# TRABAJO PRACTICO Nº3

Asignación de arbitro con algoritmo goloso.

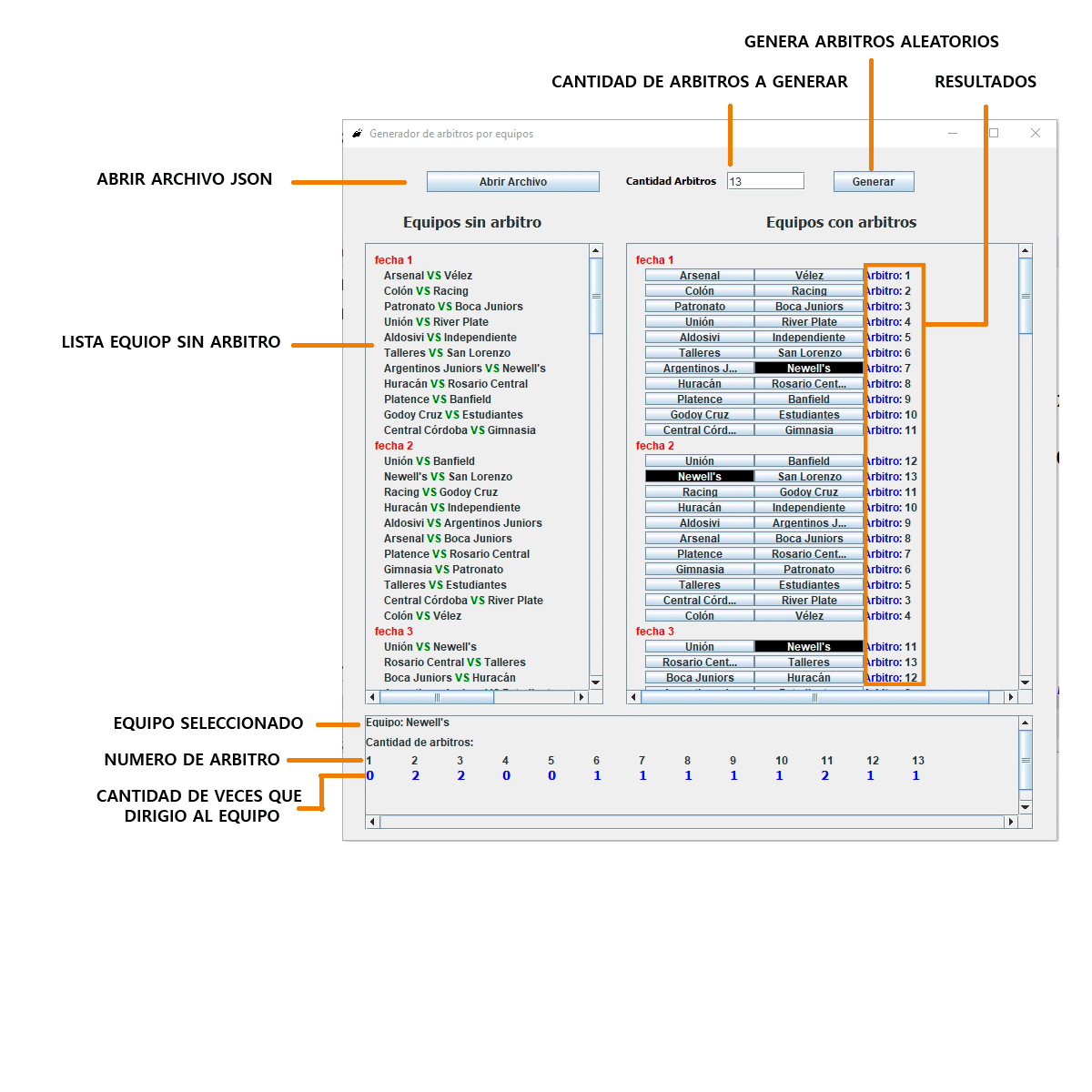
Nombre y Apellido: Javier Hernán Monzon.

Comisión: 01.

Turno: Noche.

Asignación de arbitro con algoritmo goloso.

Interfaz grafica



Resumen de decisiones.

Asignar == false

generar

Crear fechas

Asignar == false

Asignar arbitro, aun no repetido en encuentro

Asignar arbitro menos asignado, entre los equipos

Asignar arbitro consecutivo

Asignar == false

Asignar arbitro menos repetido hasta la fecha

Especificación de clases y métodos

Clase MainUI

Métodos para la creación de la interfaz gráfica de nuestro programa, en donde mostraremos los resultados de la generación de nuestro logaritmo goloso.

Métodos:

InicializarLogicaTP : Crea el archivo. json por defecto, luego crea la clase principal de nuestro algoritmo goloso. Luego procede a cargar los datos de nuestro archivo y la muestra en la interfaz.

cargarDatos: Carga el archivo .json, y coloca los datos o las fechas, en nuestro algoritmo goloso para ser procesado.

crearArchivo: Crea un archivo con con la extensión .json y sus valores.

abrirArchivo: Abre un archivo .json, que corresponda al formato establecido por defecto.

mostrarResultadosAlgoritmoGoloso: Muestra los resultados de nuestro algoritmo goloso en la interfaz, con la asignación de árbitros aleatoria.

colocarfechasDeArchivoUI: Coloca los datos del archivo cargado en la interfaz, en el cuadro izquierdo. Recibe la fecha y la lista de partidos para esa fecha. Coloca la fecha y los partidos en cada llamada.

colocarfechasAlgoritmoGoloso: Coloca los datos procesados en la interfaz, en el cuadro derecho. Recibe la fecha y la lista de partidos para esa fecha, pero con los árbitros ya asignados. Coloca la fecha y los partidos en cada llamada, con sus respectivos árbitros.

mostrarEstadisticas: Coloca las estadísticas individuales de cada equipo al seleccionar un equipo del cuadro derecho con sus árbitros procesados, en el cuadro inferior veremos los árbitros y la cantidad de veces que fueron asignados esos árbitros para ese equipo.

resaltarEquipo: Selecciona de la lista de árbitros procesados, la selección de nuestro equipo, y resaltara al equipo seleccionado en cada fecha.

switchGenerarBtn: Habilita y deshabilita nuestro botón de generar algoritmo goloso, cuando lo está procesando lo deshabilita y cuando termino de procesar lo deshabilita.

resetUIAlgoritmoGoloso, resetUIFechasCargadas, resetUIEstadisticas: Métodos que limpian la interfaz, si generamos nuevo número de árbitros, también limpia las estadísticas cuando seleccionamos otro equipo para dibujar las del equipo seleccionado.

keyReleased : Lee los datos colocados en la caja de texto para ingresar el número de árbitros, verificara que la cantidad sea mayor a cero, con una expresión regular.

Clase AlgoritmoGoloso

Clase que hará los cálculos para asignar arbitro a los partidos.

Métodos:

generarAlgoritmoGoloso: Método que devolverá las fechas con los árbitros asignados, recibe las fechas sin árbitros, y la cantidad de árbitros para procesar en cada fecha.

asignarArbitro: Asigna un árbitro a un partido, recibe la fecha actual, el partido a asignar un árbitro, y la cantidad de árbitros, y las fechas para comparar árbitros anteriores.

obtenerPartidosEquipo: Obtiene todos los partidos del equipo en las fechas, ya sea de local o visitante.

getCantidadVecesArbitroEnEquipo: Obtiene la cantidad de veces en que un árbitro fue asignado en cada partido, devuelve un HashMap con el árbitro correspondiente en su key, y como valor la cantidad de árbitros.

obtenerArbitroNoAsignado: Devuelve un árbitro que no se ha sido seleccionado en algunos de los equipos y que no se ha asignado hasta la fecha actual. Recibe un ArrayList con los árbitros que ya han sido asignado en la fecha, también recibe la cantidad de árbitros. Y por otro lado también recibe dos HashMap, uno para cada equipo que corresponde a los árbitros que ya han dirigido hasta el momento, con la información de la cantidad de árbitros y el árbitro que dirigió al equipo en las fechas.

getArbitroMenosAsignado: Hace el cálculo del árbitro que menos se ha seleccionado en ambos equipos, y devuelve ese árbitro, en caso de que la cantidad sea igual devolverá cero, y será el método menorCantidadDeArbitroRepetido, que tratara este caso.

.

**int** calcularSiguienteArbitro: Buscamos el árbitro que no se haya colocado en ambos equipos, en caso contrario buscamos el menos repetido. Recibe Dos HashMap con la información de cada equipo de sus árbitros.

.

obtenerArbitrosDeFecha: Devuelve los árbitros que ya han sido seleccionado para una fecha.

**int** menorCantidadDeArbitroRepetido: Buscamos el árbitro que menos se haya repetido en todas las fechas, recibe las fechas. Recibe como parámetro las fechas, y la cantidad de árbitros.

obtenerArbitroConsecutivo: Devuelve los árbitros en forma consecutiva, en los primeros partidos, cuando se terminen de asignar todos los árbitros devolverá cero.

resetArbitros**:** Resetea los árbitros para la fecha cuando volvemos a generar nuevos árbitros.

Clase FixtureGenerador

Crea una lista de equipos de forma aleatoria, con sus fechas.

**enum** Equipos: Lista de quipos que agregaremos, siempre debe ser pares.

crearListaPorDefectoEquipos**:** Crea la lista de equipos de defecto por medio del enum de nombre Equipos.

generarFechas**:** Genera las fechas, siempre que los equipos agregados sean pares.

agregarEquipo**:** Agrega un equipo, siempre que no exista en los equipos agregados y sea un nombre con longitud mayor a tres caracteres.

armarEquipo**:** Arma los equipos en forma aleatoria, siempre removiendo el que se seleccionó y eligiendo nuevamente su rival.

Clase ArchivoJSON

Clase que creara nuestro archivo .json, también leerá un archivo .json que cumpla con el formato, de {"datoJson": {"Numero Fecha": [{"local": "Banfield","visitante": "Argentinos Juniors","arbitro": -1},…], “Numero Fecha”…

Clase Partido

Clase que representa nuestro partido, con el equipo local y visitante. Contiene getters y setters.