MIMO **Février 2017**

Projet C

Jeu d’échecs

Cahier des charges

Hsuning CHANG **François D’HUBERT**

Sommaire

I. Enjeux 5
1.1 Objectif 5
1.2 Documentation 5
1.3 Description du jeu 5
1.4 Contenu du programme 5
II. Analyse 6
2.1 L'échiquier 6
2.2 Les pièces 6
2.3 Positions de départ 7
2.4 Mouvements des pièces 8
2.4.1 Règles générales 8
2.4.2 Prise d'une pièce adverse 8
2.4.3 Mouvements du pion 8
2.4.4 Mouvements de la tour 9
2.4.5 Mouvements du cavalier 9
2.4.6 Mouvements du fou 9
2.4.7 Mouvements de la dame 9
2.4.8 Mouvements du roi 10
2.4.9 Roque 10
2.5 Echecs et fin de partie 10
2.5.1 Echec au roi 10
2.5.2 Echec et mat 11
2.5.3 Partie nulle 11
2.6 Sauvegarde 11
2.7 Chargement 11

2.8 Quitter 11
III. Démarches de développement 12
3.1 Programme principal 12
3.2 Nouvelle partie 13
3.2.1 Procédure initialisation 16
3.2.2 Procédure sauvegarde 16
3.2.3 Fonction ifnoir 17
3.2.4 Procédure mouvement 18
3.2.5 Fonction verificationEntree 20
3.2.6 Fonction conv_colonne_Chartolnt 21
3.2.7 Fonction checkPiece 21
3.2.8 Fonction verifoccupe 22
3.2.9 Fonction mouvementsPossibles 22
3.2.10 Procédure departToArrivee 24
3.2.11 Fonction finLigne_pion 25
3.2.12 Fonction deplacementCavalier 27
3.2.13 Fonction deplacementHorVer 27
3.2.14 Fonction deplacementDiagonale 28
3.2.15 Fonction checkRoi 29
3.2.16 Fonction haspiece 30
3.2.17 Procédure roq 30
3.2.18 Fonction checkmat 31
3.2.19 Fonction PartieNull 31
3.2.20 Procédure doubledeplacement 31
3.2.21 Procédure doubledeplacementRoi 32
3.3 Charger une partie 32
3.4 Procédure quitter 32

IV. Difficultés rencontrées 33
4.1 Nommage des pièces 33
4.2 Fonction pour vider le buffer 33
4.3 Règles non appliquées 34
4.4 Textes sans accents 34
4.5 Lire les espaces dans le fichier 34
4.6 Partie nulle 34
V. Annexes 35
5.1 Les exigences 35
5.1.1 Plateau 35
5.1.2 Positions de départ 36
5.1.3 Mouvements 37
5.1.4 Echecs 39
5.1.5 Options 40

I. Enjeux

1.1 Objectif

Le projet doit permettre à un utilisateur unique de pouvoir faire une partie de jeu d'échecs contre lui-même à partir d'un programme écrit en langage C. Ce programme reprendra tous les éléments essentiels au déroulement du jeu : le plateau de jeu, les 32 pièces, les différents mouvements et les conditions de victoire. Le programme permettra à l'utilisateur de sauvegarder une partie et de la reprendre plus tard.

1.2 Documentation

Conformément aux conditions établies pour les projets concernant des jeux, aucune aide en ligne n'a été utilisée à propos du code. La documentation se résume donc aux règles du jeu établies par la Fédération Française des Échecs, lesquelles sont disponibles via le lien suivant : http://www.echecs.asso.fr/livrearbitre/110.pdf

Il est toutefois nécessaire de préciser que ces règles s'appliquent à des conditions de jeu comprenant un véritable plateau et deux joueurs s'affrontant dans un même lieu. Certaines règles ne peuvent donc pas être appliquées et le programme simplifiera certains éléments non essentiels au déroulement de la partie.

1.3 Description du jeu

Le jeu d'échecs est composé d'un plateau (échiquier) de 64 cases (8 cases sur 8) et de 32 pièces réparties en deux camps : les blancs et les noirs. Chaque camp possède les mêmes pièces : huit pions, deux tours, deux cavaliers, deux fous, un roi et une dame. Chaque type de pièce a un mouvement spécifique. Les deux camps jouent alternativement jusqu'à ce que l'un des rois soit maté ou qu'il y ait partie nulle. Le camp maté perd la partie.

1.4 Contenu du programme

Le programme sera constitué d'un menu permettant de :

- débuter une nouvelle partie
- charger une partie sauvegardée
- quitter

Au cours d'une partie, l'utilisateur aura le choix de :

- jouer un nouveau coup
- sauvegarder la partie
- quitter

L'échiquier sera visible en permanence. Le tour de jeu (blanc ou noir) sera affiché ainsi que le numéro du coup joué. Au commencement de la partie, le compteur affichera « 1 » pour « premier coup ». Ces deux éléments serviront d'indicateurs pour l'utilisateur afin qu'il puisse se repérer facilement dans le déroulement de la partie. Ces fonctionnalités seront surtout utiles lors du chargement d'une partie sauvegardée.

II. Analyse

2.1 L'échiquier

Le plateau de jeu comprend 64 cases (huit cases sur huit) numérotées de 1 à 8 en ordonnée (colonne) et de a à h en abscisse (ligne). Les 64 cases et les numérotations doivent apparaître en permanence à l'écran. Dans un souci de lisibilité, un espace sera introduit entre le bord gauche de l'écran et la numérotation en ordonnée. En revanche, aucun espace ne séparera une numérotation d'une case.

Chaque case est unique. Elle est nommée d'après sa ligne et sa colonne de la manière qui suit : a1, b1, c1, […], h8. Une case sera un rectangle de la forme suivante :

```
+-+-+-+-+
+       +
+       +
+-+-+-+-+
```

La case doit pouvoir contenir lisiblement une lettre symbolisant une pièce.

2.2 Les pièces

L'utilisateur doit pouvoir distinguer les pièces blanches des pièces noires facilement. Il est difficile d'arriver à une solution optimale à ce niveau car une seule couleur est possible.

Les pièces blanches seront symbolisées par un caractère majuscule renvoyant au nom français de cette pièce : P (pion), T (tour), C (cavalier), F (fou), D (dame) et R (roi).

Les pièces noires seront symbolisées par un caractère minuscule. Pour accentuer la différence, les pions noirs seront représentés par un « i » rappelant la forme générale de la

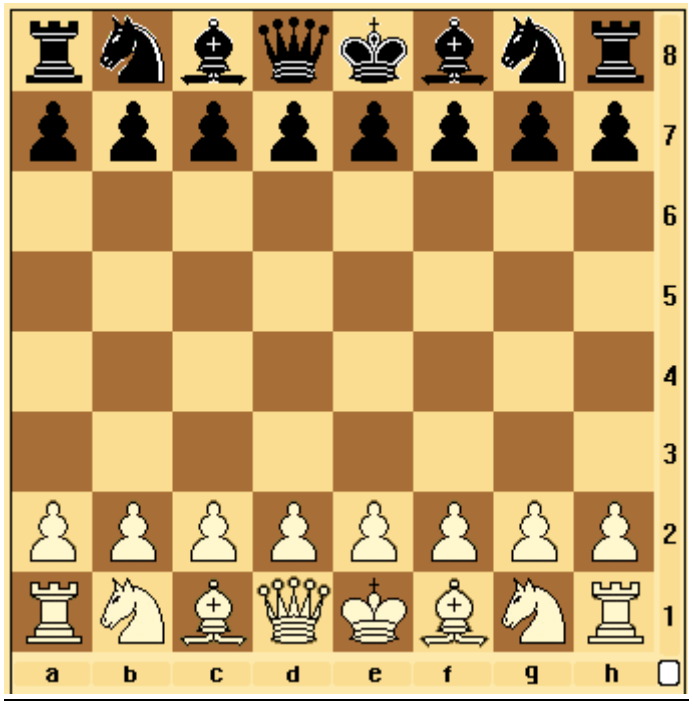
pièce. Ainsi, les pièces noires seront les suivantes : i (pion), t (tour), c (cavalier), f (fou), d (dame) et r (roi).

Les pièces ne sont pas nommées spécifiquement (par exemple, il n'y aura pas de P1, P2, P3, etc.) car cela complexifie les commandes, en particulier pour le cas spécifique d'un pion arrivant en bout de ligne (cf. 2.4.3).

2.3 Positions de départ

Le placement des pièces au début d'une partie est toujours le même. Les blancs sont positionnés en bas du plateau et les noirs en haut du plateau. La dame sera positionnée à gauche du roi dans les deux camps. Les positions des pièces en début de partie sont les suivantes :

	Blanc	**Noir**
Tour (2 pièce*2)	a1 et h1	a8 et h8
Cavalier (2 pièce*2)	b1 et g1	b8 et g8
Fou (2 pièce*2)	c1 et f1	c8 et f8
Dame (1 pièce*2)	d1	d8
Roi (1 pièce*2)	e1	e8
Pion (8 pièce*2)	a2, b2, c2, d2, e2, f2, g2 et h2	a7, b7, c7, d7, e7, f7, g7 et h7



Positionnement des pièces en début de partie

2.4 Mouvements des pièces

2.4.1 Règles générales

Les blancs jouent toujours en premier. L'utilisateur ne doit donc pas pouvoir faire bouger une pièce noire au premier coup.

Les deux camps jouent alternativement. Un camp ne peut donc effectuer qu'un seul et unique mouvement par coup. Une seule pièce sera déplacée, sauf dans le cas particulier du roque (voir 2.4.9).

Pour effectuer un mouvement, l'utilisateur sélectionnera un type de pièce (exemple : P), indiquera la case où elle se trouve (exemple : a2) puis sa case d'arrivée (exemple : a3).

Il est impossible d'annuler un mouvement. Cette règle fait référence au « pièce touchée = pièce jouée » : si l'utilisateur sélectionne une pièce, il sera obligé de la faire bouger, à condition que celle-ci ait au moins un mouvement autorisé.

2.4.2 Prise d'une pièce adverse

Une pièce peut effectuer un mouvement vers une case occupée à condition que la pièce présente dans cette case soit du camp opposé. Dans ce cas, la pièce qui occupait cette case disparaît définitivement de la partie.

2.4.3 Mouvements du pion

Un pion peut se déplacer d'une case inoccupée immédiatement devant lui sur la même colonne. Il ne peut jamais reculer.

Si un pion est en ligne 2 (blanc) ou ligne 7 (noir), il peut se déplacer d'une ou deux cases inoccupées immédiatement devant lui sur la même colonne.

Un pion peut se déplacer vers une case occupée par une pièce adverse située devant lui en diagonale sur une colonne adjacente.

Si un pion atteint sa ligne la plus éloignée de sa position de départ (ligne 8 pour les blancs, ligne 1 pour les noirs), l'utilisateur doit le remplacer au choix par une dame, un fou, un cavalier ou une tour.

2.4.4 Mouvements de la tour

Une tour peut se déplacer sur toutes les cases d'une ligne ou d'une colonne sur laquelle elle se trouve jusqu'à ce qu'elle rencontre un obstacle (pièce ou bord du plateau). Si cet obstacle est une pièce adverse, la tour peut prendre la case occupée mais ne pourra pas aller au-delà de celle-ci. Il sera nécessaire de contrôler que toutes les cases situées entre la case de départ et la case d'arrivée de la tour soient vides.

Une tour peut effectuer un roque sous certaines conditions (voir 2.4.9).

2.4.5 Mouvements du cavalier

Un cavalier peut se déplacer sur l'une des cases les plus proches de celle qu'il occupe à condition qu'elle ne se situe pas sur la même ligne, colonne ou diagonale. La case d'arrivée sera donc à deux lignes et une colonne ou à une ligne et deux colonnes de la case de départ. Exemple : pour une case de départ d5, la case d'arrivée pourra être c7, e7, f6, f4, e3, c3, b4 ou b6. Le cavalier d'emprunte pas de chemin à proprement parler. Il se place directement sur sa case d'arrivée si celle-ci n'est pas occupée par une pièce du même camp. Il ne prend pas en compte les autres cases environnantes.

2.4.6 Mouvements du fou

Le fou peut se déplacer sur une case située dans la diagonale de la case où il se trouve jusqu'à ce qu'il rencontre un obstacle (pièce ou bord du plateau). Si cet obstacle est une pièce adverse, le fou peut prendre la case occupée mais ne pourra pas aller au-delà de celle-ci. Il sera nécessaire de contrôler que toutes les cases situées entre la case de départ et la case d'arrivée du fou soient vides.

2.4.7 Mouvements de la dame

La dame cumule les mouvements du fou et de la tour. Elle peut donc se déplacer sur toutes les cases situées sur la même ligne ou colonne ou diagonale que sa case de départ jusqu'à ce qu'elle rencontre un obstacle (pièce ou bord du plateau). Si cet obstacle est une pièce adverse, la reine peut prendre la case occupée mais ne pourra pas aller au-delà de celle-ci. Il sera nécessaire de contrôler que toutes les cases situées entre la case de départ et la case d'arrivée de la reine soient vides.

2.4.8 Mouvements du roi

Le roi peut se déplacer sur une case adjacente à celle qu'il occupe (ligne, colonne ou diagonale). Il ne peut pas se déplacer sur une case menacée par un mouvement d'une pièce adverse.

Un roi peut effectuer un roque sous certaines conditions.

2.4.9 Roque

Le roque consiste en un mouvement d'un roi et d'une tour du même camp. Il s'agit du seul mouvement de jeu autorisant le déplacement de deux pièces. Un camp ne peut effectuer qu'un seul roque par partie. Le roque n'est plus possible si le roi ou la tour concernée ont déjà été déplacés lors de la partie en cours. Les cases séparant le roi et la tour sur la même ligne doivent être vides.

Lors du roque, le roi se déplace de deux cases vers la tour à partir de sa case initiale. La tour se déplace alors sur la dernière case vide traversée par le roi. Le roque est impossible si l'une des cases que le roi doit traverser ou celle qu'il doit occuper est menacée par un mouvement d'une pièce adverse. Le roque est également impossible si le roi est en échec.

2.5 Echecs et fin de partie

2.5.1 Echec au roi

Si le roi est menacé par le mouvement d'une pièce adverse, il est en échec. L'utilisateur a alors l'obligation de mettre le roi à l'abri des mouvements de l'adversaire :

- soit en déplaçant le roi vers une position non menacée
- soit en protégeant le roi de la menace par une pièce interposée
- soit en supprimant la menace, c'est-à-dire par la prise de la pièce dont le mouvement menace directement le roi

Si le roi est en échec, aucun mouvement de protégeant pas le roi n'est possible. De même, un mouvement mettant son propre roi en échec est interdit. Il faut donc vérifier, après avoir joué un coup, si le roi du camp agissant est en échec. Si c'est le cas, le mouvement doit être interdit.

2.5.2 Echec et mat

Il y a échec et mat si un roi de l'un des camps ne peut plus être mis à l'abri d'un mouvement de l'adversaire. Le roi ne peut plus se déplacer et aucune pièce ne peut le protéger. Le camp maté n'a donc plus de mouvement autorisé. La partie s'arrête lorsqu'il y a échec et mat. Le camp du roi en échec a perdu.

2.5.3 Partie nulle

Il y a partie nulle lorsqu'un camp ne peut plus effectuer de mouvement alors que son roi n'est pas en échec. La partie s'achève alors par une égalité. L'utilisateur doit pouvoir déclarer que la partie est nulle selon son bon vouloir.

2.6 Sauvegarde

L'utilisateur pourra sauvegarder sa partie à tout moment. S'il s'agit d'une première sauvegarde, il lui sera demandé de donner un nom à sa sauvegarde. Une partie non sauvegardée ne sera pas conservée dans la mémoire du programme.

2.7 Chargement

Le menu proposera à l'utilisateur de charger une partie sauvegardée. Celui-ci entrera le nom de sa sauvegarde. Le plateau s'affichera avec les positions des pièces, le compteur de coups joués et le tour de jeu correspondant à la situation de la partie au moment de la sauvegarde.

2.8 Quitter

L'utilisateur pourra quitter la partie à tout moment. Il aura l'option de sauvegarder sa partie. S'il ne le fait pas, il y a partie nulle.

III. Démarches de développement

3.1 Programme principal

Affichage :

Bienvenue dans le jeu d'échec !
1- Nouvelle partie 2- Charger une partie 0- Quitter

Fonctionnement :

Le programme principal affiche un menu proposant trois choix possibles visibles de l'utilisateur. Celui-ci indique son choix manuellement et donne ainsi une valeur (1, 2 ou 0) à une variable « choix ». Le choix « 1 » appelle les sous-programmes « initialisation » (cf. 3.2.1) et « nouvellePartie » (cf. 3.2), le choix « 2 » appelle le sous-programme « chargement » (cf. 3.3) et le choix « 0 » arrête le programme après avoir affiché le message « au revoir ». Si l'utilisateur entre une autre valeur que celles mentionnées, le message d'erreur « Choix erroné » s'affiche.

Pour entrer dans la boucle du menu, il faut initialement donner une valeur à « choix » différente de 0. Au démarrage du programme, « choix » prendra donc la constante CHOIX_NON_DEFINI égale à 99.

Recette :

Scénario 1 :
- L'utilisateur tape « 1 ».
- Les sous-programmes "initialisation" et "nouvellePartie" sont appelés.

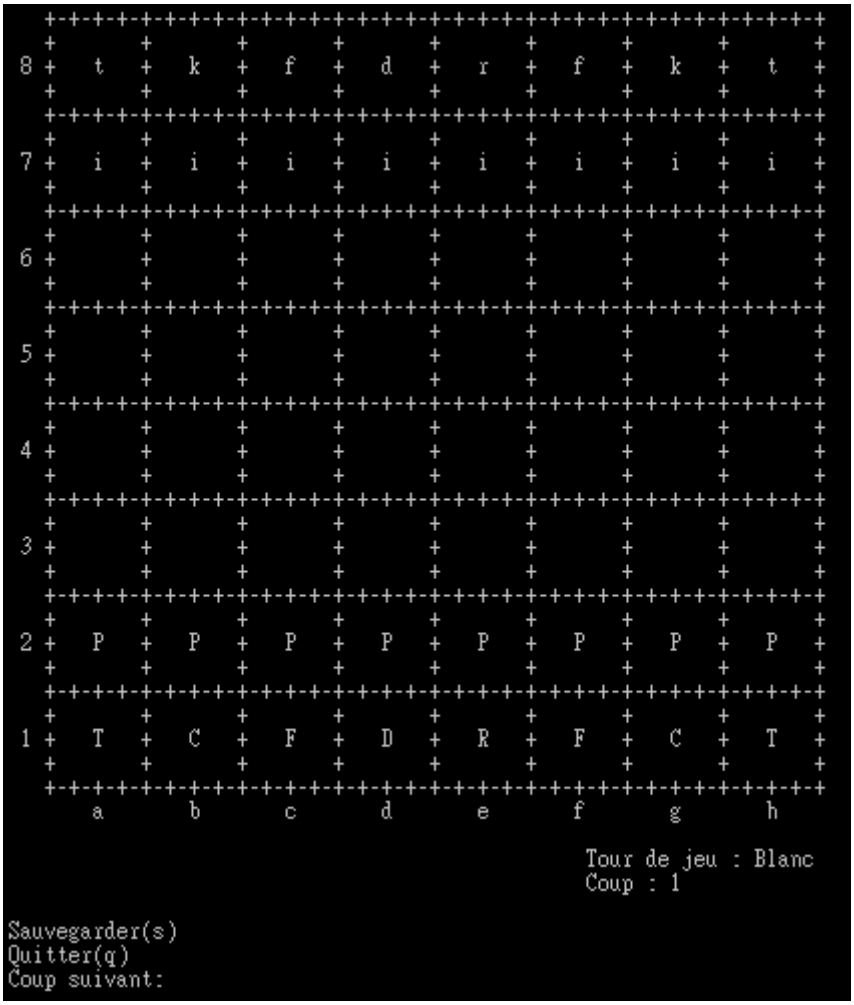
Scénario 2 :
- L'utilisateur tape « 2 ».
- Le sous-programme « chargement » est appelé.

Scénario 3 :
- L'utilisateur tape « 0 ».
- Le message « Au revoir » s'affiche.
- Le programme s'arrête.

Scénario 4 :
- L'utilisateur tape « a », « 3 » « -1 », « -2 », « 99 » ou appuie sur « Entrée ».
- Le message « Choix erroné » s'affiche.

3.2 Nouvelle partie

Affichage :



Fonctionnement :

La procédure « nouvellePartie » permet d'afficher l'échiquier avec les pièces dans leurs positions de départ fixées par les règles du jeu (cf. 2.3), le tour de jeu et le numéro du coup à jouer. Elle propose également à l'utilisateur de saisir une commande : sauvegarde, quitter ou mouvement d'une pièce.

En lançant une nouvelle partie, une procédure d'initialisation de l'échiquier est appelée (cf. 3.2.1), permettant de positionner les pièces à leur place réglementaire en début de partie. La variable globale « coupJeu » correspond au numéro du coup à jouer. Elle est fixée par défaut à 1. La variable globale « tour » correspond au tour de jeu. Elle prend par défaut la chaîne de caractères « Blanc » car les blancs jouent en premier.

Un test permet de modifier « tour ». Il utilise la fonction « ifnoir » (cf. 3.2.3) qui indique si « coupJeu » est pair. Si c'est le cas, « tour » prend la chaîne de caractères « Noir » via la fonction strcpy (rajouter la bibliothèque <string.h>), sinon elle prend la chaîne « Blanc ». En

effet, le tour des blancs correspondra toujours à un coup de jeu impair et celui des noirs à un tour pair, « coupJeu » étant à 1 par défaut et les blancs jouant en premier.

L'échiquier s'affiche selon le modèle de case défini en 2.1 : cinq « + » horizontaux reliés par des « - » et cinq « + » verticaux. Au niveau du troisième « + » vertical d'une case située sur le bord gauche de l'échiquier sera affiché le numéro de ligne (de 8 à 1) à l'extérieur du plateau. Un caractère est positionné au centre de chaque case. Il renvoie à une case de la variable globale « plateau » : un tableau de caractères à deux dimensions. Au niveau de la ligne 1 sont affichés, à l'extérieur du plateau, le tour de jeu puis, à la ligne suivante, le numéro du coup à jouer. Tout en bas de l'échiquier, à l'extérieur des cases, est affiché la numérotation des colonnes (de a à h). Un saut de ligne sera ensuite intégré pour plus de clarté.

L'utilisateur peut ensuite voir les options qui lui sont proposées. Il entre alors un caractère correspondant à la variable locale « coupSuivant ». Selon cette variable, plusieurs actions sont possibles :

- Si l'utilisateur tape « s », il appelle la procédure « sauvegarde » (cf. 3.2.2) permettant de sauver sa partie en cours.
- Si l'utilisateur tape « q », il appelle la procédure « quitter » (cf. 3.4) et quitte le sous-programme.
- Si l'utilisateur tape un caractère correspondant à une pièce blanche, un test vérifie si c'est le tour de jeu des blancs en appelant la fonction « ifnoir » (cf. 3.2.3). Si c'est le cas, la procédure « mouvement » (cf. 3.2.4) est appelée, sinon le message « Vous ne pouvez pas bouger une pièce blanche » s'affiche.
- Si l'utilisateur tape un caractère correspondant à une pièce noire, un test vérifie si c'est le tour de jeu des noirs en appelant la fonction « ifnoir » (cf. 3.2.3). Si c'est le cas, la procédure « mouvement » (cf. 3.2.4) est appelée, sinon le message « Vous ne pouvez pas bouger une pièce noire » s'affiche.
- Si l'utilisateur tape un autre caractère, le message d'erreur « commande incorrecte » s'affiche.

Tant que l'utilisateur ne donne pas « q » à la variable « coupSuivant », le sous-programme affiche l'échiquier à jour, les options possibles et propose à l'utilisateur de saisir à nouveau « coupSuivant ». Pour entrer dans la boucle, il faut initialement donner une valeur à « coupSuivant » différente de « q ». On attribue donc à « coupSuivant » la constante CHOIX_NON_DEFINI au démarrage de la procédure « nouvellePartie ».

L’utilisateur va entrer un caractère et un saut de ligne dans « coupSuivant », il faut donc vider le buffer de la mémoire en utilisant une variable globale « bidon » prévue à cet effet.

Après plusieurs coups de jeu, la procédure "roq" (cf. 3.2.17) est appelée. Celle-ci retourne le booléen "ifroq" qui, s'il est vrai, incrémente "coupJeu".

Si la fonction "checkRoi" (cf. 3.2.15) est vraie, alors le message "échec au roi" s'affiche avant la sélection du coup suivant.

Un test permet de vérifier si la pièce sélectionnée dans "coupSuivant" existe encore en appelant la fonction "hasPiece" (cf. 3.2.16). Si c'est le cas, la procédure "mouvement" est appelée (cf. 3.2.4).

Recette :

Scénario 1 :
- L’utilisateur tape « 1 » dans le menu principal.
- L’échiquier s’affiche avec les numérotations de ligne et de colonne bien positionnés. Le compteur de coup affiche « 1 ». Le tour de jeu affiche : « Blanc ».
- L’utilisateur tape « s » dans « Coup suivant ».
- La procédure « sauvegarde » est appelée.

Scénario 2 :
- L’utilisateur tape « 1 » dans le menu principal.
- L’échiquier s’affiche avec les numérotations de ligne et de colonne bien positionnés. Le compteur de coup affiche « 1 ». Le tour de jeu affiche : « Blanc ».
- L’utilisateur tape « q » dans « Coup suivant ».
- La procédure « quitter » est appelée.
- Le menu principal apparaît.

Scénario 3 :
- L’utilisateur tape « 1 » dans le menu principal.
- L’échiquier s’affiche avec les numérotations de ligne et de colonne bien positionnés. Le compteur de coup affiche « 1 ». Le tour de jeu affiche : « Blanc ».
- L’utilisateur tape « T », « C », « F », « D », « R » ou « P » dans « Coup suivant ».
- La fonction « ifnoir » est appelée.

Scénario 4 :
- L’utilisateur tape « 1 » dans le menu principal.
- L’échiquier s’affiche avec les numérotations de ligne et de colonne bien positionnés. Le compteur de coup affiche « 1 ». Le tour de jeu affiche : « Blanc ».
- L’utilisateur tape « t », « k », « f », « d », « r » ou « i » dans « Coup suivant ».
- La fonction « ifnoir » est appelée.

Scénario 5 :
- L’utilisateur tape « 1 » dans le menu principal.
- L’échiquier s’affiche avec les numérotations de ligne et de colonne bien positionnés. Le compteur de coup affiche « 1 ». Le tour de jeu affiche : « Blanc ».
- L’utilisateur tape « 2 », « g », « I », « p », « W » ou appuie directement sur la touche « Entrée » dans « Coup suivant ».

- Le message « Commande incorrecte » s'affiche.

3.2.1 Procédure initialisation

Fonctionnement :

Ce sous-programme permet d'initialiser l'échiquier dans sa configuration réglementaire de début de partie. Il utilise pour cela deux entiers « i » (ligne) et « j » (colonne) qui modifient les cases du tableau « plateau ». A noter que la première case de « plateau » est [0][0] et la dernière est [7][7]. « i » et « j » auront donc pour valeur initiale « 0 » et pour valeur finale « 7 ».

Il faut placer les caractères représentant les pièces aux bons endroits dans le tableau. En remontant ligne par ligne avec la variable « i », on s'arrête sur les cas suivants :

- à la ligne 1 (i = 0), selon le numéro de colonne « j », on positionne dans le « plateau » les caractères des pièces blanches selon les positions de départ établies en 2.3, en tenant compte que « j » varie de « 0 » à « 7 ».
- à la ligne 2 (i = 1), on place le caractère « P » (pion blanc) dans chaque colonne de « plateau », pour « j » variant de « 0 » à « 7 ».
- à la ligne 7 (i = 6), on place le caractère « i » (pion noir) dans chaque colonne de « plateau », pour « j » variant de « 0 » à « 7 ».
- à la ligne 8 (i = 7), selon le numéro de colonne « j », on positionne dans le « plateau » les caractères des pièces noires selon les positions de départ établies en 2.3, en tenant compte que « j » varie de « 0 » à « 7 ».
- dans tous les autres cas, les cases sont vidées : on y entre le caractère ' '.

Recette :

Scénario 1 :

- L'utilisateur tape 1 dans le menu principal.
- L'échiquier s'affiche avec les caractères symbolisant les pièces placées conformément aux positions de départ fixées en 2.3.

3.2.2 Procédure sauvegarde

Fonctionnement :

Cette procédure sauvegarde les valeurs des variables dans un fichier. Si "coupJeu" est égale "1", le message "Aucune partie à sauvegarder" s'affiche. Sinon, l'utilisateur doit rentrer le nom du fichier de sauvegarde. Le fichier s'ouvre en écriture, on y inscrit le tour de jeu

(blanc ou noir), le compteur de tour "coupJeu" et le tableau "plateau". La variable globale "a_sauvegarder" passe à 0, indiquant qu'il n'y a plus de donnée à sauvegarder. Le message "Partie sauvegardée" s'affiche.

Recette :

Scénario 1 :

- L'utilisateur tape « 1 » dans le menu principal.
- L'échiquier initialisé s'affiche avec les numérotations de ligne et de colonne bien positionnés. Le compteur de coup affiche « 1 ». Le tour de jeu affiche : « Blanc ».
- L'utilisateur tape « s » dans « Coup suivant ».
- Le message « Aucune partie à sauvegarder » s'affiche.

3.2.3 Fonction ifnoir

Fonctionnement :

Cette fonction vérifie si la pièce sélectionnée dans « coupSuivant » est noire ou pas. Elle retourne un booléen : « noir ».

Ce booléen est vrai si « coupJeu » est pair (c'est-à-dire si le reste de la division coupJeu/2 vaut 0). Sinon, il est faux.

Recette :

Scénario 1 :

- L'utilisateur tape « 1 » dans le menu principal.
- L'échiquier initialisé s'affiche avec les numérotations de ligne et de colonne bien positionnés. Le compteur de coup affiche « 1 ». Le tour de jeu affiche : « Blanc ».
- L'utilisateur tape « T », « C », « F », « D », « R » ou « P » dans « Coup suivant ».
- La procédure « mouvement » est appelée.
- L'échiquier s'affiche. Le compteur de coup affiche « 2 ». Le tour de jeu affiche : « Noir ».
- L'utilisateur tape « t », « k », « f », « d », « r » ou « i » dans « Coup suivant ».
- La procédure « mouvement » est appelée.
- L'échiquier s'affiche. Le compteur de coup affiche « 3 ». Le tour de jeu affiche : « Blanc ».

Scénario 2 :

- L'utilisateur tape « 1 » dans le menu principal.
- L'échiquier initialisé s'affiche avec les numérotations de ligne et de colonne bien positionnés. Le compteur de coup affiche « 1 ». Le tour de jeu affiche : « Blanc ».
- L'utilisateur tape « t », « k », « f », « d », « r » ou « i » dans « Coup suivant ».
- Le message « Vous ne pouvez pas bouger une pièce noire » s'affiche. L'utilisateur doit ressaisir « Coup suivant ».

- L'utilisateur tape « T », « C », « F », « D », « R » ou « P » dans « Coup suivant ».
- La procédure « mouvement » est appelée.
- L'échiquier s'affiche. Le compteur de coup affiche « 2 ». Le tour de jeu affiche : « Noir ».
- L'utilisateur tape « T », « C », « F », « D », « R » ou « P » dans « Coup suivant ».
- Le message : « Vous ne pouvez pas bouger une pièce blanche » s'affiche. L'utilisateur doit ressaisir « Coup suivant ».

3.2.4 Procédure mouvement

Fonctionnement :

Cette procédure permet de déplacer une pièce de sa case de départ à sa case d'arrivée. Elle appelle tout un panel de sous-programmes vérifiant la validité du mouvement.

Après que l'utilisateur ait sélectionné un type de pièce (exemple : « P »), il doit rentrer la case de départ de cette pièce afin d'indiquer précisément la pièce à déplacer. Il indique d'abord la colonne de départ (caractère) puis la ligne de départ (entier). Celles-ci se retrouvent dans les variables locales « colonneD » et « ligneD ». L'utilisateur va entrer un caractère et un saut de ligne dans « colonneD », il faut donc vider le buffer de la mémoire en utilisant une variable globale « bidon » prévue à cet effet.

Après avoir indiqué une case de départ, la fonction « verificationEntree » (cf. 3.2.5) qui retourne « 1 » si l'utilisateur a saisi une case valide. Si ce n'est pas le cas, le message « Erreur de saisie » apparaît.

Si la case est correcte, il faut vérifier la correspondance avec la variable « plateau ». Pour cela, la commande de l'utilisateur doit correspondre avec une case de la variable « plateau ». « colonneD » est converti en entier en appelant la fonction « conv_colonne_Chartolnt » (cf. 3.2.6). Cette conversion devient la variable locale « c1 ». Une variable locale « l1 » prend la valeur « ligneD-1 » car la première ligne de « plateau » est « 0 » et pas « 1 ». Ainsi, il est possible d'appeler la fonction « checkPiece » (cf. 3.2.7) qui vérifie si la pièce située dans la case départ est conforme avec celle saisie dans « coupSuivant ».

« checkPiece » renvoie un booléen qui est mis dans la variable locale « trouve ». Si « trouve » est faux, alors un message d'erreur de type « Il n'y a pas de P en a2 » s'affiche. Ledit message s'adapte aux entrées effectuées par l'utilisateur dans « coupSuivant » et « Case départ ». L'utilisateur est alors invité à ressaisir une case de départ.

Si « trouve » est vrai, le coup est valide et l'utilisateur est invité à entrer une case d'arrivée. Pour entrer dans cette première boucle, « trouve » doit être différent de « 1 ». On lui attribue donc la valeur « 0 » par défaut dans la déclaration.

Le processus de la case d'arrivée est identique à celle de la case de départ. Les valeurs entrées par l'utilisateur sont placées dans les variables locales « colonneA » et « ligneA ». Le buffer de la mémoire est vidé de la même manière pour pouvoir lire « colonneA » à chaque fois. La fonction « verificationEntree » est appelée pour vérifier la validité de la case d'arrivée. Si ce n'est pas la case, le message « Erreur de saisie » s'affiche.

La fonction « mouvementsPossibles » (cf. 3.2.9) vérifie si le déplacement demandé par l'utilisateur est autorisé. Si ce n'est pas le cas, le message « Mouvement impossible » s'affiche. Sinon, la procédure « departToArrivee » (cf. 3.2.10) met l'échiquier à jour avec les nouvelles positions et un message décrit le mouvement à l'utilisateur pour qu'il ait la confirmation que son coup est valide (exemple : « Déplacement de P de a2 en a3 »). Si la condition d'échec et mat ne se déclenche pas, on procède à l'incrémentation de la variable globale « coupJeu » qui augmente de 1 et « a_sauvegarder » prend « 1 », donnant la possibilité de sauvegarder lors du prochain coup.

Pion en bout de ligne

Dans le cas particulier où le mouvement conduit un pion noir (« i ») ou blanc (« P ») en dernière ligne, c'est-à-dire si la fonction finLigne_pion (cf. 3.2.11) est appelée, un message informe l'utilisateur qu'il a l'opportunité de changer sa pièce. Si c'est le tour des noirs (appel de la fonction « ifnoir »), il devra choisir entre une dame noire « d », un fou noir « f », un cavalier noir « k » ou une tour noire « t ». Sinon, c'est le tour des blancs et il devra choisir entre une dame blanche « D », un fou blanc « F », un cavalier blanc « C » ou une tour blanche « T ». Dans les deux cas, l'utilisateur ne sortira pas de la boucle tant qu'il n'aura pas donné à « coupSuivant » l'une des valeurs mentionnées ci-dessus. Quand il entre un caractère correct, celui-ci est mis dans « coupSuivant ». Il reste à actualiser le tableau « plateau ». Le caractère entré par l'utilisateur prend la place de « i » ou de « P » dans la case concernée.

Echec au roi

Si « ifnoir » est vraie et que le roi noir « r » est en échec (checkRoi()=2), le message « Attention, votre roi est en échec ! » s'affiche. La fonction "departToArrivee" et la variable "valeurInit" replacent alors les pièces dans leurs positions initiales. Le même mécanisme est utilisé pour les blancs (« ifnoir » faux, checkRoi() = 1).

Echec et mat

Dans le cas où les fonctions « checkRoi » et « checkmat » sont vraies toutes les deux. Le message « Echec et mat !!! » s'affiche. Un second message affiche le camp du vainqueur, correspondant à la variable « tour ». La variable « a_sauvegarder » passe à « 0 », indiquant qu'il n'y a plus de donnée à sauvegarder. « coupSuivant » prend ensuite « q » pour sortir de la boucle dans la procédure « nouvellePartie ».

Recette :

Scénario 1 :

- L'utilisateur tape « 1 » dans le menu principal.
- L'échiquier initialisé s'affiche avec les numérotations de ligne et de colonne bien positionnés. Le compteur de coup affiche « 1 ». Le tour de jeu affiche : « Blanc ».
- L'utilisateur tape « P » dans « Coup suivant ».
- L'utilisateur est invité à saisir la Case départ.
- L'utilisateur tape « t5 » dans « Case départ ».
- La fonction « verificationEntree » est appelée.

3.2.5 Fonction verificationEntree

Fonctionnement :

Cette fonction vérifie si l'utilisateur a saisi une case existante. Elle utilise les paramètres « colonne » (caractère) et « ligne » (entier). Elle retourne un booléen : « verif » qui est vrai si la commande est valide ou faux si la commande est incorrecte. Il est faux par défaut.

La colonne est vérifiée en premier. Au cas où l'utilisateur aurait entré un caractère majuscule (exemple : A2), on utilise la fonction tolower pour convertir le caractère en minuscule (rajouter la bibliothèque <ctype.h>). Ensuite, « verif » passe à vrai si le caractère correspond à une colonne valide (c'est-à-dire « a », « b », « c », « d », « e », « f », « g » et « h »). Sinon il reste faux et est retourné comme tel.

Si « verif » est vrai pour la colonne. La ligne est vérifiée à son tour. Pour les cas valides (c'est-à-dire « 1 », « 2 », « 3 », « 4 », « 5 », « 6 », « 7 » et « 8 »), « verif » reste vrai et est retourné comme tel. Pour tous les autres cas, « verif » passe à faux et est retourné comme tel.

Recette :

Scénario 1 :

- L'utilisateur tape « 1 » dans le menu principal.
- L'échiquier initialisé s'affiche avec les numérotations de ligne et de colonne bien positionnés. Le compteur de coup affiche « 1 ». Le tour de jeu affiche : « Blanc ».
- L'utilisateur tape « P » dans « Coup suivant ».
- L'utilisateur est invité à saisir la Case départ.

- L'utilisateur tape « t5 » dans « Case départ ».
- Le message « Erreur de saisie » apparaît. L'utilisateur doit ressaisir « Case départ ».
- L'utilisateur tape « a9 » dans « Case départ ».
- Le message « Erreur de saisie » apparaît. L'utilisateur doit ressaisir « Case départ ».
- L'utilisateur tape « b2 » dans « Case départ ».
- Les fonctions « conv_colonne_Chartolnt » et « checkPiece » sont appelées.

3.2.6 Fonction conv_colonne_Chartolnt

Fonctionnement :

Cette fonction transforme le caractère de la colonne indiquée dans « Case départ » en entier afin de faire la correspondance avec le tableau d'entiers « plateau ». Elle utilise le paramètre « paramc » pour copier la commande de l'utilisateur et retourne l'entier « c1 ».

Au cas où l'utilisateur aurait entré un caractère majuscule (exemple : A2), on utilise la fonction tolower pour convertir le caractère « paramc » en minuscule. Ensuite, selon « paramc », on attribue une valeur à « c1 » (cas « a », c1=0 ; cas « b » : c1=1 ; etc.). Si « paramc » n'est pas un caractère réglementaire pour une colonne (exemple : « m »), « c1 » vaut « -1 » afin qu'il ne puisse pas prendre place dans le tableau « plateau ».

3.2.7 Fonction checkPiece

Fonctionnement :

Cette fonction vérifie le caractère entré dans « coupSuivant » est le même que celui contenu dans la case de départ. Elle utilise les paramètres « paramc » et « paraml » qui copient les variables « c1 » et « l1 » de la procédure « mouvement ». Elle retourne un booléen : « trouve », faux par défaut, qui devient vrai si le bon caractère est trouvé dans le tableau « plateau ».

Il s'agit de vérifier si dans « plateau », la case de ligne « paraml » et de colonne « paramc » contient le même caractère que « coupSuivant ». Si c'est le cas, « trouve » passe à vrai et est retourné comme tel. Sinon, il est retourné comme faux.

Recette :

Scénario 1 :
- L'utilisateur tape « 1 » dans le menu principal.
- L'échiquier initialisé s'affiche avec les numérotations de ligne et de colonne bien positionnés. Le compteur de coup affiche « 1 ». Le tour de jeu affiche : « Blanc ».
- L'utilisateur tape « P » dans « Coup suivant ».
- L'utilisateur est invité à saisir la Case arrivée.

- L'utilisateur tape « b2 » dans « Case départ ».
- L'utilisateur est invité à saisir la Case arrivée.
- L'utilisateur tape « t5 » dans « Case arrivée ».
- Le message « Erreur de saisie » apparaît. L'utilisateur doit ressaisir « Case arrivée ».
- L'utilisateur tape « a9 » dans « Case arrivée ».
- Le message « Erreur de saisie » apparaît. L'utilisateur doit ressaisir « Case arrivée ».
- L'utilisateur tape « b5 » dans « Case arrivée ».
- La fonction « mouvementsPossibles » est appelée.

Scénario 2 :

- L'utilisateur tape « 1 » dans le menu principal.
- L'échiquier initialisé s'affiche avec les numérotations de ligne et de colonne bien positionnés. Le compteur de coup affiche « 1 ». Le tour de jeu affiche : « Blanc ».
- L'utilisateur tape « P » dans « Coup suivant ».
- L'utilisateur est invité à saisir la Case départ.
- L'utilisateur tape « d1 » dans « Case départ ».
- Le message « Il n'y a pas de P en d1 » s'affiche. L'utilisateur doit ressaisir « Case départ ».

3.2.8 Fonction verifoccupe

Fonctionnement :

Cette fonction vérifie si la « Case arrivée » sélectionnée par l'utilisateur est occupée par une autre pièce. Ceci est impératif pour gérer les mouvements des pièces, des pions en particulier. Elle utilise les paramètres entiers « paramla » et « paramca » qui copient les variables « l2 » et « c2 » de la procédure « mouvement ». Elle retourne un entier : « verif », égal à « 0 » par défaut.

Si la case du tableau « plateau » ayant pour ligne « paramla » et pour colonne « paramca » est vide, alors « verif » prend « 1 ». Sinon, selon que la case contienne une pièce blanche ou noire, « verif » prendra soit la valeur « 3 » pour une pièce blanche ou la valeur « 5 » pour une pièce noire. « verif » est alors retourné avec la valeur qu'il aura pris.

3.2.9 Fonction mouvementsPossibles

Fonctionnement :

Cette fonction vérifie la validité d'un mouvement entré par l'utilisateur. Elle utilise quatre paramètres représentant les lignes et colonnes de départ (« paramld », « paramcd ») et d'arrivée (« paramla », « paramca ») de la pièce bougée. Elle renvoie un booléen : « possible », faux par défaut. Les conditions pour mettre « possible » à vrai sont différentes selon le type de pièces jouées. Chaque cas est traité différemment.

Mouvements du pion : Dans le cas du pion blanc « P », il faut d'abord envisager la condition particulière où la ligne de départ est « 1 » dans le tableau « plateau ». Trois types de mouvements mettent « possible » à vrai :

1) Si « P » avance d'une ligne (paramla = paramld+1) sur une même colonne (paramca = paramcd) à condition que la fonction « verifoccupe » ciblant la case d'arrivée renvoie « 1 », signifiant que cette case est vide.

2) Si « P » avance d'une ligne sur une colonne différente (paramca = paramcd+1 OU paramca = paramcd-1) à condition que la fonction « verifoccupe » ciblant la case d'arrivée renvoie « 5 », signifiant que cette case est occupée par une pièce noire.

3) Si « P » avance de deux lignes (paramla = paramld+2) sur une même colonne. La fonction « verifoccupe » doit assurer que la case de transit (paramld+1, paramca) et la case d'arrivée sont vides. Elle doit donc renvoyer « 1 » les deux fois.

Si la ligne de départ est différente de « 1 », seuls les mouvements 1) et 2) cités ci-dessus mettent « possible » à vrai. A noter que la ligne de départ ne pourra jamais être « 7 » du fait de la procédure « mouvement ».

Le cas du pion noir « i » fonctionne de manière identique. Pour une ligne de départ « 6 » dans le tableau « plateau », trois types de mouvements mettent « possible » à vrai :

1) Si « i » avance d'une ligne (paramla = paramld-1) sur une même colonne (paramca = paramcd) à condition que la fonction « verifoccupe » ciblant la case d'arrivée renvoie « 1 », signifiant que cette case est vide.

2) Si « i » avance d'une ligne sur une colonne différente (paramca = paramcd+1 OU paramca = paramcd-1) à condition que la fonction « verifoccupe » ciblant la case d'arrivée renvoie « « », signifiant que cette case est occupée par une pièce blanche.

3) Si « i » avance de deux lignes (paramla = paramld-2) sur une même colonne. La fonction « verifoccupe » doit assurer que la case de transit (paramld-1, paramca) et la case d'arrivée sont vides. Elle doit donc renvoyer « 1 » les deux fois.

Si la ligne de départ est différente de « 6 », seuls les mouvements 1) et 2) cités ci-dessus mettent « possible » à vrai. A noter que la ligne de départ ne pourra jamais être « 0 » du fait de la procédure « mouvement ».

Mouvements du cavalier : Pour les deux cas du cavalier blanc « C » et du cavalier noir « k », « possible » prend la valeur renvoyée par la fonction « deplacementCavalier » (cf. 3.2.12).

Mouvements de la tour : Pour les deux cas de la tour blanche « T » et la tour noir "t", « possible » prend la valeur renvoyée par la fonction « deplacementHorVer » (cf. 3.2.13). On détecte également si les tours ont déjà été déplacées en utilisant les variables globales "cptT1", "cptT2", "cptt1" et "cptt2" qui augmentent de 0 à 1 quand un mouvement d'une de ces tours est effectué.

Mouvements du fou : Pour les deux cas du fou blanc « F » et du fou noir « f », « possible » prend la valeur renvoyée par la fonction « deplacementDiagonale » (cf. 3.2.14).

Mouvements de la dame : Pour les deux cas de la dame blanche « D » et de la dame noire « d », « possible » prend la valeur renvoyée par la fonction « deplacementHorVer » (cf. 3.2.12). Si « possible » reste faux, le booléen prend la valeur renvoyée par la fonction « deplacementDiagonale » (cf. 3.2.14). En effet, si un déplacement horizontal ou vertical est possible, il n'est pas nécessaire de vérifier si le déplacement diagonal l'est aussi, la dame ayant de toute manière au moins un mouvement autorisé.

Mouvements du roi : Pour les deux cas du roi blanc "R" et du roi noir "r", on vérifie les huit mouvements possibles, vers les cases adjacentes. Pour chaque cas, on appelle la fonction "verifoccupe". Si celle-ci renvoie "1" (case vide), "possible" devient vrai ; si elle renvoie "3" (case occupée par une pièce blanche), "possible" devient vrai si c'est "r" qui est déplacé ; si elle renvoie "5" (case occupée par une pièce noire", "possible" devient vrai si c'est "R" qui est déplacé.

Si "possible" est vrai, on met le compteur de "R" ou "r" à "1" en utilisant les variables globales "cptR" et "cptr" qui seront utiles pour le cas du roque.

3.2.10 Procédure departToArrivee

Fonctionnement :

Cette procédure actualise le tableau « plateau », et par extension l'affichage de l'échiquier, après un coup valide. Elle utilise les paramètres entiers « c1 », « l1 » (colonne et ligne de départ), « c2 » et « l2 » (colonne et ligne d'arrivée).

La case de départ est vidée dans « plateau » : on y place le caractère « espace ». La case d'arrivée de « plateau » prend le caractère indiqué par l'utilisateur dans « coupSuivant ».

Recette :

Scénario 1 :

- L'utilisateur tape « 1 » dans le menu principal.
- L'échiquier initialisé s'affiche avec les numérotations de ligne et de colonne bien positionnés. Le compteur de coup affiche « 1 ». Le tour de jeu affiche : « Blanc ».
- L'utilisateur tape « P » dans « Coup suivant ».
- L'utilisateur tape « b2 » dans « Case départ ».
- L'utilisateur tape « b5 » dans « Case arrivée ».
- Le message « Mouvement impossible » apparaît.
- L'utilisateur tape « P » dans « Coup suivant ».
- L'utilisateur tape « b2 » dans « Case départ ».
- L'utilisateur tape « b4 » dans « Case arrivée ».
- Le message « Déplacement de P de b2 en b4 » apparaît.
- L'échiquier apparaît avec la case b2 vide et la case b4 occupée par « P ». Le compteur de coup affiche « 2 ». Le tour de jeu affiche : « Noir ».

Scénario 2 :

- C'est le tour des Blancs, le roi blanc « R » est dans la case d7. La case d8 est vide.
- L'utilisateur tape « R » dans « coupSuivant ».
- L'utilisateur tape « d7 » dans « Case départ ».
- L'utilisateur tape «e5 » dans « Case arrivée ».
- Le message « Mouvement impossible » apparaît.
- L'utilisateur tape « R » dans « coupSuivant ».
- L'utilisateur tape « d7 » dans « Case départ ».
- L'utilisateur tape «d8 » dans « Case arrivée ».
- Le message « Déplacement de R de d7 en d8 » apparaît.
- L'échiquier apparaît avec la case d7 vide et la case d8 occupée par « R ». Le tour de jeu affiche : « Noir ».

3.2.11 Fonction finLigne_pion

Fonctionnement :

Quand elle est appelée, cette fonction retourne « 1 » si la ligne d'arrivée d'une pièce, indiquée par le paramètre « paramla », est l'une des deux lignes extrêmes de l'échiquier, c'est-à-dire « 0 » ou « 7 » dans « plateau ». Cette fonction sera surtout utile dans le cas où un pion atteint sa ligne maximale afin que l'utilisateur puisse le transformer en une pièce de son choix.

Le booléen « fin » est faux et est retourné si « paramla » est différent de « 7 » ou de « 0 ». Dans le cas contraire, la fonction retourne vrai.

Recette :

Scénario 1 :

- C’est le tour des Blancs, un pion blanc « P » est dans la case b7. La case b8 est vide.
- L’utilisateur tape « P » dans « coupSuivant ».
- L’utilisateur tape « b7 » dans « Case départ ».
- L’utilisateur tape « b8 » dans « Case arrivée ».
- Le message « Choisissez (reine (D), fou (F), cavalier (C) ou tour (T)) : » apparaît.
- L’utilisateur tape « D ».
- Le caractère « D » s’affiche en case b8. Le tour de jeu affiche : « Noir ».

Scénario 2 :

- C’est le tour des Blancs, un pion blanc « P » est dans la case e7. La case f8 est occupée par la pièce noire « f ».
- L’utilisateur tape « P » dans « coupSuivant ».
- L’utilisateur tape « e7 » dans « Case départ ».
- L’utilisateur tape « f8 » dans « Case arrivée ».
- Le message « Choisissez (reine (D), fou (F), cavalier (C) ou tour (T)) : » apparaît.
- L’utilisateur tape « C ».
- Le caractère « C » s’affiche en case f8. Le tour de jeu affiche : « Noir ».

Scénario 3 :

- C’est le tour des Blancs, un pion blanc « P » est dans la case h7. La case g8 est occupée par la pièce noire « k ».
- L’utilisateur tape « P » dans « coupSuivant ».
- L’utilisateur tape « h7 » dans « Case départ ».
- L’utilisateur tape « g8 » dans « Case arrivée ».
- Le message « Choisissez (reine (D), fou (F), cavalier (C) ou tour (T)) : » apparaît.
- L’utilisateur tape « T ».
- Le caractère « T » s’affiche en case g8. Le tour de jeu affiche : « Noir ».

Scénario 4 :

- C’est le tour des Noirs, un pion noir « i » est dans la case a2. La case a1 est vide.
- L’utilisateur tape « i » dans « coupSuivant ».
- L’utilisateur tape « a2 » dans « Case départ ».
- L’utilisateur tape « a1 » dans « Case arrivée ».
- Le message « Choisissez (reine (d), fou (f), cavalier (k) ou tour (t)) : » apparaît.
- L’utilisateur tape « f ».
- Le caractère « f » s’affiche en case a1. Le tour de jeu affiche : « Blanc ».

Scénario 5 :

- C’est le tour des Noirs, un pion noir « i » est dans la case f2. La case e1 est occupée par la pièce blanche « D ».
- L’utilisateur tape « i » dans « coupSuivant ».
- L’utilisateur tape « f2 » dans « Case départ ».
- L’utilisateur tape « e1 » dans « Case arrivée ».
- Le message « Choisissez (reine (d), fou (f), cavalier (k) ou tour (t)) : » apparaît.
- L’utilisateur tape « k ».
- Le caractère « k » s’affiche en case e1. Le tour de jeu affiche : « Blanc ».

Scénario 6 :

- C’est le tour des Noirs, un pion noir « i » est dans la case g2. La case h1 est occupée par la pièce blanche « T ».
- L’utilisateur tape « i » dans « coupSuivant ».

- L'utilisateur tape « g2 » dans « Case départ ».
- L'utilisateur tape « h1 » dans « Case arrivée ».
- Le message « Choisissez (reine (d), fou (f), cavalier (k) ou tour (t)) : » apparaît.
- L'utilisateur tape « t ».
- Le caractère « t » s'affiche en case h1. Le tour de jeu affiche : « Blanc ».

3.2.12 Fonction deplacementCavalier

Fonctionnement :

Cette fonction renvoie le booléen « possible » à vrai si le déplacement du cavalier est autorisé. Elle utilise les paramètres représentant la case de départ (« paramld » et paramcd ») et la case d'arrivée (« paramla » et « paramca »). « possible » est faux par défaut.

Pour chaque mouvement du cavalier défini en 2.4.5, la fonction « verifoccupe » regarde si la case d'arrivée est vide ou occupée par une pièce adversaire à celle rentrée dans « coupSuivant ». Si « verifoccupe » renvoie « 1 », « possible » devient vrai. Si elle renvoie « 3 » (case occupée par une pièce blanche), « possible » devient vrai si la fonction « ifnoir » est vraie. Si « verifoccupe » renvoie « 5 » (pièce occupée par une pièce noire), « possible » devient vrai si la fonction « ifnoir » est fausse.

Recette :

Scénario 1 :

- C'est le tour des Noirs, le cavalier noir « k » est dans la case e5. La case f7 est occupée par la pièce blanche « D ».
- L'utilisateur tape « k » dans « coupSuivant ».
- L'utilisateur tape « e5 » dans « Case départ ».
- L'utilisateur tape « e6 » dans « Case arrivée ».
- Le message « mouvement impossible » apparaît.
- L'utilisateur tape « k » dans « coupSuivant ».
- L'utilisateur tape « e5 » dans « Case départ ».
- L'utilisateur tape « f7 » dans « Case arrivée ».
- Le message « Déplacement de k de e5 en f7 » apparaît.
- Le caractère « k » s'affiche en case f7. Le tour de jeu affiche : « Blanc ».

3.2.13 Fonction deplacementHorVer

Fonctionnement :

Cette fonction vérifie la validité des mouvements horizontal et vertical, utilisés par la tour et la dame. Elle renvoie le booléen « possible » à vrai si le déplacement du cavalier est autorisé. Elle utilise les paramètres représentant la case de départ (« paramld » et paramcd ») et la case d'arrivée (« paramla » et « paramca »). « possible » est faux par défaut.

Pour le mouvement horizontal (paramld = paramla), la fonction vérifie chaque colonne traversée par la pièce jusqu'à la case d'arrivée (paramacd = paramca). Pour chaque nouvelle colonne, la fonction « verifoccupe » est appelée. Si « verifoccupe » renvoie « 1 », « possible » devient vrai. Si « verifoccupe » renvoie « 3 » (case occupée par une pièce blanche), « possible » devient vrai si la fonction « ifnoir » est vraie et si paramcd = paramca, sinon « possible » devient faux. Si « verifoccupe » renvoie « 5 » (pièce occupée par une pièce noire), « possible » devient vrai si la fonction « ifnoir » est fausse et si paramcd = paramca, sinon « possible » devient faux. Dans chaque cas, « paramcd » est incrémenté, permettant ainsi de passer à la case suivante ou de sortir de la boucle si la case d'arrivée est atteinte.

La fonction utilise le même mécanisme pour le mouvement vertical (paramcd = paramca). Pour les deux types de mouvement, la fonction applique le traitement approprié en fonction des valeurs de « paramcd » et « paramld », inférieures ou supérieures « paramca » et « paramla ».

Recette :

Scénario 1 :

- C'est le tour des Blancs, la dame blanche « D » est dans la case b1. Les cases c1, d1, e1et g1 sont vides. La case f1 est occupée par une pièce blanche.
- L'utilisateur tape « D » dans « coupSuivant ».
- L'utilisateur tape « b1 » dans « Case départ ».
- L'utilisateur tape « g1 » dans « Case arrivée ».
- Le message « mouvement impossible » apparaît.
- L'utilisateur tape « D » dans « coupSuivant ».
- L'utilisateur tape « b1 » dans « Case départ ».
- L'utilisateur tape « e1 » dans « Case arrivée ».
- Le message « Déplacement de D de b1 en e1 » apparaît.
- Le caractère « D » s'affiche en case e1. Le tour de jeu affiche : « Noir ».

3.2.14 Fonction deplacementDiagonale

Fonctionnement :

Cette fonction vérifie la validité des déplacements en diagonale effectués par le fou ou la dame. Elle renvoie le booléen « possible » à vrai si le déplacement du cavalier est autorisé. Elle utilise les paramètres représentant la case de départ (« paramld » et paramcd ») et la case d'arrivée (« paramla » et « paramca »). « possible » est faux par défaut.

Le mécanisme est quasiment le même que celui décrit pour le déplacement horizontal (cf. *supra*). La seule différence est que la valeur de « paramld » varie de +1 ou de -1 à chaque vérification de case. Elle varie à chaque fois que « verifoccupe » renvoie « 1 » (case vide). « paramld » varie toujours de +1 ou toujours de -1 selon la valeur de « paramla ».

Recette :

Scénario 1 :

- C’est le tour des Noirs, le fou noir « f » est dans la case f8. La case d6 est vide. La case e7 est occupée par une pièce blanche.
- L’utilisateur tape « f » dans « coupSuivant ».
- L’utilisateur tape « f8 » dans « Case départ ».
- L’utilisateur tape « d6 » dans « Case arrivée ».
- Le message « mouvement impossible » apparaît.
- L’utilisateur tape « f » dans « coupSuivant ».
- L’utilisateur tape « f8 » dans « Case départ ».
- L’utilisateur tape « e7 » dans « Case arrivée ».
- Le message « Déplacement de f de f8 en e7 » apparaît.
- Le caractère « f » s’affiche en case e7. Le tour de jeu affiche : « Blanc ».

3.2.15 Fonction checkRoi

Fonctionnement :

Cette fonction indique si l’un des deux rois de l’échiquier est en échec. Elle retourne la variable « checkRoi », mise à « 0 » par défaut.

Les variables locales « i » et « j » sont utilisées pour parcourir l’ensemble du tableau « plateau ». Si une case n’est pas vide, un traitement s’opère selon la pièce présente dans la case (type de pièce et couleur). Chaque cas analyse tous les mouvements possibles pour cette pièce de prendre une pièce adverse. S’il y a possibilité pour ladite pièce de prendre la case occupée par un roi adverse, « checkRoi » prend la valeur « 1 » si le roi blanc est en échec ou « 2 » si le roi noir est en échec.

Pour une pièce effectuant un mouvement horizontal, vertical ou diagonal, la variable « k » est utilisée pour parcourir les cases de transit, qui doivent être vides, jusqu'à la case cible.

Recette :

Scénario 1 :

- C’est le tour des Noirs. « R » est en d1, « t » est en d8, « f » est en d7, les cases d6, d5, d4, d3, d2 et e6 sont vides.
- L’utilisateur tape « f » dans « coupSuivant ».
- L’utilisateur tape « d7 » dans « Case départ ».
- L’utilisateur tape « e6 » dans « Case arrivée ».
- Le message « Echec au roi !!! » apparaît.
- Le caractère « f » s’affiche » en case e6. La case d7 est vide. Le tour de jeu affiche : « Blanc ».

3.2.16 Fonction haspiece

Fonctionnement :

Cette fonction vérifie si la pièce sélectionnée dans "coupSuivant" est présente sur l'échiquier. Les variables "i" et "j" sont utilisées pour vérifier si le caractère entré dans "coupSuivant" existe dans le tableau "plateau". Le booléen "hasPiece" devient vrai. "i" et "j" prennent ensuite la valeur "8" pour sortir de la boucle.

Recette :

Scénario 1 :

- C'est le tour des Blancs, il n'y a plus de « F » sur l'échiquier.
- L'utilisateur tape « F » dans « coupSuivant »
- Le message « La pièce n'existe plus sur le plateau » apparaît.

3.2.17 Procédure roq

Fonctionnement :

Cette procédure vérifie si les conditions sont requises pour effectuer un roque, demande à l'utilisateur s'il désire faire ce coup spécifique et l'exécute en cas de réponse affirmative.

Le roque peut s'appliquer dans quatre situations, c'est-à-dire pour les quatre tours présentes sur l'échiquier (blanches et noires). Si la fonction « ifnoir » est vraie, on vérifie que les cases séparant le roi noir de la tour noire concernée sont vides et que les variables globales « cptr » et « cptt1 » ou « cptt2 » sont nulles. Pour rappel, ces variables sont modifiées lorsque le roi ou les tours effectuent un mouvement. Le fait qu'elles soient nulles identique que ni le roi ni la tour n'ont encore été bougées.

Si les conditions sont réunies pour faire un roque, le message « Voulez-vous faire un roque (O/N) » s'affiche en boucle tant que l'utilisateur n'entre pas l'une des réponses attendues (lue dans la variable « reponse »). Le message s'adapte selon les roques possibles (à gauche ou à droite ou les deux). Une fonction permet de vider le buffer d'un éventuel saut de ligne et toupper transforme le caractère entré en majuscule au cas où l'utilisateur tape « o » ou « n ». Si la réponse est positive, les caractères sont positionnés dans les cases adéquates selon les règles établies en 2.4.9 et la variable globale « ifroq » devient vraie, indiquant que ce coup est joué dans la procédure « nouvellePartie ».

Le mécanisme est identique si la fonction « ifnoir » est fausse (tour de jeu Blanc). Cette fois, ce sont les variables « cptR », « cptT1 » et « cptT2 » qui sont utilisées.

Recette :

Scénario 1

- C’est le tour des Blancs. « R » est en e1, « T » est en h1, les cases f1 et g1 sont vides. « R » et « T » n’ont pas effectué de déplacement au cours de la partie.
- Le message « Voulez-vous faire un roque (O/N) : » apparaît.
- L’utilisateur tape « o » au clavier.
- L’échiquier s’affiche avec « T » en f1, « R » en g1, les cases e1 et h1 sont vides. Le tour de jeu affiche « Noir ».

3.2.18 Fonction checkmat

Fonctionnement :

Cette fonction définit s'il y a échec et mat ou non. Elle renvoie le booléen "checkmat" qui est vrai par défaut.

Les variables locales "i" et "j" parcourent toutes les cases du tableau "plateau". A chaque fois qu'une case est occupée, on vérifie si une pièce a des mouvements autorisés en appelant la fonction "mouvementsPossibles" puis, au cas où un roi serait en échec, si l'un des mouvements permet de ne plus avoir d'échec au roi en appelant la procédure "doubledeplacement" (cf. 3.2.20). Dans ce cas, "checkmat" passe à faux, sinon il est retourné comme vrai. Tous les cas de pièces possibles sont traités de cette façon, les pièces ayant des mouvements verticaux, horizontaux ou diagonaux utilisent en plus la variable "k".

3.2.19 Fonction PartieNull

Fonctionnement :

Cette procédure vérifie pour chaque pièce s'il y a des mouvements possibles. Si c'est le cas, le booléen local "null" est faux. Sinon, si seuls les mouvements du roi sont autorisés mais que cela le mène en position d'échec, "null" reste vrai et est retournée comme tel.

3.2.20 Procédure doubledeplacement

Fonctionnement :

Cette procédure vérifie si, quand un roi est en échec, un mouvement d'une pièce présente sur l'échiquier peut empêcher l'échec et mat. Elle utilise les paramètres de ligne départ, colonne départ, ligne arrivée, colonne arrivée et de la valeur retournée par "checkRoi" ("1" si le roi blanc est en échec, "2" si le roi noir est en échec).

La valeur d'origine de la case d'arrivée est tout d'abord sauvegardée dans une variable locale "valeurOrigine". Le mouvement théorique est ensuite effectué grâce à la procédure "departToArrivee". La fonction "checkRoi" est appelée. Si celle-ci retourne une valeur différente de celle contenue dans "paramcheckRoi", alors la variable globale ifcheckmat devient fausse, indiquant qu'il n'y a pas échec et mat.

Les cases modifiées par ce mouvement théorique sont ensuite remises dans leur état initial : "departToArrivee" replace "paramcd" et "paramld" dans leur case de départ puis la valeur sauvée dans "valeurOrigine" est réattribuée à la case d'arrivée du tableau "plateau".

3.2.21 Procédure doubledeplacementRoi

Fonctionnement :

Elle est identique à la procédure double déplacement à la différence qu'elle vérifie si, après un déplacement théorique du roi, celui-ci se retrouve en échec.

3.3 Charger une partie

Fonctionnement :

Cette procédure permet de lire une partie qui a été sauvegardée sur un fichier. L'utilisateur doit rentrer le nom du fichier à lire. Le fichier s'ouvre en lecture. Un contrôle vérifie si le fichier est contient des données, si ce n'est pas le cas, le message "Pas de données à lire" s'affiche.

Sinon, on lit d'abord le tour de jeu (blanc ou noir) puis le compteur de tour ("coupJeu"). Ensuite, jusqu'à la fin du fichier, on lit les données dans le tableau "plateau".

3.4 Procédure quitter

Fonctionnement :

Cette procédure permet de quitter une partie en cours. Elle propose à l'utilisateur de sauvegarder sa partie. Si celui-ci refuse, il y a partie nulle.

Si l'utilisateur n'a pas déjà sauvegardé sa partie via la fonction « sauvegarde » (cf. 3.2.2), le message « Voulez-vous sauvegarder la partie (o/n) ? » s'affiche en boucle tant que l'utilisateur n'entre pas « o » ou « n » au clavier. Un caractère est lu, il faut donc vider la mémoire du buffer contenant un saut de ligne. L'utilisateur peut entrer « O » ou « N », la

fonction tolower effectuant l'ajustement pour passer ces caractères en minuscules afin de sortir de la boucle.

Si la réponse est « o », la procédure « sauvegarde » est appelée, sinon le message « Partie nulle » s'affiche.

Recette :

Scénario 1 :
- Après plusieurs coups de jeu, l'utilisateur tape « s » dans « coupSuivant ».
- Le message « Voulez-vous sauvegarder la partie (o/n) ? » apparaît.
- L'utilisateur tape « O » au clavier.
- Le message « Partie sauvegardée » apparaît.
- Le menu apparaît.

Scénario 2 :
- Après plusieurs coups de jeu, l'utilisateur tape « s » dans « coupSuivant ».
- Le message « Voulez-vous sauvegarder la partie (o/n) ? » apparaît.
- L'utilisateur tape « n » au clavier.
- Le message « Partie nulle » apparaît.
- Le menu apparaît.

IV. Difficultés rencontrées

4.1 Nommage des pièces

Une contrainte importante était de faire en sorte que l'utilisateur puisse reconnaître facilement une pièce sur l'échiquier et surtout le camp d'appartenance de chaque pièce. Mettre les pièces blanches en majuscules et les noires en minuscules était une première étape. Toutefois, les caractères « C » et « c » d'une part ainsi que « P » et « p » d'autre part se ressemblent trop, ce qui pouvait prêter à confusion.

Le cavalier noir a donc été converti en « k » (*knight*) et le pion noir en « i ». Cette solution présentait le défaut d'introduire des lettres différentes pour ces deux types de pièces mais il a été jugé que l'utilisateur s'habituerait rapidement à cette petite gymnastique.

4.2 Fonction pour vider le buffer

La mécanique de jeu et de navigation conduit régulièrement l'utilisateur à entrer un caractère et un saut de ligne dans la mémoire. Il faut donc vider le buffer pour pouvoir lire un autre caractère ultérieurement. L'utilisation de la fonction fflush(stdin) s'est révélé aussi

inutile qu'embêtante pour le bon fonctionnement du programme. La fonction while((bidon=getchar()) != '\n') lui a donc été préférée.

4.3 Règles non appliquées

Plusieurs règles du jeu d'échec non essentielles au déroulement global d'une partie n'ont pas été intégrées au programme par manque de temps et du fait de leur trop grande complexité. Ainsi, la prise en passant du pion a été abandonnée après quelques essais. De même, différentes conditions pour déclencher la partie nulle ont été délaissées :

- S'il n'y a plus possibilité pour aucun des camps de mater l'adversaire.
- Si un positionnement identique des pièces est intervenu trois fois sur le plateau.
- Si chaque camp a joué 50 coups consécutifs sans mouvement de pion ni aucune prise de pièce.

Enfin, la règle indiquant qu'un roque est impossible si une pièce menace l'une des cases que le roi doit traverser avant d'atteindre sa position finale n'a pas été prise en compte, le programme faisant en sorte que le roi se déplace directement dans sa case d'arrivée.

4.4 Textes sans accents

Lorsque les textes du programme contenaient des mots avec des accents, des problèmes de d'encodage survenaient. Il a donc été choisi de supprimer tous les accents pour simplifier la lecture.

4.5 Lire les espaces dans le fichier

Dans la procédure "chargement", la restitution fidèle de l'échiquier de la partie sauvegardée nécessitait de pouvoir lire les espaces dans le fichier de sauvegarde, ce qui n'a jamais pu être possible de façon satisfaisante.

Pour pallier à ce problème, la procédure "sauvegarde" remplace tous les espaces par un "9" dans le fichier de sauvegarde. Ensuite, dans la procédure "chargement", tous les "9" sont remplacés par des espaces au moment de la lecture du fichier.

4.6 Partie nulle

La partie nulle n'a pas pu être testée par manque de temps.

V. Annexes

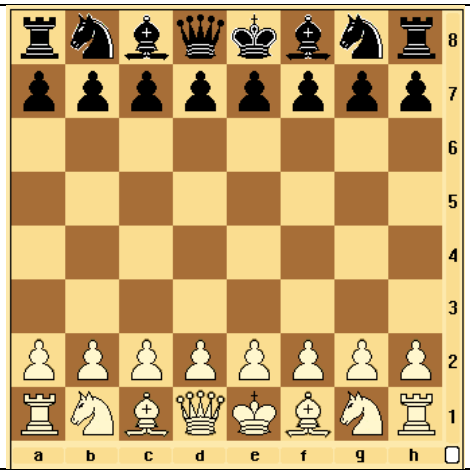
5.1 Les exigences

5.1.1 Plateau

N°	Code	Description	Remarques
1	PLA	Le plateau de jeu comprend 64 cases (8 cases sur 8), numérotées de 1 à 8 en ordonnée et de a à h en abscisse.	Faire un tableau à deux dimensions.
2	PLA	Les 64 cases et les lignes de numérotation doivent apparaître en permanence à l'écran.	
3	PLA	Les lettres de numérotation en abscisse doivent être en minuscules afin qu'elles ne soient pas confondues avec les pièces.	
4	PLA	Dans un souci de lisibilité, un espace devra être introduit entre le bord gauche de l'écran et la numérotation en ordonnée.	
5	PLA	Le plateau doit être lisible. L'utilisateur doit pouvoir reconnaître une pièce au premier coup d'œil.	
1	CAS	Chaque case est unique. Elle est nommée d'après son abscisse et son ordonnée sous la forme suivante : a1, b1, c1, […], h8.	
2	CAS	Une case est un carré ou un rectangle s'apparentant à un carré.	Elle aura la forme suivante : + + + + + + + + + + 5 + en horizontal séparés les uns des autres par un espace. 5 + en vertical
3	CAS	Une case doit pouvoir contenir lisiblement une lettre symbolisant une pièce.	Bien démarquer la lettre de la case (espaces, taille, gras, couleur) : + + + + + + P + + + + + +
4	CAS	Aucun espace ne séparera une numérotation d'une case.	Exemple : + + + + 1 + + + + + + a

5.1.2 Positions de départ

N°	Code	Description	Remarques
1	PIE	Le jeu est disputé entre deux camps : les blancs et les noirs.	
2	PIE	Chaque camp possède seize pièces au début de la partie : 8 pions : P 2 tours : T 2 cavaliers : C 2 fous : F 1 reine : Q ou D 1 roi : K ou R	
3	PIE	L'utilisateur doit pouvoir reconnaître sans hésitation à quel camp appartient une pièce.	Mettre les noirs en gras ?
1	DEB	Le placement de départ doit toujours être le même en début de partie.	Voir schéma ci-dessous.
2	DEB	Les blancs doivent être positionnés en bas du plateau en début de partie.	Voir schéma ci-dessous.
3	DEB	Les noirs doivent être positionnés en haut du plateau en début de partie.	Voir schéma ci-dessous.
4	DEB	La reine doit être positionnée à gauche du roi en début de partie.	Voir schéma ci-dessous.
5	DEB	Les positions des pièces en début de partie sont les suivantes : Tours noires en a8 et h8 Cavaliers noirs en b8 et g8 Fous noirs en c8 et f8 Reine noire en d8 Roi noir en e8 Pions noirs en a7, b7, c7, d7, e7, f7, g7 et h7 Tours blanches en a1 et h1 Cavaliers blancs en b1 et g1 Fous blancs en c1 et f1 Reine blanche en d1 Roi blanc en e1 Pions blancs en a2, b2, c2, d2, e2, f2, g2 et h2	Voir schéma ci-dessous.



5.1.3 Mouvements

Général (GEN), Manger une pièce (MAN), pion (PION), tour (TOUR), cavalier (CAV), fou (FOU), reine (DAM), roi (ROI) et roque (ROQ).

N°	Code	Description	Remarques
1	GEN	Les blancs jouent toujours en premier.	
2	GEN	Un camp ne peut jamais jouer deux fois d'affilé.	
3	GEN	Il est impossible d'annuler un mouvement.	Principe de pièce posée = pièce jouée
4	GEN	L'utilisateur ne peut bouger qu'une seule pièce à la fois.	Sauf pour le roque.
5	GEN	Un mouvement s'écrira sous la forme : « Déplacer x pièce en case 1 en case 2 »	Faire des variables pour les lignes et colonnes. Pour les lignes, faire un char et utiliser le code ascii : a = 65, a+1 = 66 = b, etc.
1	MAN	Une pièce ne peut pas prendre la case d'une autre pièce de sa couleur.	Détection couleur
2	MAN	Une pièce peut prendre la case d'une pièce de la couleur adverse.	Détection couleur
3	MAN	Une pièce peut avancer (selon ses mouvements autorisés) jusqu'à ce qu'elle rencontre une case occupée.	Détection vide
4	MAN	Une pièce arrête son mouvement lorsqu'elle rencontre un obstacle.	En prenant la case dudit obstacle ou en s'arrêtant une case avant.
1	PION	Un pion peut se déplacer d'une case inoccupée immédiatement devant lui sur la même colonne.	Ligne + 1 (blanc), ligne – 1 (noir)

2	PION	Un pion ne peut jamais reculer.	
3	PION	Si le pion est ligne 2 (blanc) ou ligne 7 (noir), il peut se déplacer d'une ou deux cases inoccupées immédiatement devant lui sur la même colonne.	Ligne+1 ou ligne+2 (blanc) Ligne-1 ou ligne-2 (noir)
4	PION	Un pion peut se déplacer vers une case occupée par une pièce adverse située devant lui en diagonale sur une colonne adjacente.	Ligne+1 (blanc), ligne-1 (noir) Colonne +1 ou -1.
5	PION	Si un pion a avancé de deux cases d'un seul coup à partir de sa case initiale, un pion adverse peut prendre ce pion comme s'il n'avait avancé que d'une case. Ce coup doit s'effectuer immédiatement après l'avancée de deux cases.	« Prise en passant ». Règle secondaire.
6	PION	Si un pion atteint sa ligne la plus éloignée de sa position de départ (ligne 8 pour les blancs, ligne 1 pour les noirs), l'utilisateur doit le remplacer au choix par une reine, un fou, un cavalier ou une tour.	Afficher le message : « Choisissez : reine (Q), fou (F), cavalier (C) ou tour (T) : »
1	TOUR	Une tour peut se déplacer sur toutes les cases de la ligne ou colonne sur laquelle elle se trouve jusqu'à ce qu'elle rencontre un obstacle (pièce ou bord du plateau).	Ligne +/- 1 à 7 ou Colonne +/- 1 à 7
2	TOUR	Cf. Roque	
1	CAV	Un cavalier ne peut pas se déplacer sur une case de la même ligne, colonne ou diagonale que la case qu'il occupe.	
2	CAV	Un cavalier se déplace de deux cases (en ligne ou en colonne) puis d'une case (en ligne ou en colonne).	Colonne +/- 2 ou +/- 1 Ligne +/- 2 ou +/- 1
1	FOU	Le fou peut se déplacer sur toutes les cases d'une diagonale sur laquelle il se trouve jusqu'à ce qu'il rencontre un obstacle (pièce ou bord du plateau).	Ligne +/- 1 à 7 et Colonne +/- 1 à 7
1	DAM	La reine peut se déplacer sur toutes les cases d'une ligne, colonne ou diagonale sur laquelle elle se trouve jusqu'à ce qu'elle rencontre un obstacle (pièce ou bord du plateau).	Cumul des mouvements de la tour et du fou.
1	ROI	Le roi ne peut pas se déplacer sur une case menacée par un mouvement d'une pièce adverse.	
2	ROI	Le roi peut se déplacer sur une case adjacente à la sienne (ligne, colonne ou diagonale).	Ligne (-1, 0 ou +1) Colonne (-1, 0 ou +1)
3	ROI	Cf. Roque	
1	ROQ	Le roque consiste en un mouvement du roi et d'une tour de la même couleur.	
2	ROQ	Le roque est le seul mouvement de jeu autorisant de déplacer deux pièces.	

3	ROQ	Un camp ne peut effectuer qu'un seul roque dans une partie.	
4	ROQ	Le roque n'est plus possible si le roi ou la tour concernée ont été déplacés lors de la partie en cours.	
5	ROQ	Lors du roque, le roi se déplace de deux cases vers la tour à partir de sa case initiale.	Petit roque : colonne+2 Grand roque : colonne-2
6	ROQ	La tour se déplace alors sur la dernière case traversée par le roi.	Petit roque : colonne-2 Grand roque : colonne+3
7	ROQ	Le roque est impossible si l'une des cases que le roi doit traverser est occupée ou menacée par une pièce adverse.	
8	ROQ	Le roque est impossible si le roi est en échec.	
9	ROQ	Le roque est impossible si la case que doit occuper le roi est menacée par une pièce adverse.	

5.1.4 Echecs

ECHR (échec au roi), ECHM (échec et mat), PAT (partie nulle)

N°	Code	Description	Remarques
1	ECHR	Si le roi est menacé par le mouvement d'une pièce adverse, il est en échec.	Afficher le message : « Echec au roi » Après mouvement d'une pièce, il faut détecter si une pièce menace le roi adverse pour son mouvement suivant.
2	ECHR	Si un roi est en échec, l'utilisateur a l'obligation de le mettre à l'abri des mouvements de l'adversaire.	Soit en bougeant le roi dans une position non menacée, soit en éliminant la menace (prise de pièce) soit en protégeant le roi par pièce interposée.
3	ECHR	Si le roi est en échec, aucun mouvement ne protégeant pas le roi n'est possible.	Vérifier si le roi est toujours en échec après mouvement. Si oui => message « mouvement non autorisé ».
1	ECHM	Il y a échec et mat si un roi de l'un des camps ne peut plus être à l'abri d'un mouvement du camp	

		adverse.	
2	ECHM	Le roi ne peut plus se déplacer.	
3	ECHM	Aucune pièce ne peut protéger le roi.	
4	ECHM	La partie s'arrête lorsqu'il y a échec et mat. Le camp du roi en échec a perdu.	Message : « Les [blancs/noirs] ont gagné »
1	PAT	Il y a partie nulle lorsque l'un des camps ne peut plus effectuer de mouvement alors que son roi n'est pas en échec.	
2	PAT	Il y a partie nulle lorsqu'il n'est plus possible ni pour un camp ni pour l'autre de mater le roi adverse.	
3	PAT	Il y a partie nulle si un positionnement identique des pièces est intervenu au moins trois fois sur le plateau.	secondaire
4	PAT	Il y a partie nulle si chaque camp a joué 50 coups consécutifs sans aucune prise de pièce.	secondaire
5	PAT	L'utilisateur doit pouvoir déclarer que la partie est nulle selon son bon-vouloir.	Option quitter

5.1.5 Options

N°	Code	Description	Remarques
1	OPT	Prévoir la consultation d'un guide des règles du jeu.	
2	OPT	Donner la possibilité à l'utilisateur de quitter la partie en cours.	Cf. PAT 5
3	OPT	Donner la possibilité à l'utilisateur de sauvegarder une partie.	
4	OPT	Donner la possibilité à l'utilisateur de charger une partie.	