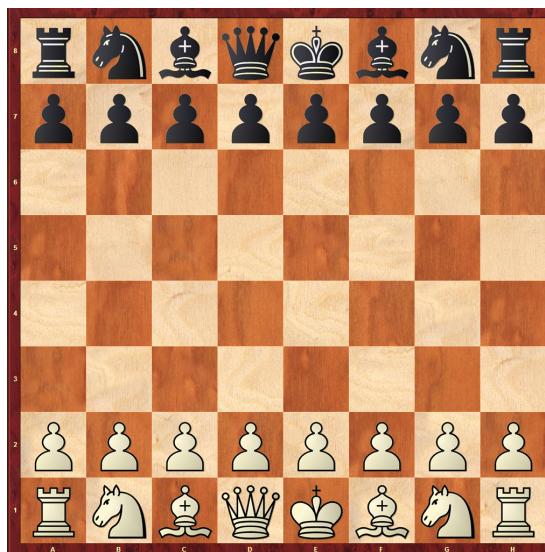


# SAE R 201 :

## Modélisation et implémentation d'un programme arbitrant une partie d'échecs entre 2 joueurs humains

Sujet proposé par  
Hanene AZZAG et Dominique Bouthinon

*BUT Informatique*  
*IUT de Villetaneuse Université Sorbonne Paris Nord*



### Note

Il n'est pas nécessaire de savoir jouer aux échecs pour faire ce projet, le sujet suffit (voir en annexe pour les règles du jeu d'échecs). Le joueur au trait est celui dont c'est le tour de jouer.

## 1 Présentation

L'objectif du projet est de modéliser en UML (diagramme de classes) et d'implémenter en java une partie d'échecs simplifiée (ni roque, ni promotion, ni prise en passant) entre 2 joueurs humains, la partie étant arbitrée par le programme.

### 1.1 Fonctions du programme

Ce programme doit :

- demander et enregistrer le nom de chaque joueur,
- afficher (en mode texte) la position courante de la partie,
- (optionnellement) afficher (en mode textuel) 1 horloge pour chaque joueur indiquant le temps de réflexion total (temps de réflexion pris pour l'ensemble des coups joués par le joueur) et le temps de réflexion écoulé depuis que le joueur a le trait (c'est à son tour de jouer),

- afficher les coups joués par chaque joueur,
- saisir le coup proposé par le joueur ayant le trait,
- valider ou invalider le coup proposé,
- détecter les situations de fin de partie et déclarer le vainqueur ou partie nulle.

## 1.2 Déroulement d'une partie

En début de partie les pièces sont disposées comme dans la figure de la page de titre. Dans votre représentation de l'échiquier les pièces blanches sont obligatoirement en bas. Dès qu'un joueur a le trait le programme :

1. affiche la position courante et la liste des coups joués par les 2 joueurs,
2. (optionnellement) démarre l'horloge du joueur au trait et arrête l'horloge du joueur adverse,
3. saisit le coup proposé par le joueur au trait,
4. examine le coup et tant qu'il n'est pas valide revient à l'étape précédente,
5. enregistre le coup,
6. vérifie si la partie est terminée, dans l'affirmative proclame le vainqueur ou une partie nulle, sinon on reprend à l'étape 1.

## 2 Organisation du projet

### 2.1 Modélisation

Il est important de correctement définir les objets du monde réel impliqués dans une partie d'échecs (joueur, échiquier, pièces etc...), chaque type d'objet sera représenté par une classe dans la modélisation.

Ainsi vous devez au minimum considérer :

- une classe *Partie* dont la responsabilité est d'arbitrer la partie, gérer les scores, les pendules etc.,
- une classe *Echiquier*,
- une classe *Case* correspondant à une case de l'échiquier,
- des classes représentant les types de pièce (tout ce qui est commun aux différentes pièces doit être mis dans une classe *Pièce*),
- une classe *Joueur*.

Pour modéliser ce système, vous devrez utiliser la méthode suivante :

1. Proposer un premier diagramme des classes UML faisant apparaître tous les liens entre classes (associations et héritages) sans préciser les attributs (variables) et les opérations (méthodes).
2. Après analyse des interactions au sein d'une partie ajouter les attributs et opérations. Ne pas modifier le diagramme précédent, faire apparaître chaque classe sur un autre diagramme avec ses attributs et opérations (dont constructeurs).

NOTE IMPORTANTE :

- Chaque type de pièce (Tour, fou, etc..) sait comment elle se déplace (voir annexe). Ainsi, proposer pour chaque type de pièces une méthode *boolean deplacement(Case destination)* qui retourne *true* si le déplacement de la pièce de sa position actuelle vers la position de destination correspond à un déplacement correct, sans tenir compte des autres pièces sur l'échiquier ni que le déplacement est hors échiquier, sinon *false*. Par exemple le déplacement d'une tour où la case actuelle et la case de destination sont les extrémités d'une diagonale est incorrect.

Il est important que la classe *Piece* contienne la méthode abstraite *boolean deplacement(Case destination)* afin de mettre en oeuvre le polymorphisme.

- la classe *Partie* gère la correction des déplacements par rapport au contenu et aux dimensions de l'échiquier. Ainsi un coup de *Tour* peut correspondre à un déplacement correct, vertical ou horizontal, mais s'avérer impossible sur l'échiquier, soit parce qu'il est hors échiquier, soit parce que d'autres pièces font obstacle au déplacement.

## 2.2 organisation du travail

Le développement de l'application s'effectue en binôme. NOTE IMPORTANTE : pour ne pas faire un double travail il est fortement recommandé que chaque développeur prenne en charge une classe entière.

## 2.3 Implémentation

L'implémentation java s'effectue APRES la modélisation UML et doit strictement respecter cette dernière. S'il s'avère que le modèle UML n'est pas adéquate, vous le modifierez avant de poursuivre l'implémentation. Toute rétro-analyse consistant à concevoir le modèle UML à partir d'un programme java est très visible et sera FORTEMENT PENALISEE.

L'utilisation du polymorphisme est obligatoire dès qu'il est possible. La clarté et la concision du programme seront considérées : le programme doit être commenté de façon synthétique, les noms des classes, variables et méthodes seront évocateurs et les méthodes contiendront peu de lignes.

L'affichage des menus et de l'échiquier doivent être (le plus possible) indépendants de la gestion du jeu.

## 2.4 Modalités de remise du projet

**Ce projet est à réaliser en binôme.** Le dossier de rendu comprendra :

1. le code java de toutes les classes sous forme d'un fichier zip (obligatoire)
2. un document pdf (obligatoire) comprenant :
  - les noms, prénoms et groupe du binôme,
  - une indication précisant comment lancer le programme. NOTE IMPORTANTE : LE PROGRAMME DOIT ETRE LANCE DEPUIS UN TERMINAL sans avoir besoin d'ouvrir un environnement de développement intégré (IDE) type Eclipse,
  - les diagrammes des classes,
  - un bilan précisant ce qui a été fait, non fait, ce qui fonctionne et ce qui ne fonctionne pas,
  - la liste des classes prises en charge par chaque membre du binôme

Ces deux documents (zip et pdf) doivent être envoyés obligatoirement par mail à votre chargé de travaux dirigés au plus tard le 10 Juin 2025 minuit. Aucun envoi après cette date ne sera considéré.

## 2.5 Evaluation du projet

La modélisation UML, la qualité de la programmation, la correction de l'exécution du programme et le bilan seront évalués.

Les programmes seront vérifiés par un système de détection de plagiat. TOUT PLAGIAT ENTRE BINÔMES OU AVEC UNE INTELLIGENCE ARTIFICIELLE SERA TRÈS FORTEMENT PÉNALISÉ.

LA NOTE SERA INDIVIDUELLE, ainsi les deux membres d'un binôme pourront avoir des notes différentes en fonction des classes qu'il a développées (les responsables des travaux dirigés interrogeront les binômes).

### 3 Annexe : règles du jeu d'échecs

Le jeu d'échecs se joue entre deux adversaires qui déplacent alternativement des pièces sur un échiquier. Le joueur qui joue en premier possède les pièces blanches, son adversaire les pièces noires. L'objectif de chaque joueur est de capturer le roi adverse. Cependant, l'usage est de ne pas prendre le roi mais de signaler au joueur adverse que son roi ne peut plus échapper à la prise en disant "échec et mat". Le joueur dont le roi est *maté* a perdu la partie.

#### 3.1 L'échiquier

L'échiquier comporte 64 cases claires (blanches) et sombres (noires). On distingue :

- les colonnes, chacune formée de 8 cases verticales. Il y a 8 colonnes étiquetées de *A* à *H*.
- les rangées, chacune formée de 8 cases formant une ligne horizontale. Il y a 8 rangées numérotées de 1 à 8.

Ainsi chaque case de l'échiquier est identifiée par une étiquette de colonne et un numéro de rangée, par exemple *e2*. La colonne *A* est formée des cases *a1* à *a8*, la rangée 2 des cases *a2* à *h2*.

#### 3.2 Les pièces

Chaque joueur possède 16 pièces de même couleur (blanches ou noires) :

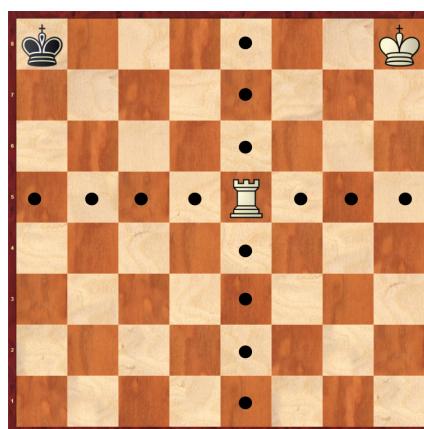
- 1 roi,
- 1 dame,
- 2 fous,
- 2 cavaliers,
- 2 tours,
- 8 pions.

#### 3.3 Le mouvement des pièces

Tout mouvement d'une pièce est un déplacement de la case où elle se situe (case de départ) vers une autre case de l'échiquier (case d'arrivée). Si la case d'arrivée contient une pièce d'une couleur différente de la pièce déplacée on la retire de l'échiquier (elle est capturée).

##### 3.3.1 Mouvements de la tour

La tour se déplace sur n'importe quelle case située dans la rangée (déplacement horizontal) ou la colonne (déplacement vertical) de la case de départ.



### 3.3.2 Mouvements du fou

Le fou se déplace sur n'importe quelle case située dans une des 2 diagonales comprenant la case de départ.



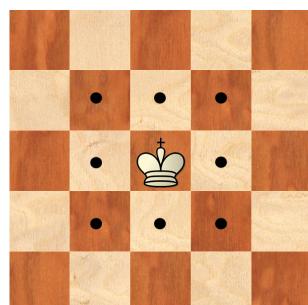
### 3.3.3 Mouvements de la dame

La dame se déplace comme une tour ou comme un fou (à chaque mouvement on décide soit un déplacement de tour soit un déplacement de fou).



### 3.3.4 Mouvements du roi

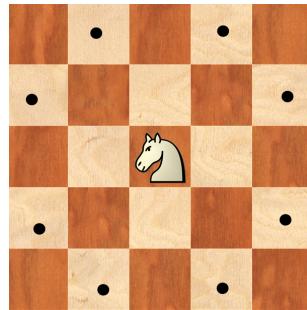
Le roi se déplace sur l'une des cases adjacentes à la case de départ.



### 3.3.5 Mouvements du cavalier

Le cavalier peut se déplacer sur toute case de l'échiquier située à  
— une rangée et deux colonnes de sa case de départ,  
— une colonne et deux rangées de sa case départ.

On peut représenter le déplacement d'un cavalier par un "L" (vertical ou horizontal) entre la case de départ et la case d'arrivée.

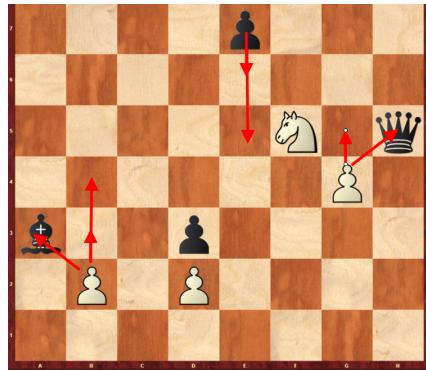


### 3.3.6 Mouvements du pion

Le pion possède 2 déplacements différents :

- déplacement sans capture : le pion se déplace sur la case inoccupée située sur la rangée immédiatement devant lui. S'il s'agit de son premier déplacement, le pion peut (sans obligation) se déplacer sur la case inoccupée située 2 rangées devant lui.
- déplacement avec capture : le pion se déplace en diagonale vers une case située sur une colonne adjacente et une rangée devant lui, uniquement si la case d'arrivée est occupée par une pièce adverse qui est capturée.

Les pions blancs se déplacent vers le haut de l'échiquier (la rangée de la case d'arrivée est supérieure à celle de la case de départ), et les pions noirs vers le bas (la rangée de la case d'arrivée est inférieure à celle de la case de départ).



### 3.4 Validité d'un déplacement

Le déplacement d'une pièce (coup) est valide s'il satisfait les conditions suivantes :

- la case de départ et la case d'arrivée sont sur l'échiquier,
- la case de départ contient une pièce de la même couleur que les pièces du joueur ayant le trait,
- la case d'arrivée ne contient pas une pièce de même couleur que la pièce déplacée,
- il n'y a pas de pièce située entre la case de départ et la case d'arrivée, sauf si la pièce déplacée est un cavalier (il saute au dessus des pièces qu'il rencontre),
- à l'issue du déplacement le roi de même couleur que la pièce déplacée ne doit pas être en prise (une pièce est en prise si le camp adverse peut la capturer).

### 3.5 Fin de partie

La partie est terminée si le joueur ayant le trait rencontre une des situations suivantes :

- mat : son roi ne peut échapper à une prise (le joueur a perdu),
- pat : son roi n'est pas en prise mais le joueur ne dispose daucun coup valide (partie nulle).