Falten noms i/o número d'equip

Nota: 9,5

Molt bon disseny

Nota: Els constructors per paràmetres de les classes només tenen una intenció organitzativa de cara a plantejar els mètodes. No sabem a hores d'ara qui serà el responsable de guardar segons què o com. El primer pròposit és cobrir les necessitats del programa de la manera més intuïtiva.

Tipus Tauler

Descripció general: Tauler d'escacs format per n files i m columnes, i sobre el qual es mouen les peces del joc mentre es disputa la partida.

Operacions:

Millor List<Enroc> : usar interface List, o fins i tot Collection si no importa l'ordre

```
Tauler(int n, int m, ArrayList<Enroc> enrocsDisponibles)
Pre:  n>0 /\ m>0.
Post: Crea un tauler amb els paràmetres d'entrada.

void  posicionarPeca(ArrayList<Peca>  llPecesB,  ArrayList<Peca>
llPecesN)
Pre:  ---
Post: Es col·loquen totes les peces de llPecesB i llPecesN sobre el
      tauler, a la posició que els pertoca.
```

Ull, s'hauria de precisar més

```
bool comprovarJugadaOrdinaria(JugadaOrdinaria jugada)
Pre:  ---
Post: Retorna cert si l'estat del tauler permet realitzar la jugada
      demanada.

bool efectuarJugadaOrdinaria(JugadaOrdinaria jugada)
Pre:  ---
Post: S'ha realitzat la jugada, sempre i quan l'estat del tauler i les
      característiques de la peça ho permetin, modificant la peça del
      tauler involucrada en la jugada.  Retorna false si durant la
      jugada no s'ha capturat cap peça, cert altrament.

bool comprovarJugadaEnroc(JugadaEnroc jugada)
Pre:  ---
Post: Retorna cert si es pot realitzar l'enroc de la jugada d'entrada,
      fals altrament.

Void efectuarJugadaEnroc(JugadaEnroc jugada)
Pre: ---
Post: S'ha realitzat l'enroc, sempre i quan aquest existeixi i
      l'estat del tauler ho permeti. El tauler ha quedat modificat
      consegüentment.
```

```
void desferJugada(Jugada jugada)
Pre: ---
Post: L'estat del tauler torna a ser el mateix que abans de realitzar
     la jugada, sempre que sigui aplicable desfer la jugada.
```

No s'ha de desfer simplement l'última?

Falta operació refer?

```
bool estaReiOfegat(char jugador)
Pre:  jugador és b o n.
Post: Retorna cert si es produeix la situació en què no es pot
      realitzar cap jugada per jugador i el rei de jugador no es troba
      en estat d'escac, fals altrament.

int escac(char jugador)
Pre:  jugador és b o n.
Post: Retorna 1 si el rei del jugador està sota amenaça immediata per
      alguna peça del jugador contrari(escac), retorna 2 si el rei de
      jugador es troba sota escac i mat i 0 si no es troba en cap
      de les altres situacions.

Peca retornaPeca(Posicio pos)
Pre:  pos és una posició existent a tauler
Post: Retorna la peça que hi ha en la posició indicada pel paràmetre
      d'entrada. Si no hi ha peça retorn un null.
```

pos

Tipus Peca

Descripció general: Peça d'escacs que es mou sobre un tauler efectuant moviments permesos.

Operacions:

```
Peca(string nom, char simbol, string imgB, string imgN, int valor,
ArrayList<Moviment> mov, ArrayList<Moviment> movIni, bool promo, bool
invulnerable, bool moguda, bool morta)
Pre:  A <= simbol <= Z, imgB i imgN són rutes a imatges correctes,
      valor > 0.
Post: Es crea una peça amb els paràmetres entrats.

Moviment potArribar(Posicio inicial, Posicio final)
Pre:  ---
Post: Retorna el moviment que li permet anar d'inicial a final. En
      cas de no poder anar-hi, retorna NULL.

void afegirPosicioInicial(Posicio inicial)
Pre:  ---
```

Descriure què representen

interessant...

Per què és important l'inicial? No podria ser qualdevol posició?

Post: Guarda a peça la seva posició inicial dins del tauler.

bool esInvulnerable()
Pre: ---
Post: Retorna cert si la peça no pot ser capturada, fals altrament.

bool esMoguda()
Pre: ---
Post: Retorna cert si la peça ha estat moguda de la seva posició inicial, fals altrament.

String nomPeca()
Pre: ---
Post: Retorna el nom de la peça.

char caracterPeca()
Pre: ---
Post: Retorna el caràcter representatiu de la peça.

Tipus Moviment

Descripció general: Moviment sobre el pla que pot efectuar una peça, entenent com aquest la direcció, sentit i increment aplicada a una posició. A més inclou la possibilitat de saltar i capturar.

Operacions:

Moviment(char vertical, char horitzontal, int capturar, int saltar)
Pre: ---
Post: Es crea un objecte Moviment amb els paràmetres indicats.

Descriure el significat dels paràmetres del constructor a l'estil

int movCaptura()
Pre: ---
Post: Retorna 0 si no pot capturar una peça a destí, 1 si pot i 2 si el moviment només es pot realitza capturant peça a destí.

int movSalta()
Pre: ---
Post: Retorna 0 si el moviment no permet saltar, 1 si sí ho permet i 2 si a més de saltar captura a la vegada.

Tipus Posicio

Descripció general: Una casella del taules d'escacs.

Operacions:

Posicio(int fila, char columna)
Pre: fila > 0.
Post: Es crea una Posició amb els paràmetres indicats.

int fila
Pre: ---
Post: Retorna el valor de fila

char columna
Pre: ---
Post: Retorna el valor de la columna

Tipus Jugada

Descripció general: Jugada d'escacs realitzada per un jugador durant el seu torn.

Operacions:

Podríeu guardar el nom a la classe Jugador, i aquí passar el Jugador

Jugada(Posicio pospri, Posicio posseg, String jugador)
Pre: ---
Post: Crea una jugada amb els paràmetres d'entrada.

String jugador()
Pre: ---
Post: Retorna un string amb qui ha sol·licitat aquesta jugada.

void resultatJugada(String res)
Pre: ---
Post: S'ha assignat a la jugada el seu efecte a la partida.

?

Tipus JugadaEnroc refina Jugada

Descripció general: Jugada d'escacs que representa un moviment especial de fortificació. Consisteix en moure dues peces a la vegada, una es mou a la posició del mig entre ambdues i l'altra al costat de la primera després d'haver-se creuat.

Tipus JugadaOrdinaria refina Jugada

Descripció general: Típica jugada d'escacs que es basa en moure una sola peça a una nova posició.

Tipus JugadaInterrompuda refina Jugada

Descripció general: Jugada que altera el curs de la partida, sigui perquè el jugadora ha sol·licitat taules, ha acceptat taules, ha decidit rendir-se o ajornar la partida.

Tipus Jugador

Descripció general: Jugador d'escacs amb llibertat d'elecció.

Operacions:

Jugador(char color)
Pre: color == 'b' o color == 'n'
Post: Es crea un jugador del color indicat amb el paràmetre color

Jugada demanarJugada()
Pre: ---
Post: Es retorna un objecte Jugada del tipus que toca amb les dades entrades per teclat.

Tipus JugadorCPU

Descripció general: Jugador d'escacs controlat per la CPU

Operacions:

Jugador(char color)
Pre: color == 'b' o color == 'n'.
Post: Es crea un jugador del color indicat amb el paràmetre color.

Jugada demanarJugada(Tauler tauler)
Pre: ---
Post: Es retorna una jugada generada a partir d'un esquema algorítmic.

Tipus Partida

Descripció general: Partida d'escacs, encarregada de gestionar el joc I de moure les peces del tauler.

Operacions:

Partida(Tauler t, Jugador jugador1, Jugador jugador2, int limitEscacs, int limitInaccio)
Pre: ---

Descriure mínimament els paràmetres no evidents

Post: Crea una partida a partir dels paràmetres d'entrada.

void repartirPeces(HashMap<String,peça> pecesDisponibles,
ArrayList<String> ordrePeces)
Pre: ---
Post: Es reparteixen, segons l'ordre marcat per ordrePeces, totes les peces d'escacs de pecesDisponibles sobre el tauler on es disputa la partida.

Hauríeu d'intentar agrupar les operacions d'interacció amb l'usuari en un sol mòdul. Mireu l'exemple de les Dames al Moodle (en mode text).

Jugada llegirJugada()
Pre: ---
Post: Retorna la jugada sol·licitada pel jugador que té el torn en aquell moment.

void actualitzarTorn()
Pre: ---
Post: Es canvia el torn per a que a la propera jugada pertanyi al jugador contrari.

void gestionarJugada(Jugada jugada)
Pre: ---
Post: Si la jugada és de tipus ordinaria o enroc s'ha procedit a realitzar-la, sempre que la jugada hagi estat vàlida i la situació de la partida ho hagi permés. També s'ha anotat el resultat, si és destacable, a jugada. En cas que hagi finalitzat en "escac i mat" es dona per finalitzada la partida i es guarda el desenvolupament. Si és una jugada interrompuda es determina què fer segons la causa: continuar amb la partida o finalitzar-la.

void guardarJugada(Jugada jugada)
Pre: ---
Post: jugada ha estat afegida a l'historial de jugades.

Void desferJugada()
Pre: ---
Post: L'estat de la partida és l'anterior a la darrera jugada archivada.

Falta referJugada()

Tipus Enroc

Descripció general: Característiques d'un moviment especial entre dues peces. També es contemplen requeriments previs, com que mai hagin estat mogudes i/o que hi hagi espais en buit entre elles per tal

d'efectuar-se l'enroc.

Operacions:

Per què no Peça?

```
Enroc(String pecaA, String pecaB, bool quiets, bool buit)
Pre:  ---
Post: S'ha creat un objecte tipus Enroc amb els paràmetres d'entrada.

String peca(char opcio)
Pre:  opcio == "A" o opcio == "B"
Post: Retorna el nom de la peça que indica opcio.

bool quietes()
Pre: ---
Post: Retorna cert si les peces no s'han pogut moure previament per
      fer l'enroc actual, fals altrament.

bool buit()
Pre: ---
Post: Retorna cert si no pot haver-hi peces entre ambdues per fer
      l'enroc, fals altrament.
```

Modul funcional ES_JSON

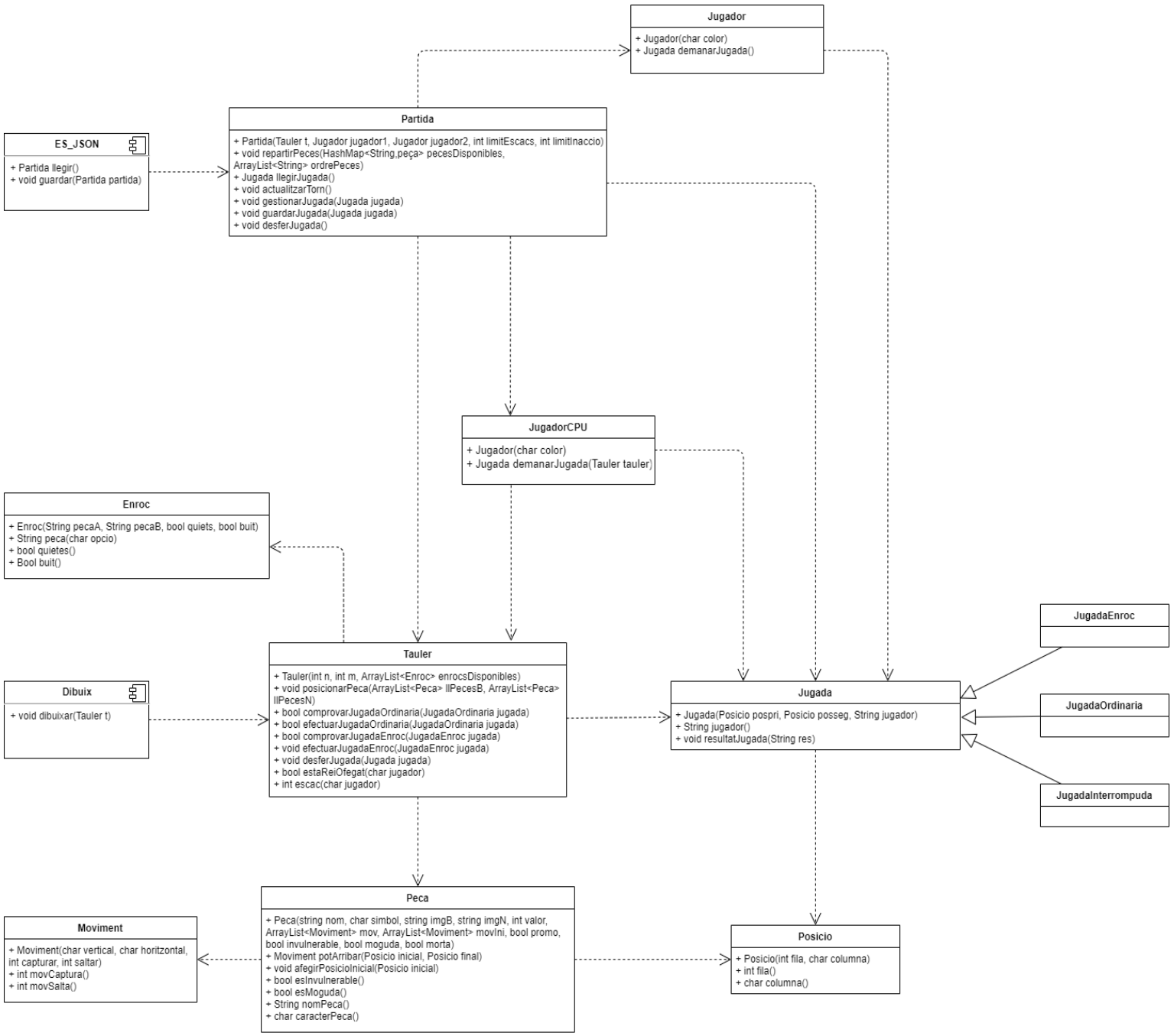
```
Partida llegir()
Pre:  ---
Post: Llegeix el fitxer de regles del joc i la partida guardada per
      retornar un objecte Partida amb la partida iniciada i preparada
      per jugar.

void guardar(Partida partida)
Pre:  ---
Post: S'ha generat un fitxer que descriu el desenvolupament de la
      partida.
```

Modul funcional Dibuix

```
void dibuixar(Tauler t)
Pre: ---
Post: Dibuixa el tauler t en una finestra.
```

Ho teniu molt ben encarrilat com a punt de partida. Evidentment, s'entén que poden haver-hi canvis.


Jugador
+ Jugador(char color)
+ Jugada demanarJugada()
Partida
+ Partida(Tauler t, Jugador jugador1, Jugador jugador2, int limitEscacs, int limitInaccio)
+ void repartirPeces(HashMap<String,peça> pecesDisponibles,
ArrayList<String> ordrePeces)
+ Jugada llegirJugada()
+ void actualitzarTorn()
+ void gestionarJugada(Jugada jugada)
+ void guardarJugada(Jugada jugada)
+ void desferJugada()
ES_JSON
+ Partida llegir()
+ void guardar(Partida partida)
JugadorCPU
+ Jugador(char color)
+ Jugada demanarJugada(Tauler tauler)
Enroc
+ Enroc(String pecaA, String pecaB, bool quiets, bool buit)
+ String peca(char opcio)
+ bool quietes()
+ Bool buit()
Tauler
+ Tauler(int n, int m, ArrayList<Enroc> enrocsDisponibles)
+ void posicionarPeca(ArrayList<Peca> llPecesB, ArrayList<Peca>
llPecesN)
+ bool comprovarJugadaOrdinaria(JugadaOrdinaria jugada)
+ bool efectuarJugadaOrdinaria(JugadaOrdinaria jugada)
+ bool comprovarJugadaEnroc(JugadaEnroc jugada)
+ void efectuarJugadaEnroc(JugadaEnroc jugada)
+ void desferJugada(Jugada jugada)
+ bool estaReiOfegat(char jugador)
+ int escac(char jugador)
Dibuix
+ void dibuixar(Tauler t)
Jugada
+ Jugada(Posicio pospri, Posicio posseg, String jugador)
+ String jugador()
+ void resultatJugada(String res)
JugadaEnroc
JugadaOrdinaria
JugadaInterrompuda
Peca
+ Peca(string nom, char simbol, string imgB, string imgN, int valor,
ArrayList<Moviment> mov, ArrayList<Moviment> movIni, bool promo,
bool invulnerable, bool moguda, bool morta)
+ Moviment potArribar(Posicio inicial, Posicio final)
+ void afegirPosicioInicial(Posicio inicial)
+ bool esInvulnerable()
+ bool esMoguda()
+ String nomPeca()
+ char caracterPeca()
Moviment
+ Moviment(char vertical, char horitzontal,
int capturar, int saltar)
+ int movCaptura()
+ int movSalta()
Posicio
+ Posicio(int fila, char columna)
+ int fila()
+ char columna()