Problemáticas

Cuatro en Línea

- Se requería el uso de una Clase Tablero que pudiera almacenar la información correspondiente al tablero mismo, además de proveer las funciones necesarias para su impresión y para que se pudieran hacer las jugadas por parte del jugador o la IA
- Se requería una IA que pudiese hacer búsqueda adversaria para jugar contra el usuario. Se utilizó una técnica de NegaMax con una búsqueda limitada del tablero, además de una heurística para que la IA pudiese estimar la mejor jugada sin necesidad de llegar hasta el final del árbol.
- Para poder hacer la búsqueda de NegaMax, se necesitaba poder duplicar la información del tablero, para ello se implementó el método *clone()* el cuál duplicó columna por columna el arreglo, ya que de otra forma solamente pasaba la referencia y quedaban todos los tableros exactamente iguales, algo que no era un comportamiento deseado cuando se quieren explorar distintas posibilidades.

Emoticons

- Se requería leer archivos, para esto se utilizó la librería *File* de Java que viene implementada en Kotlin, así como el almacenamiento de los 0's y 1's como un arreglo bi-dimensional de booleanos.
- Para poder determinar cuál es el emoticon que se quiere leer, se implementó una comparación dado un set de referencia, se revisa a cuál de los 7 es más parecido y se asume que ese es el que escribieron en la matriz
- La fiabilidad de esta solución depende del set de referencia que se use, en el caso actual, el set tiene un margen de error del 25%. Algo que si bien es alto, fue a la solución que conseguí llegar dentro del tiempo establecido.