Tipus JocsEscacs (main file)

Descripció general: És l'encarregat de inicialitzar, gestionar els arguments en els que hi haurà la mida del tauler i es podrà triar entre mode gràfic i mode text.

Operacions:

//pre: \p args és per veure si s'executa l'aplicació en mode text o en mode gràfic
//post: S'executa un joc d'Escacs; amb el paràmetre -g s'executa en mode gràfic
void main();

Tipus PartidaText

Descripció general: Administrador de la interfície del joc en mode text

Operacions:

public static void iniciarAplicacio() throws Exception
Pre: --
Post: Iniciador del joc, introdueixes les dades bàsiques i aquest crida els mètodes necessaris.

public static int demanarNjugadors()
Pre: --
Post: retorna el nombre de jugadors que entra l'usuari.

public static boolean esEnter(String s)
Pre: --
Post: True si s és un enter, false altrament.

private static void jugar() throws Exception
Pre: --
Post: S'ha jugat una partida

private static boolean tirada(StringBuilder res, StringBuilder posInici, StringBuilder posFinal, StringBuilder colorTorn) throws Exception
Pre: --
Post: retorna true si s'ha realitzat la tirada, false altrament

private static String promocio(StringBuilder posInici, StringBuilder posFinal)
Pre: --
Post: retorna un missatge donant informació del resultat de la promoció

private static void menuPromocio()
Pre: --
Post: S'ha triat una Peça per a convertir la Peça promocionada.

private static boolean taules()
Pre: --

Post: S'ha guardat una partida que ha acabat amb taules

private static boolean processarRes(StringBuilder res, StringBuilder colorTorn) throws Exception
Pre: --
Post: retorna true si s'ha realitzat la tirada, false altrament.

private static String llegirPosicioInici() throws Exception
Pre: --
Post: Retorna un avís amb el resultat de la posició inicial que ha seleccionat l'usuari

private static String llegirPosicioDesti() throws Exception
Pre: --
Post: Retorna un avís amb el resultat de la posició destí que ha seleccionat l'usuari

private static boolean posicioCorrecteOrigen(String s) throws Exception
Pre: --
Post: retorna si la posició origen és vàlida, false altrament.

private static String posicioCorrecteDesti(String s) throws Exception
Pre: --
Post: retorna si la posició destí és vàlida, false altrament.

Tipus PartidaGrafica

Descripció general: Administrador de la interfície del joc en mode gràfic

Operacions:

//pre: --
//post: s'executa una aplicació per poder jugar a escacs
public static void main(){

//pre: --
//post: Escena principal on l'usuari pot triar si començar una partida nova o carregar-ne una de començada o sortir.
private void crearEscenaPrincipal(){

//pre: \p opcio si igual a 1 indica que es comença una partida, si val 2 es carrega una partida
//post: Es crea una escena secundaria on l'usuari haurà d'entrar el nom del fitxer que necessita i \p opcio igual a 1, el nombre de jugadors, juntament amb un botó per confirmar el que ha entrat i un botó per anar a l'escena anterior en cas que s'hagi equivocat.
private void escenaSecundaria(int opcio){

//pre: \p opcio si igual a 1 indica que es comença una partida, si val 2 es carrega una partida

//post: return Un node on hi haurà un TextField perquè l'usuari hi entri el nom del fitxer, si \p opcio igual a 1, també hi haurà un CheckBox per triar el nombre de jugadors i per últim un botó per confirmar les dades que ha entrat
private Node prepararCentre(int opcio) {

//pre: \p s és el text que es mostrarà com a títol, \p d és una variable perquè el label quedi centrat.
//post: return un node on hi haurà el títol que es posarà a l'escena secundaria.
private static Node topLabel(String s, ReadOnlyDoubleProperty d){

//pre: \p s és el text que hi haurà dins el botó
//post: return un node que serà contindrà un botó que tindrà la funcionalitat de tirar enrere i anar a l'escena principal
private static Node botoInferior(String s){

//pre: --
//post: Escena de la partida creada on hi haurà el taulell i a la dreta una part amb diferents opcions per fer durant la partida
private static void crearEscenaPartida(){

//pre: \p p és el panell de l'escena de la partida, necessari per funcionalitats on es requereix canviar l'escena
//post: return un node que conté un BorderPane on al top hi ha un label per els avisos útils per l'usuari, al centre hi ha les opcions que són diferents botons i al bottom hi ha un botó per sortir de l'aplicació
private static Node crearOpcions(BorderPane p){

//pre: \p p és el panell de l'escena de la partida, necessari per funcionalitats on es requereix canviar l'escena
//post: return un node que conté un text on indica a l'usuari quin dels dos ha demanat les taules i dos botons, un per acceptar les taules i l'altre per declinar-les
private static Node crearPaneTaules(BorderPane p){

//pre: \p p és el panell de l'escena de la partida, necessari per funcionalitats on es requereix canviar l'escena
//post: return un node que conté un Pane on hi ha el taulell muntat i les peces posades a la seva posició corresponent
private static Node crearContingutPartida(BorderPane p){

//pre: \p f és la peça lògica que es vol representar gràficament, \p i i \p j són la columna i fila, respectivament on s'ha de posicionar la peça i \p p és el panell de l'escena de la partida, necessari per funcionalitats on es requereix canviar l'escena

//post: return un node que conté un BorderPane on al top hi ha un label per els avisos útils per l'usuari, al centre hi hales opcions que són diferents botons i al bottom hi ha un botó per sortir de l'aplicació
private static PecaGrafica crearFitxa(Peca f, int i, int j,BorderPane pane){

//pre: \p mov és una string on hi ha el moviment que es vol fer, \p newX i newY són les noves posicions gràfiques on anirà la peça gràfica en cas de que la tirada s'hagi realitzat correctament, \p p és la peça gràfica amb la que es vol realitzar la tirada, \p p és el panell de l'escena de la partida, necessari per funcionalitats on es requereix canviar l'escena i \p oldX és la columna on estava la peça gràfica, necessari en el cas de que es fagi enroc per trobar on moure les peces de l'enroc.
//post: S'ha intentat fer una tirada lògica i es processa el resultat que retorna:
-si retorna tiradaV es mou \p p a \p newX i \p newY i es canvia de torn
-si retorna tiradaMort s'eliminen les peces mortes i es mou \p p a \p newX i \p newY
-si retorna noTirada tornem \p p a la posició inicial ja que no s'ha pogur realitzar la tirada
-si retorna promocio es mou \p p a \p newX \p newY i es demana per quina peça es vol promocionar
-si retorna enrocFet es busca l'altre peça participant a l'enroc i es mouen a les posicions que toquen i es passa de torn
-si retorna escac si fa falta eliminem peçes que hagin mort en aquesta tirada, movem \p p a \p newX i \p newY i passem de torn
-si retorna EscacSeguits s'acaba la partida per taules per masses torns seguits amb escac al rei d'un equip
-si retorna TornsInacció s'acaba la partida per taules per masses torns seguits sense cap peça morta
-si retorna noTiradaEscac tornem \p p a la posició inicial perquè no s'ha pogut fer la tirada
si retorna escacmat s'acaba la partida per escac i mat i es canvia per una escena final
si retorna enrocNo tornem \p p a la posició inicial perquè no s'ha pogut realitzar l'enroc demanat.
private static void realitzarTirada(String mov, int newX, int newY, PecaGrafica p, BorderPane pane, int oldX){

//pre --
//post Es crea l'escena amb un label indicant el jugador que ha guanyat, i un gif. private static void crearEscenaFinal(){

//pre: \p pane és el panell de l'escena de la partida, necessari per funcionalitats on es requereix canviar l'escena
//post: Es demanen les peces eliminades en la tirada que s'ha realitzat i si n'hi ha, s'eliminen del taulell gràfic
private static void eliminarPeces(BorderPane pane) {

//pre: \p p és el panell de l'escena de la partida, necessari per funcionalitats on es requereix canviar l'escena, \p mov és el moviment que s'ha fet, necessàri per cridar el fer promoció lògic, \p x i \p y són les coordenades de la posició destí desgloçades
//post: Es crea un panell on hi ha un label preguntant per quina peça es vol canviar i un botó per cada peça per la que es pugui fer la promoció
private static void crearEscenaPromoció(BorderPane p, String mov, int x, int y){

//pre \p s és l'avís que volem que es mostri per l'usuari i \p p és el panell de l'escena de la partida, necessari per funcionalitats on es requereix canviar l'escena
//post Label d'avisos modificat amb el nou avís siguent \s
private static void modificarLblAvisos(String s, BorderPane p){

//pre: \p p és el panell de l'escena de la partida, necessari per trobar el panell d'opcions i poder indicar el canvi de torn graficament
//post: return un node que conté un BorderPane on al top hi ha un label per els avisos útils per l'usuari, al centre hi ha les opcions que són diferents botons i al bottom hi ha un botó per sortir de l'aplicació
private static void passarTorn(BorderPane p){

//pre: \p x i \p y són les coordenades de la posició de la peça que s'ha d'eliminar
//post: La peça que es trobava a la posició amb coordenades \p x i \p y s'ha eliminat del group _fitxes i del taulell gràfic
private static void eliminarPeca(int x, int y){

//pre: \p p són pixels
//post: return un enter que és la posició del taulell gràfic dels píxels \p p
private static int posTauler(double p){

Tipus Partida

Descripció general: És l'encarregat de gestionar tota la partida

Operacions:

//pre: fitxerPartida existeix i esta en un format correcte
//post: totes les variables per a continuar una partida estan settejades.
public Partida(String fitxerPartida) throws Exception {

//pre: fitxer de carrega existent
//post: Tirades carregades
public void carregarPartida(){

//pre: fitxerRegles existeix i esta en un format correcte
//post: totes les variables per a començar una partida estan settejades.
public Partida(String fitxerRegles, int jugadors) throws Exception {

```
//pre: --
//post: TornCanviat
public String canviarTorn() {

//pre: --
//post: Avisem del valor de la posició
public String posCorrecteOrigen (String posicioIndefinida) {

//pre: --
//post: Avisem del valor de la posició
public String posCorrecteDesti (String posicioIndefinida) {


//pre: Les posicions han estat validades
//post: S'ha realitzat una tirada
public String ferTirada (String tirada) {

//pre: --
//post: Es tanca la partida amb un empat
public void taulesTornsInaccio() {

//pre: --
//post: Es tanca la partida amb un empat.
public void taulesEscacsSeguits() {

//pre: --
//post: Es tanca la partida amb un empat.
public void taulesPerReiOfegat() {

//pre: --
//post: Es tanca la partida amb el guanyador siguent l'equip contrari
public void rendirse () {

//pre: --
//post: Es tanca la partida amb un empat
public void taules () {

//pre: --
//post: Es tanca la partida, sense resultat, per a poder segui-la
public void ajornar () {

//pre: --
//post: Es mostra el taulell en format text
public void mostrarTaulell () {
```

```
//pre: Posició p té una peça
//post: Retorna la peça de la posició entrada
public Peca getPeca ( Posicio p) {


//pre: --
//post: Retorna el taulell
public Taulell getTaulell()

//pre: --
//post: Retorna el torn
public String getProperTorn()

//pre: --
//post: Retorna el limit de torns
public int getLimitTornsInaccio()

//pre: --
//post: Retorna el limit d'escacs seguits
public int getLimitEscacsSeguits ()

//pre: --
//post: Retorna les files
public int getFiles ()

//pre: --
//post: Retorna les columnes
public int getColumnes ()

//pre: posicions origen i destí vàlides.
//post: S'ha promocinat la peça del jugador actual i s'avisa al principal si ha sortit bé o no.
public String ferPromocio(String tirada, String nomPeça){

//pre: Hi ha alguna tirada per a desfer
//post: Desfet l'ultima tirada al taulell i al fitxer de partida
public TiradaSimple desferTirada (StringBuilder resultat) {

//pre: Hi ha alguna tirada per a refer
//post: Desfet l'ultima tirada al taulell i al fitxer de partida
public TiradaSimple referTirada (StringBuilder resultat) {

//pre: --
//post: Retorna un llista de String Peca de cada tipus de Peça de la partida
public String [] getLlistaPeces() {

//pre --
```

//post Es tanca la partida, guanya el jugador actual.
public void escacIMat () {

Tipus PecaGrafica

Descripció general: Administrador de la interfície de la peça (quin aspecte tindrà en mode gràfic).

Operacions:
//pre: --
//post: Retorna lstring simbol
public String get_simbol() {

//pre:--
//post:Retorna lstring del imatge blanca
public String get_imgBlanca() {

//pre:--
//post:Retorna lstring del imatge blanca
public String get_imgNegra() {

//pre:--
//post:Retorna el valor de valor
public int get_valor() {

//pre:--
//post:Retorna la llista de moviments
public ArrayList<Moviment> get_llistaMoviments() {

//pre:--
//post:retorna la llista de moviments inicials
public ArrayList<Moviment> get_llistaMovimentsInicials() {

//pre:--
//post:retorna la llista de moviments d'enroc
public ArrayList<CaracteristiquesEnroc> get_llistaEnrocs() {

//pre: Tots els camps s'han omplert correctament
//post: Totes les variables per a assignar un tipus de peça estan settejades.
public TipusPeca(String nom, String simbol, String blanca, String negra, int valor, ArrayList<Moviment> listMov, ArrayList<Moviment> listIni, boolean promocio, boolean invulnerabilitat){

//pre TiradaSimple t és correcte
//post S'ha validat si el tipus de Peça pot realitzar la tirada t.
public boolean tipusPecaValida(TiradaSimple t, boolean primerMov){

//pre:--
//post:retorna lstring nom
public String get_nom(){

//pre:--
//post:retorna el boolea invulnerabilitat
public boolean get_invulnerabilitat()

//pre: --
//post: S'ha afegit l'enrroc amb caracteristiques c
public void afegirEnrroc(CaracteristiquesEnroc c){

//pre: --
//post S'ha validat l'Enrroc
public boolean validarEnrroc(Enrroc e, Peca b){

//pre: --
//post: retorna el boolea de promocio
public boolean getPromocio(){

Tipus Rajola

Descripció general: Administrador de la interfície de la rajola.

Operacions:
//pre:--
//post:
public Rajola(Image ima, int amplada)

Tipus Taulell

Descripció general: Matriu que defineix les dimensions que tindrà el taulell durant una partida.

Operacions:

//pre: c i f nombres enters entre 4 i 16
//post: Taulell creat de dimensio f x c
public Taulell(int c, int f){

//pre--:
//post: retorna el numero de files
public int getFiles() {

//pre--:
//post: retorna el numero de columnes

```
public int getColumnes() {

//pre: b i n diferent de null
//post: peces afegides al taulell
public void afegirPeca(Peca b, Peca n){

//per: peca i pos diferents de null
//post: peca afegides al taulell a la posicio pos
public void assignarPecaTauler(Peca peca, Posicio pos){

//pre: t diferent de null
//post: mira si la tirada es valida o no
public boolean tiradaValida(TiradaSimple t){

//pre: --
//post: Retorna una Posició amb una Peça del equip demanat.
public Posicio escollirPosPeca(boolean equip){

//pre: --
//post: mira si hi ha escac o no
public TiradaSimple hihaJaque(boolean equip){

//pre: --
//post: mira si hi ha escac i mat o no
public boolean hiHaJaqueMate(TiradaSimple t){

//pre:--
//post: mira si alguna peça pot ficarse en mig la tirada o no
public boolean potFicarseMig(TiradaSimple t){

//pre: --
//post: mira si se pot matar a la peça amenaçant o no
public boolean potMatarPeça(TiradaSimple t){

//pre: --
//post: mira si el rei pot tirar o no
public boolean potTirarRei(TiradaSimple t){

//pre: t diferent de null
//post: mira si es pot fer l'habilitat de matar que s'executa en la tirada
private boolean validMatar(TiradaSimple t, Peca p){

//pre: t diferent de null
//post: mira si es pot fer l'habilitat de volar que s'executa en la tirada
private boolean validVolar(TiradaSimple t){
```

```
//pre: t diferent de null
//post: mira si hi ha peces entremig de t o no
public boolean hiHaPecesEntremig(TiradaSimple t){

//pre: t diferent de null
//post: tirada feta i taulell modificat
public int realitzarTirada(TiradaSimple t){

//pre: t diferent de null
//post: mira si hi ha peces entremig de t o no i les mata si n'hi ha
private int eliminarPecesEntremig(TiradaSimple t, TreeMap<Posicio, Peca> eli){

//pre: cert
//post: posicio del rei trobada
public Posicio buscaRei(boolean equip){

//pre: cert
//post: retorna el map de peces eliminades
public TreeMap<Posicio,Peca> getEliminats(){

//pre: e diferent de null
//post: mira si l'enroc es valid o no
public boolean validarEnrroc(Enrroc e){

//pre: cert
//post: mira si el taulell te una peça a la posicio p
public boolean contePeçaCasella(Posicio p){

//pre: cert
//post: retorna la peça del taulell que esta a la poscio p
public Peca getPeca(Posicio p) {

//pre: cert
//post: mira si es pot fer promocio o no
public boolean hiHaPromocio(Posicio p, boolean _equip){

//pre: peça i pos existeixen
//post: promocio feta
public void realitzarPromocio(Posicio pos, Peca pec){

//pre: cert
//post: mira si _tiradesRefer esta buit o no
public boolean estaBuidaRefer(){

//pre: --
//post: Refa les tirades a partir d'una partida anterior per a que es pugui seguir.
```

```
public void carregarTirades(TiradaSimple t, String resultat, TreeMap<String,TipusPeca>
mapTipus, String Torn) {

//pre: --
//post: tirada desfeta
public TiradaSimple desferTirada(TiradaSimple t, String resultat,
TreeMap<String,TipusPeca> mapTipus){

//pre: --
//post: tiradesRefer reiniciat
public void reiniciaTiradesRefer(){

//pre: s'ha fet  minim un desfer previament
//post: tirada refeta
public TiradaSimple referTirada(StringBuilder resultat){

//pre: --
//post: moviments retorna la Tirada que s'ha de fer per a matar la Peça amenaçada
public TiradaSimple hiHaAlgunaAmenaça(boolean equip){

//pre: Posició dins el Taulell, Peça != null
//post: Retorna la Tirada que amenaça a la Peça
public TiradaSimple posAmenaçada(Posicio p, Peca pObservada, boolean equip, String rei)
{

//pre: taulell creat
//post: mostra el taulell
public String mostra() {
```

Tipus Posicio

Descripció general: Controlador de posicions del taulell

Operacions:

```
//pre: p té un valor vàlid
//post: S'ha assignat la posició entrada
public Posicio(String p){

//pre: fila i columna tenen un valor vàlid
//post: S'ha assignat la posició entrada
public Posicio(int fila, int columna){

//pre: --
//post: retorna la fila
public int get_fila(){
```

//pre: --
//post: retorna la columna
public int get_columna(){

//pre: --
//post: retorna la posicio amb una string
public String get_posicio(){

//pre: Posició o és valida
//post: 0 si són iguals, > 0 si l'actual és més gran, < 0 si l'actual és més petita
public int compareTo(Posicio o) {

//pre: Posició o és valida
//post: true iguals, false altrament
public boolean equals(Object o) {

Tipus Jugador (és una classe abstracta, té dades a dintre)

Descripció general: Gestiona les tirades de la partida

Operacions:

//pre: --
//post: Jugador creat
Jugador(boolean equip){

//pre: --
//post: retorna l'equip
public boolean get_equip() {

//pre --
//post retorna 0 si no s'ha fet la tirada, mes gran de 0 si s'ha fet
public int ferTirada(Taulell taulell, Posicio origen, Posicio desti){

//pre: cert
//post: mira si es pot fer promocio o no
public boolean observarPromocio(Posicio desti, Taulell taulell){

//pre: peça i pos existeixen
//post: promocio feta
public void ferPromocio(Posicio posicio, Taulell taulell, Peca p){

//pre: --
//post: mira si hi ha escac o escac i mat o no
public int observarJaque(Taulell taulell){

//pre: --
//post: mira si hi ha escac o no
public boolean shaProvocatJaque(Taulell taulell){

//pre --
//post cert si es fa l'enrroc, fals altrament
public boolean ferEnrroc(Taulell taulell, Posicio p1, Posicio p2){

Tipus CPU (implementa els mètodes de la interfície de Jugador)

Descripció general: Gestiona totes les jugades que fa la CPU, tant predir jugades com agafar-ne de partides passades per a mirar de guanyar al adversari

Operacions:

//pre: --
//post: El jugador CPU sap el seu equip
CPU(boolean equip)

//pre: El taulell està actualitzat
//post: S'ha realitzat una tirada automàtica
public void ferTirada(Taulell taulell){

Tipus Peça

Descripció general: Defineix les propietats generals d'una peça.
Operacions:

//pre: --
//post: Peça buida generada
public Peca(){

//pre: m té mapejat un Tipus de Peça amb el nom s
//post: Peça nova generada
public Peca(String s, boolean equip, TreeMap<String,TipusPeca> m){

//pre: m té mapejat un Tipus de Peça amb el nom s
//post: Peça nova generada amb atribut moguda definit
public Peca(String s, boolean equip, TreeMap<String,TipusPeca> m, boolean moguda){

//pre: --
//post: S'ha vàlidat la tirada
public boolean movimentValid(TiradaSimple t){

```
//pre: --
//post: retorna l'equip
public boolean get_equip(){

//pre: --
//post: Atribut _primerMoviment canviat
public void primerMovFet(){

//pre: --
//post: true si la Peça actual és el REI, fals altrament
public boolean esRei(boolean equip){

//pre: --
//post: S'ha vàlidat si les dues Peces no han fet el primer Moviment
public boolean enrrocValid(Enrroc e, Peca b){

//pre: --
//post: true es diuen igual, false altrament.
public  boolean esPecaNom(String nom){

//pre: --
//post: retorna el nom de la peça
public String getNom(){

//pre: --
//post: retorna la invulnerabilitat de la peça
public  boolean esInvulnerable(){

//pre: --
//post: retorna el tipus de la peça
public TipusPeca get_tipus() {

//pre: --
//post: Retorna una lletra que és la primera del nom de la Peça, en majuscula o minuscula.
public String toString(){

//pre: --
//post: retorna si la peça pot fer promocio
public boolean getPromocio(){

//pre: --
//post: moviments incrementat
public void incrementarMov(){

//pre: --
//post: moviments decrementat
```

public void decrementarMov(){

//pre: --
//post: retorna el numero de moviments de la peça
public int getNMovs(){

//pre: --
//post: moviments restaurats
public void restaurarMov(){

Tipus CaracteristiquesEnroc

Descripció general: Administra els possibles enrocs de cada peça
Operacions:

//pre: --
//post: Atributs assignats
public CaracteristiquesEnroc(String pecaA, String pecaB, boolean quiets, boolean buitAlmig){

//pre: Enrroc != NULL
//post: Enrroc validat
public boolean enrrocValid(Enrroc e, Peca b){

Tipus Moviment

Descripció general: Administra els possibles moviments de cada peça
Operacions:

Moviment(char vertical, char horitzontal, int saltar, int matar)
pre: --
post: s'ha definit un moviment

public int potSaltar()
pre: --
post: retorna 0 si no pot saltar, 1 si pot saltar i 2 si salta matant

public int potMatar()
pre: --
post; retorna la classe de forma de matar que pot executar.

public String get_horitzontal()
retorna el valor de moviment horitzontal que té el Moviment

public String get_vertical()
retorna el valor de moviment vertical que té el Moviment

public int get_movMatar()
retorna la capacitat de matar que té una Peça

public int get_movSaltar()
retorna la capacitat de saltar que té una Peça

public boolean movimentValid (TiradaSimple t)
pre: t és el tauler de joc.
post: true si el moviment és vàlid, false altrament.

public boolean valid(int desp, String s)
pre: desp és el desplaçament i s és l'eix ón es desplaça.
post: true si el moviment és vàlid, false altrament.

Tipus Historial

Descripció general: Guarda el conjunt de moviments realitzats durant la partida i ens permet desfer i refer
Operacions:

public static void iniciarPartidaNova(String fitxerRegles) throws Exception
pre: --
post: Totes les variables estan inicialitzades

public static void carregarPartidaAnterior(String path, LlegirFitxers fitxerEntradaPartida) throws IOException
pre: Path és correcte i el fitxerEntradaPartida existeix i té el format i la informació correcte
post: Es guarda en la variable de la classe , partida, tota la informació referent a la partida carregada.

public static boolean esEnter(String s)
pre: --
post: true si s és un enter, false altrament

public static void guardarPosInicial(Posicio posicio, Peca peca, boolean moguda, boolean equip)
pre: --
post: Totes les variables han estat guardades al objecte JSON

public static void guardarFitxerRegles (String nomFitxerRegles)
pre: --
post: Nom del fitxer de Regles guardat al objecte Partida JSON

public static void guardarProperTorn (String properTorn)
pre: --

post: Nom del Equip que ha de fer el proper Torn guardat al objecte Partida JSON

public static void guardarTirada(TiradaSimple t, String resultat)
pre: --
post: Tots els atributs de tirada i resultat guardats en el fitxer JSON

public static TiradaSimple getUltimaTirada()
pre: S'ha realitzat alguna Tirada
post: S'ha retornat l'objecte TiradaSimple

public static void eliminarUltimaTirada()
pre: S'ha realitzat alguna Tirada
post: S'ha eliminat l'últim element de la llista de tirades

public static void guardarPartida(String resultat)
pre: --
post: S'ha guardat una partida en un Fitxer assignant un resultat.

public static void modificarResultatUltimaTirada(String resultat)
pre: --
post: El resultat s'ha sobreescrit en l'últim element de la llista de Tirades del objecte JSON

public static String getUltimResultat ()
pre: --
post: retorna el valor del resultat de l'última Tirada de la llista de Tirades.

public static String getResultat (int i)
pre: i està dins el rang de tirades fetes.
post: Retorna un String amb el resultat de la tirada i.

public static String getTorn (int i)
pre: i esta dins el rang de tirades fetes.
post: Retorna un String amb el torn de la tirada i.

public static TiradaSimple getTirada(int i)
pre: i esta dins el rang de tirades fetes.
post: Retorna un objecte TiradaSimple corresponent a la tirada i.

public static int longTiradades()
retorna la quantita de tirades realitzades en la partida actual.

public static void afegirResultatUltimaTirada(String resultat)
pre: --
post: S'ha guradat a partida el resultat de l'ultima tirada.

public static boolean fitxerTiradesBuit()
retorna cert si l'Array de tirades és buit, false altrament.

Tipus TipusPeça

Operacions:

TipusPeca(String nom, String simbol, String blanca, String negra, int valor, ArrayList<Moviment> listMov, ArrayList<Moviment> listIini, boolean promocio, boolean invulnerabilitat);
Pre: Tots els camps s'han omplert correctament
Post: Totes les variables per a assignar un tipus de peça estan settejades.

public String get_simbol()
retorna el simbol del tipus de Peça.

public String git_imgBlanca()
retorna la drecera de l'imatge blanca

public String git_imgNegra()
retorna la drecera de la imatge negre

public int get_valor()
retorna el nombre que dóna valor a la Peça

public ArrayList<Moviment> get_llistaMoviments()
retorna la llista de Moviments que té el tipus de peces actuals.

public ArrayList<Moviment> get_llistaMovimentsInicials()
retorna la llista de Moviments Inicials que té el tipus de peces actuals.

public ArrayList<CaracteristiquesEnroc> get_llistaEnrocs()
retorna la llista de Enrrocs que té el tipus de peces actuals.

public boolean tipusPecaValida(TiradaSimple t, boolean primerMov)
Pre: TiradaSimple t és correcte.
Post: S'ha validat si el tipus de Peça pot realitzar la tirada t.

public String get_nom()
retorna el nom del tipus de Peça.

public String get_invulnerabilitat()
retorna el valor d'invulnerabilitat de la Peça.

public void afegirEnroc(CaracteristiquesEnroc c)

Pre: --
Post: S'ha afegit l'enrroc amb caraterístiques c.

public boolean validarEnroc(Enroc e, Peca b)
Pre: --
Post: S'ha validat l'Enrroc.

public boolean getPromocio()
retorna el valor de la Promoció.

Tipus LlegirFitxers

Descripció general: Lectura dels fitxers.
Operacions:

public void llegirPartides(string color);
Pre: --
Post: S'han llegit els fitxers de partides i s'han guardat les que són útils per a la CPU. Es guarden les que son del mateix color de la CPU i han guanyat.

public JSONArray getTirades()
retorna la llista de Tirades del fitxer que llegeix

public JSONArray getPosIniBlanques()
retorna la llista de Posicions Inicials de les Peces Blanques

public JSONArray getPosIniNegres()
retorna la llista de Posicions Inicials de les Peces Negres

public TreeMap<String, TipusPeca> getConjuntPeces()
retorna el Conjunt de Peces de la Partida.

public Taulell getTaulell()
retorna l'Objecte Taulell ja construït. .

public int getLimitTornsInaccio()
retorna la quantitat de torns que es pot estar amb inacció

public int getLimitEscacsSeguits()
retorna la quantitat d'escacs que es poden fer seguits

public String getResultatFinal()
retorna el resultat final del fitxer llegit.

public String getProperTorn()

retorna l'equip que ha de començar la partida

public String getFitxerRegles()
retorna el nom del fitxer de Regles de la partida actual.

public void llegirRegles(String path, boolean comencada) throws Exception
Pre: Path és vàlid
Post: Totes les variables del fitxer de Regles estan assignades.

public void llegirPartidaAcabada(String path) throws Exception
Pre: Path és vàlid
Post: Totes les variables del fitxer de Partida estan assignades. La Partida està acabada.

public void llegirPartidaComencada(String path) throws Exception
Pre: Path és vàlid
Post: Totes les variables del fitxer de Partida estan assignades. La Partida no està acabada.

Tipus TiradaSimple (hereda de Tirada)

Descripció general:
Operacions:

public TiradaSimple(Posicio origen, Posicio desti, boolean equip);
Pre: Origen, desti i equip són correctes
Post: Totes les variables de la tirada estan ben posades

public boolean is_equip()
retorn el valor del equip que ha realitzat la tirada

public TiradaSimple(Posicio origen, Posicio desti, boolean equip, int matar, int volar)
Pre: Origen, desti, equip, matar i volar són correctes
Post: Totes les variables de la tirada estan ben posades

public Posicio get_origen()
retorn el valor de la Posició origen de la Tirada

public Posició get_desti()
Retorna el valor de la Posició destí de la Tirada

public int get_desplacamentX()
retorna el valor del desplaçament de files.

public int get_desplacamentY()
retorna el valor del desplaçament de columnes.

public void canviaSigneDesplacament();
Pre: --
Post: Desplaçament X i Desplaçament Y tenen els signes invertits.

public void guardarMatar(int n)
Pre: --
Post: A Matar s'hi assigna el valor n.

public void guardarVolar(int n)
Pre: --
Post: A Volar s'hi assigna el valor n.

public boolean get_equip()
retorn el valor de l'equip que realitza la tirada.

public int get_matar()
retorn el valor de Matar.

public int get_volar()
retorn el valor de Volar.

Tipus Enrroc (hereda de Tirada)

Descripció general:
Operacions:

public Enroc(Posicio a, Posicio b, boolean equip)
Pre: --
Post: Atributs assignats

public Enrroc(Posicio a, Posicio b, boolean equip, boolean quiets, boolean buitAlMig)
Pre: --
Post: Atributs assignats

public Posicio get_p1()
retorna la posició p1

public Posicio get_p2()
retorna la posició p2

public boolean get_equip()
retorna l'equip
void assignarCarct(boolean quiets, boolean buit)
Pre: --
Post: S'han guardat les característiques de quiets i buit.

public boolean getQuiets()
retorna true si s’han mogut les peces, false altrament.

public boolean getBuit()
retorna true si hi ha espai per a fer l’enrroc, false altrament.

public String realitzarEnrroc(Taulell t)
Pre: --
Post: retorna l'enrroc que s'ha realitzat.