RAPPORT DE PROJET

JEU DE DAME

Encadreur: Dr. Fouzi Mekhaldi

Jurys: Dr. Fouzi Mekhaldi & Dr. Kamel Ahsene Djaballah

TABLE DES MATIÈRES

INTRODUCTION:

Le jeu de dames, apprécié dans de nombreux pays, est un jeu de stratégie où la complexité réside dans l'anticipation des coups et la meilleure capture des pions adverses.

Ce projet vise à programmer en assembleur **8086** une version simplifiée de ce jeu, avec un damier de **10x10** utilisant uniquement 50 cases noires.

Les fonctionnalités incluent l'initialisation du damier, l'affichage de l'état du jeu, et la gestion des déplacements des pions, y compris la capture des pions adverses.

Ce rapport décrit la structure de données, les choix d'implémentation, les procédure utiliser, et propose un programme du jeu complet.

CHOIX D'IMPLÉMENTATION:

DÉVELOPPEMENT INITIAL EN LANGAGE C :

Avant de construire le code en assembleur **8086**, nous avons écrit toute la logique du programme en language **C**. Cette étape a facilité le développement et le débogage des algorithmes complexes. Tout le programme en language **C** est écrit par: **HOCINE MONCEF** & **HACHEMI YACINE**

CONVERSION EN ASSEMBLEUR :

HACHEMI YACINE : MAIN, DEPLACER, GETSQUARENUMBER, GETCOLUMN

HOCINE MONCEF : GENERATEAIMOVE, MUSTCAPTUREAFTERCAPTURE, VERIFQUEENS, AFFICHERDAMIER, PRINTSQUARE, INITIALISERDAMIER

ALOUACHE MANEL : DEPLACER, MUSTCAPTURE, GETSQUARECOLOR

CHETOUH AMIRA : VERIF, GETSQUARESTATE, GETROW

AMIROUCHE BADREDDINE :

DESCRIPTION DES FONCTIONS :

GETROW:

Cette procédure prend un numéro de case **N** comme entrée et retourne le numéro de la rangée correspondante (**de 1 à 10**) avec la formule (**N** - **1**) / **5**) + **1**. Elle vérifie également si **N** est dans la plage valide [**1, 50**].

```
Entrez un numero N entre 1 et 50: 9
La ligne est: 2
```

```
Entrez un numero N entre 1 et 50: 0
Erreur: N doit etre entre 1 et 50.
```

```
Entrez un numero N entre 1 et 50: 52
Erreur: N doit etre entre 1 et 50.
```

GETCOLUMN:

Cette procédure prend un numéro de case **N** comme entrée et retourne le numéro de la colonne correspondante (**de 1 à 10**). Elle utilise une logique spéciale pour déterminer la colonne en fonction de la parité de la rangée.

```
Entrez un numero N entre 1 et 50: 9
La colonne est: 7
```

GETSQUARENUMBER:

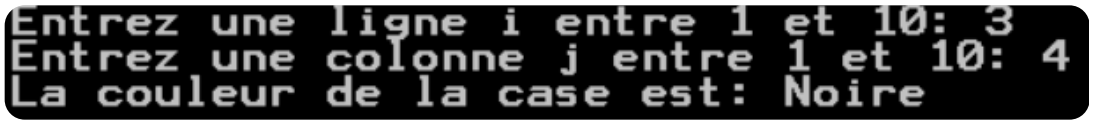
Cette procédure prend les coordonnées d'une case (**ligne i et colonne j**) comme entrée et retourne le numéro correspondant de la case (**de 1 à 50**) sur le damier avec la formule (**i** - **1**) * **5** + (**j** + **1**) / **2**. Elle gère également les cas d'erreur lorsque les coordonnées sont en dehors de la plage valide.

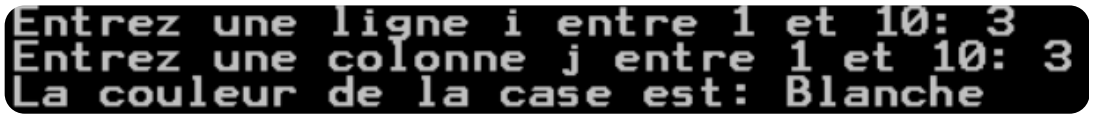
```
Entrez une ligne i entre 1 et 10: 3
Entrez une colonne j entre 1 et 10: 3
Erreur car C'est une Case blanche
```

```
Entrez une ligne i entre 1 et 10: 3
Entrez une colonne j entre 1 et 10: 4
Le numero du carre est: 12
```

GETSQUARECOLOR:

Cette procédure détermine la couleur d'une case du damier en utilisant les coordonnées fournies. Elle appelle **“getSquareNumber”** pour obtenir le numéro de la case, vérifie si la case est dans les limites valides, et définit la couleur comme **noire** ou **blanche** en conséquence.

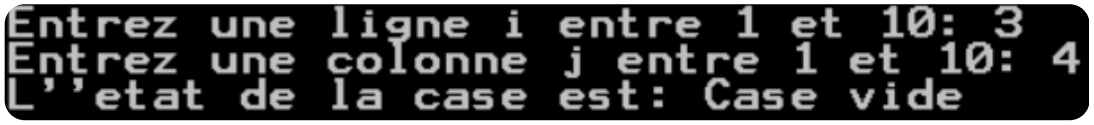




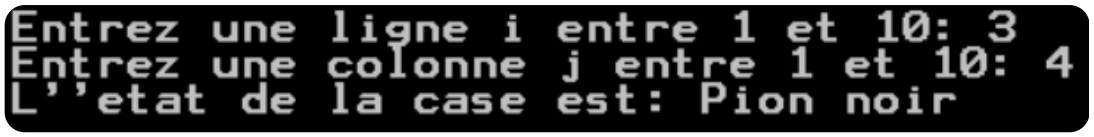
GETSQUARESTATE:

Cette procédure identifie l'état d'une case du damier à partir de ses coordonnées. Après avoir obtenu le numéro de la case avec **“getSquareNumber”**, elle consulte le tableau **board** pour déterminer si la case est **vide**, contient un **pion blanc**, un **pion noir**, une **dame blanche**, ou une **dame noire**, puis stocke le résultat.

AVANT:



APRES:



INITIALISERDAMIER:

Cette procédure initialise le damier en plaçant les pions **blancs** et **noirs** à leurs positions de **départ**. Elle parcourt le tableau **board** et assigne les valeurs appropriées pour les cases en fonction de leur position, définissant les **pions noirs** pour les **20** premières cases, les **pions blancs** pour les **20** dernières, et les cases **vides** entre les deux groupes.

AFFICHAGE:

ce fait a l'aide de deux procédures:

PRINTSQUARE:

Cette procédure affiche le contenu d'une case du damier en fonction de ses coordonnées. Elle appelle **“getSquareState”** et **“getSquareColor”** pour obtenir **l'état** et **la couleur** de la case, puis imprime le symbole correspondant (**pion blanc**, **pion noir**, **dame blanche**, **dame noire**, **case vide** avec un **“.”** ou **blanche** avec un vide **“ ”**).

AFFICHERDAMIER:

Cette procédure affiche l'ensemble du damier en itérant sur chaque case. Pour chaque case, elle appelle **“printSquare”** pour afficher son contenu, et ajoute des sauts de ligne pour séparer les rangées. Enfin, elle imprime les numéros de colonnes et de rangées pour une meilleure lisibilité.

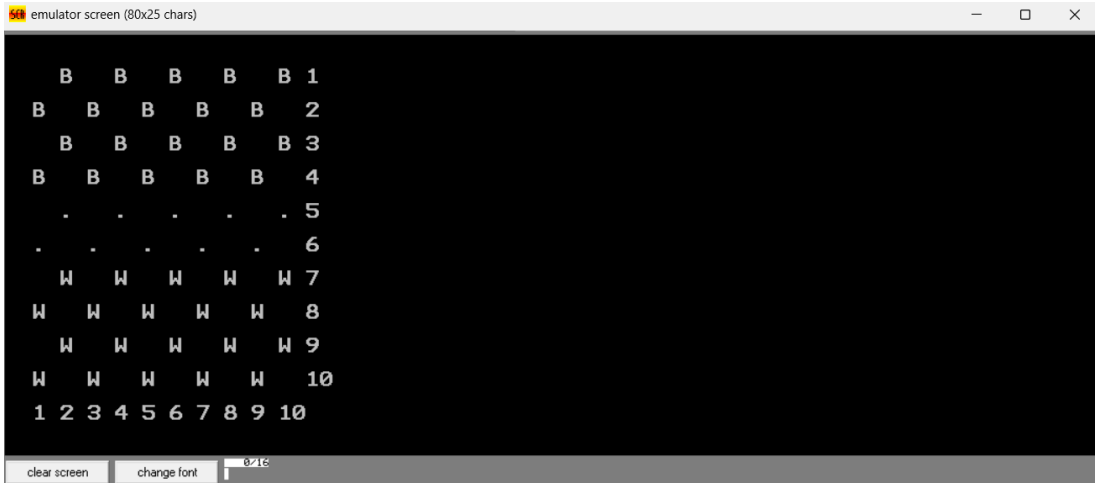


Figure -AFFICHAGE-

VERIFCATION:

ce fait a l'aide de 4 procédures:

MUSTCAPTURE:

Cette procédure vérifie si un joueur **peut capturer** une pièce **adverse** sur le plateau. Elle parcourt chaque case du board pour vérifier les **mouvements possibles** des **pions** et des **dames**. Si une capture est possible, la procédure **retourne 1**, sinon elle **retourne 0**.

MUSTCAPTUREAFTERCAPTUREE:

Cette procédure détermine si une **pièce** du **joueur peut effectuer** une **capture supplémentaire après une première capture**. Elle vérifie les mouvements **possibles** des **pions** et des **dames à partir** des **coordonnées fournies**. Si une capture supplémentaire est possible, la procédure **retourne 1**, sinon elle **retourne 0**.

VERIFQUEENS:

Cette procédure vérifie si une **dame peut capturer** une **pièce adverse** en fonction des **mouvements spécifiés** dans un **tableau de test**. Elle recherche les **coordonnées** de **départ** et **d'arrivée** dans le **tableau** pour valider la capture. Si la capture est **possible**,et les **coordonnées fournies** sont **fausse** un **message d'erreur** est **affiché** et la procédure **retourne 1**, sinon elle **retourne 0**.

VERIF:

Cette procédure vérifie la **validité** d'un mouvement de **pion** ou de **dame** en **tenant compte** des **captures** possibles. Elle compare les **coordonnées** de **départ** et **d'arrivée** avec les **captures possibles** enregistrées. Si le mouvement est **valide**, la procédure **retourne 1**, sinon elle **retourne 0**.

DEPLACER:

Cette procédure **effectue** un **déplacement** de pièce sur le plateau, en **vérifiant** d'abord la **validité** du mouvement avec la proc **“verif”**. Si une **capture** est **possible**, la **pièce capturée** est **supprimée**, et la procédure vérifie si une **promotion** à **dame** est nécessaire. Après le mouvement, elle met à jour le **score** et change le **joueur actif**.

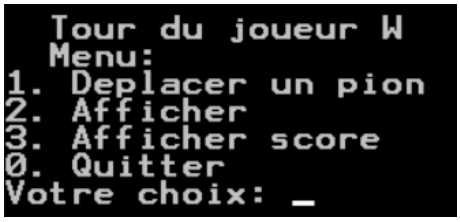
/*photo*/

GENERATEAIMOVE:

Cette procédure **génère** et **effectue** un mouvement pour **l'IA**, en commençant par **vérifier** si une **capture** est **obligatoire**. Si une capture n'est pas possible, elle **cherche** un **mouvement valide** pour une **pièce**, en **essayant** chaque **pièce** jusqu'à **trouver** un mouvement **valide**. Elle **appelle** ensuite **“deplacer”** pour **effectuer** le **mouvement** sélectionné.

MAIN:

Cette procédure **initialise** le jeu, **affiche** le plateau et **gère** le déroulement du jeu. Elle **alterne** entre les **tours** des **joueurs humains** et de **l'IA**, permettant aux **joueurs** de **choisir** leurs **actions** via un **menu**. Elle **vérifie** également les **conditions de fin** de jeu, telles que les **scores** et **déclare** le **gagnant** ou un **match nul** le cas échéant.



/*photo*/

LES DIFFICULTÉS:

CONCLUSION:

En conclusion, ce projet nous a offert une expérience enrichissante et formatrice, nous préparant à aborder des défis plus complexes dans notre parcours en informatique. Nous avons non seulement renforcé nos compétences techniques, mais aussi appris à collaborer efficacement en équipe pour atteindre nos objectifs communs.

RÉFÉRENCES:

Claude.ai

Chatgpt.com

Bard.com

Documentation-emu8086

Dev.to

Yassinebridi.github.io

Github/assembleur8086.com

Github/assembly8086.com

YouTube.com/gmae-in-video-mode

YouTube.com/assembly-projects-chanel

YouTube.com/assembly_full-tuttorial-playlist

YouTube.com/assembly-basic_tuttorial-playlist

MEMBRE DE GROUPE:

HACHEMI MOHAMED YACINE: **222231369919**

HOCINE MED ABDEL MONCEF: **222231502109**

ALOUACHE MANEL: **222231362904**

CHETOUH AMIRA NARIMENE: **222231484907**

AMIROUCHE BADREDDINE: **222231520117**