# PRACICA 2

Programació 2

Ton Llucià Senserrich

47125160T

Pralab2

# 1 Desenvolupament de la practica

He afrontat un problema el qual m'ha portat bastant temps resoldre i és que a l'haver acabat d'escriure tot el codi i afrontar els últims detalls em fallaven els tests que detecten si guanyes quan deixes a l'oponent sense moviments, i entre anar provant si les funcions que utilitza estaven bé i comprovar que ho havia pensat de la manera correcta, al final però em vaig adonar que en mirar si l'oponent tenia moviments possibles utilitzava la funció isValidFrom cosa que no em servia, perquè aquesta depèn del jugador actual i no era una opció repetir-la exactament igual, ja que en aquesta es fa servir la funció isValidTo la qual també depèn del jugador actual i copiar-les totes era molta duplicitat de codi per tant vaig crear la funció canMove la qual és com un isValidFrom però te en compte el color de la posició en el tauler en lloc de el jugador actual.

## 1.1 Move

En aquest apartat es requereix una funció que efectuí un moviment i canviï el jugador actual si fa falta i retorni l'objecte Move corresponent.

M'ha semblat complexa, ja que s'ha de tenir en compte molts escenaris i és una funció que desenvolupa diverses accions, però per tal de resoldre-ho el que jo he fet ha estat Primerament mirat si és una captura i si és així canviar la posició del mig a empty, després he canviat el validTo i després el validFrom i per últim he utilitzat les funcions auxiliars whiteHasWon i blackHasWon per tal de determinar si el jugador actual havia guanyat la partida i en aquest cas no canviar de torn i retorna l'objecte Move.

## 1.2 isValidTo

En aquest cas es demana una funció que a partir de dos objectes Position determini si és un moviment vàlid depenent del jugador actual i la situació actual del tauler, aquesta funció m'ha semblat complicada perquè en pensar-la per primer cop em vaig fer un embolic i no acabava d'entendre quines comprovacions havia de fer, després de pensar-ho vaig adonar-me que no era tan complexa, i vaig utilitzar la següent estratègia.
Primer calculo les variables utilitzades per poder fer les comprovacions necessàries, seguidament comprovo certes coses utilitzant triples condicionals i retorno el resultat obtingut.

## 1.3 sameDiagonalAs

En aquest apartat es demana una funció que a partir de dos objectes Position comprovi si pertanyen a la mateixa diagonal, aquesta funció no és realment complexa, però a mi em va costar, ja que em vaig fer un embolic, ja que pensava que havia de fer un bucle recorrent les diagonals, però finalment, després de donar-hi unes quantes voltes vaig adonar-me que si li restes la una x a l'altre i una y a l'altre t'hauria de quedar el mateix i obtens el valor absolut hauria de quedar el mateix nombre en cas que estiguessin a la mateixa diagonal, per tant aquesta solució és bastant optima i fàcil.

# 2 Conclusions

Jo he après a utilitzar en més profunditat l'entorn d'IntelliJ, ja que no havia practicat gaire amb ell, i també a optimitzar més les funcions ja creades, he après poc, ja que jo ja havia treballat amb java anteriorment.
Si tornés a començar la practica el que faria és optimitzar les funcions des de bon principi, ja que al final m'he quedat sense temps per optimitzar-ho tot al maxim.