propiedades en total) sobre un modelo de clases, tutores, estudiantes y grupos. Cada una de las 15 propiedades a capturar está especificada informalmente en castellano, en el archivo base (con definiciones de signaturas, predicados y otros elementos) que acompaña a este ejercicio. La resolución de este problema debe realizarse modificando sólo los lugares indicados para ser completados (el cuerpo de cada uno de los predicados que corresponde a las propiedades a capturar).

Ej. 4. Se cuenta con un conjunto de equipos $e_1, \ldots, e_k$ de fútbol, con k un número par, y se desea producir un *fixture*, compuesto por $k-1$ fechas, de manera tal que todos los equipos jueguen contra todos exactamente una vez. Cada equipo juega con cada uno de los otros equipos exactamente una vez en el torneo. Cada equipo juega $k/2$ partidos de local y $k/2-1$ partidos de visitante, o viceversa, y ningún equipo debe jugar más de dos partidos consecutivos de local o visitante (sólo una vez en todo el torneo).

Modele el problema en Alloy, de manera tal de aprovechar Alloy Analyzer para construir fixtures automáticamente. Puede construir soluciones para $k = 24$?

[1] Tomadas de `https://www.planetagrandt.com.ar`