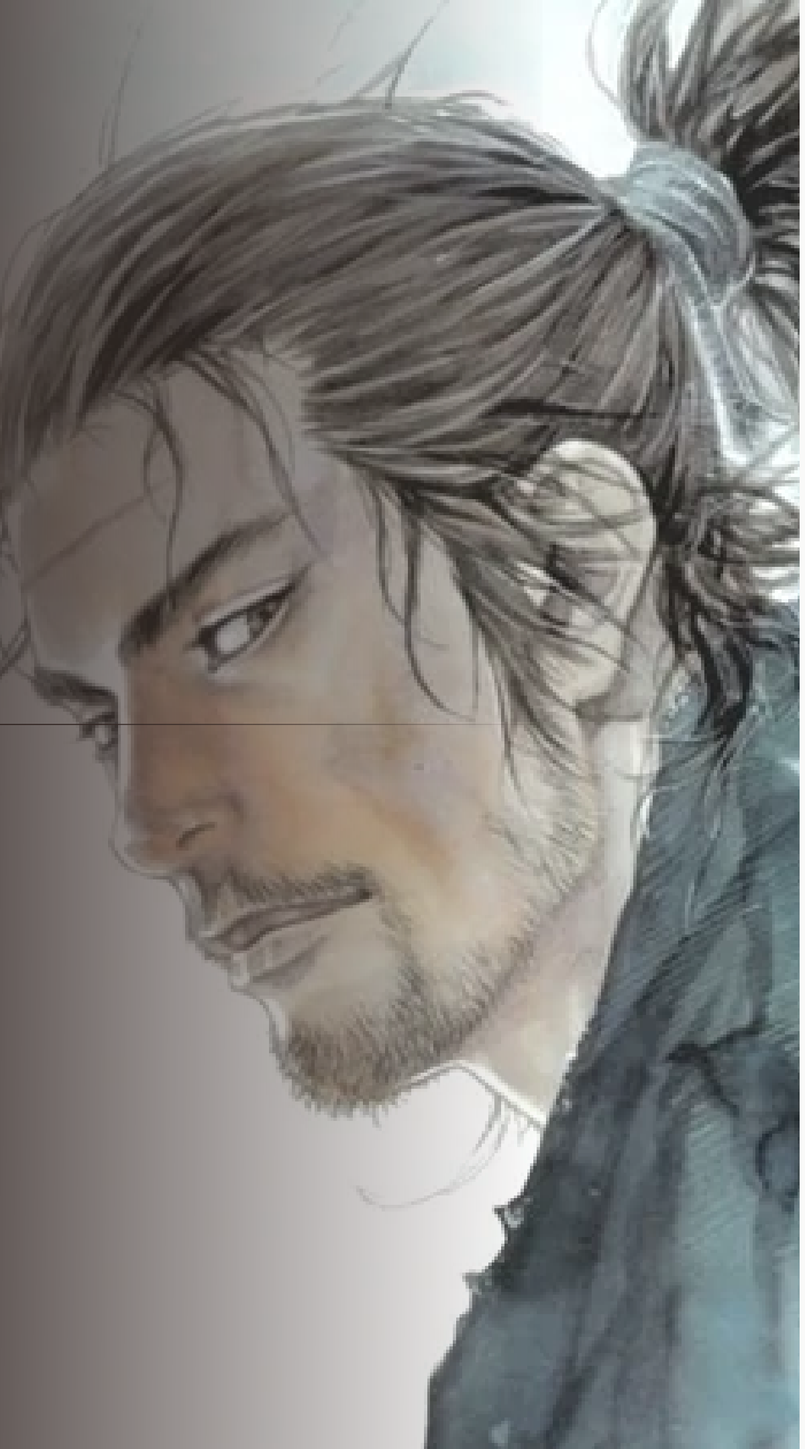


JEU MUSASHI

Minimax ___ Alphabeta

Réalisé par:

- Ilham HAFID
- Mostafa SISSI

Encadré par:

Mme ADDOU

PLAN

- Présentation du jeu
- Console et Graphique
- Conception du jeu
- Minimax VS AlphaBeta
- Démo
- Conclusion

Présentation du jeu

Jeu Musashi

- Jeu combinatoire entre deux adversaires
- les policiers (16 pions) : entourer le musashi
- le 'Musashi' (un seul pion) : tuer les policiers

    - déplacement vers les cases vides voisines suivant un chemin prévu
    - FIN :
        - Entourer le musashi ;
        - Tuer au minimum 12 policiers ;

Console Graphique

Console :

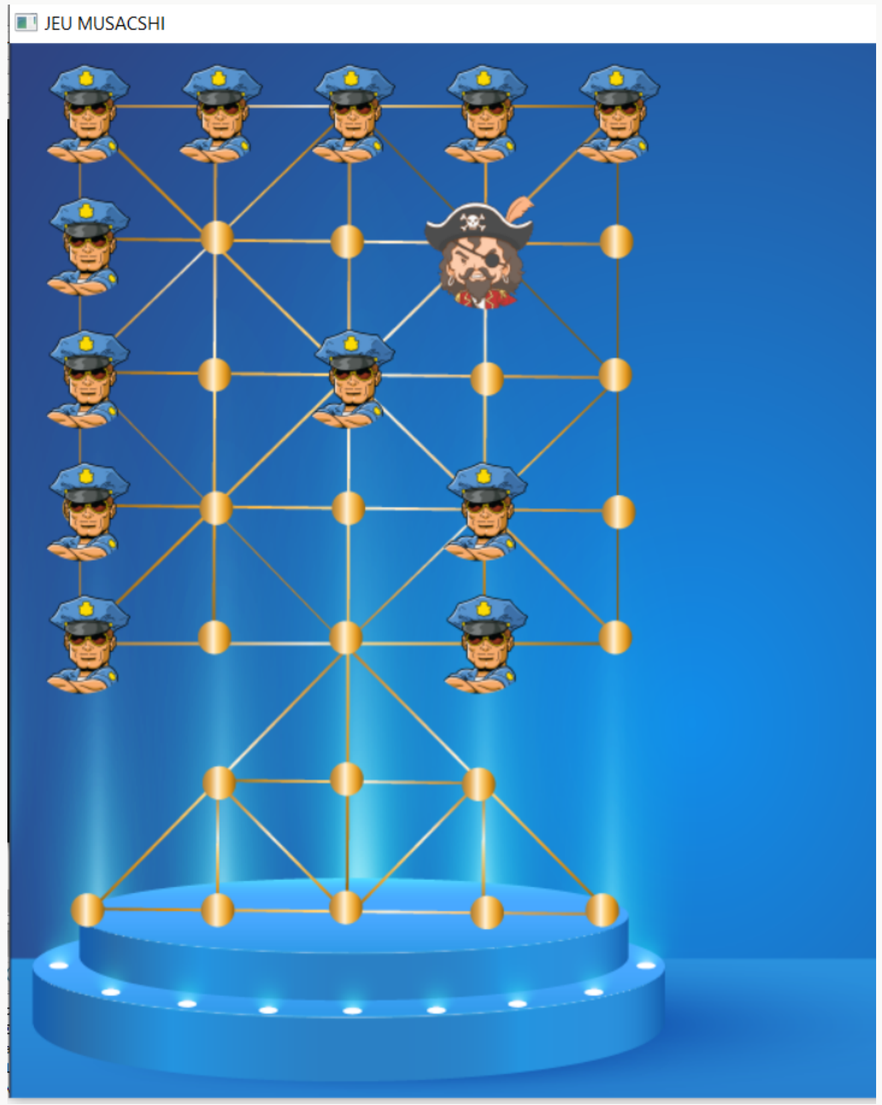
Pour le déroulement du jeu en console, le joueur saisi la direction souhaitée et qui indexée de 0 jusqu'à 7, suit par le role de la machine.
Pour chaque tour le plateau sera afficher avec le nombre resté des policiers

Graphique :

-Notre partie graphique est basée sur l'utilisation de la bibliothèque SDL version 2 du language C/C++

- SDL2_image : pour les images
- SDL2_mixer : pour le son

-Les images utilisées sont crées en totalité par le logiciel Adobe Illustrator
-Les sons sont téléchargés à partir des sites du droits libre

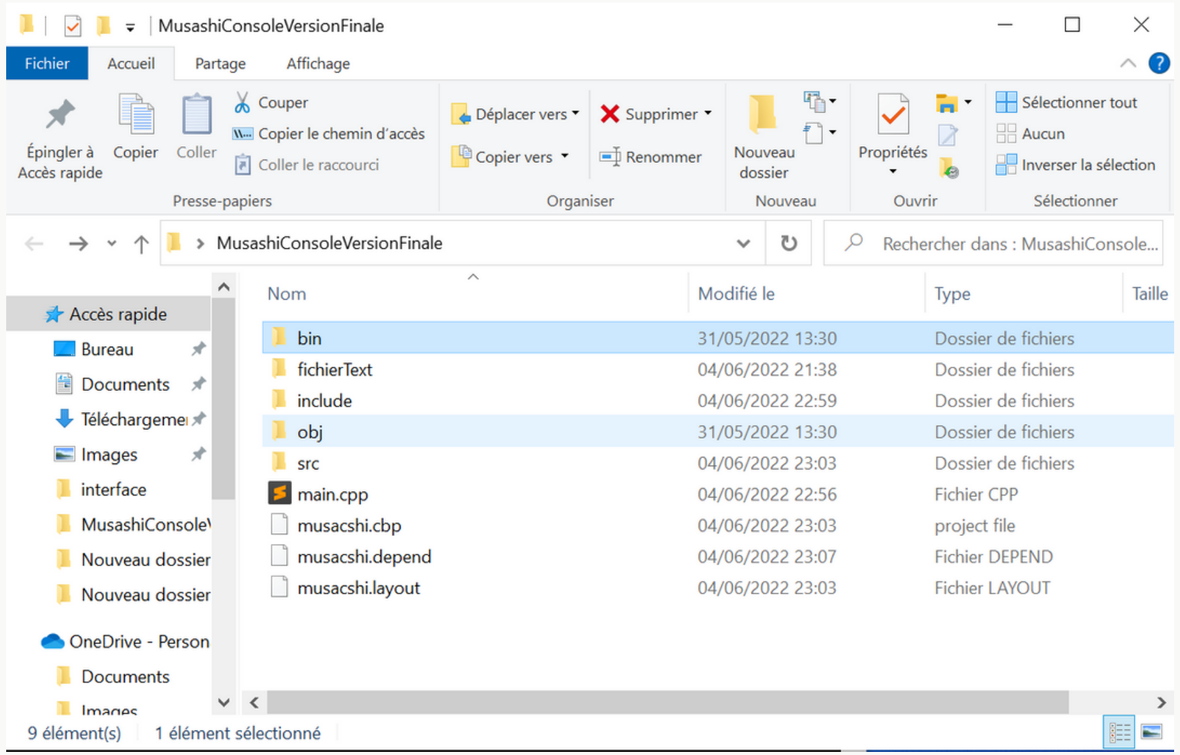


Conception du jeu

Conception du jeu

Notre jeu est sous forme de deux exécutables:

- version console
- version graphique



Conception du jeu

Le joueur est celui qui commence par défaut en premier, sa mission est de tuer le plus grand possible des policiers avant qu'ils l'entournent. A tour de rôle chaque joueur va glisser un pion vers une position voisine libre dans la table.

La structure des deux algorithmes est similaire au cours de l'élément "Résolution des problèmes".

MiniMax
VS AlphaBeta

minimax :

Pour la stratégie minimax, on a utilisé les méthodes de la classe Minimax suivantes:

- **heuristique()** =>la fonction heuristique pour évaluer les noeuds.
- **genersuccesseur()** =>la fonction qui génere les successeurs d'un état.
- **Minimax()**=>la fonction générale de minimax
- **joueurMinimax()**=> la fonction qui retourne le choix de la machine minimax.

alphabeta

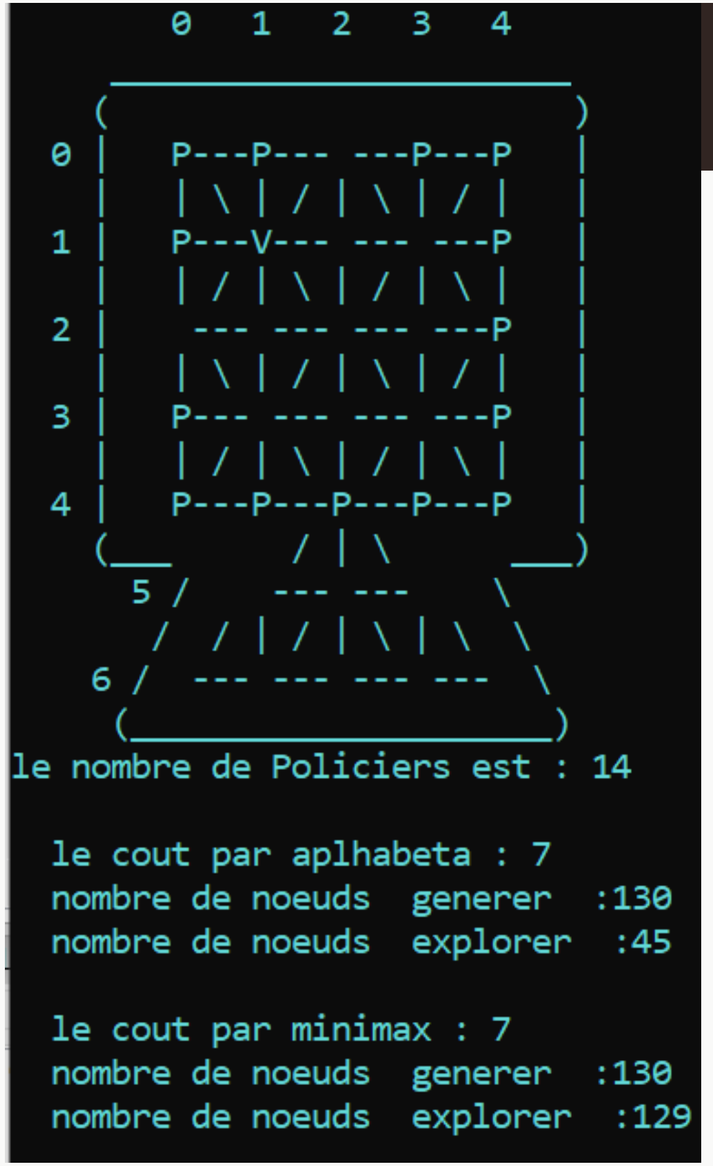
Pour la stratégie alphabeta, on a utilisé les méthodes de la classe Minimax en modifiant les méthodes suivantes:

- **Alphabeta()**=>la fonction générale de alphabeta.
- **joueurAlphabeta()**=> la fonction qui retourne le choix de la machine alphabeta.

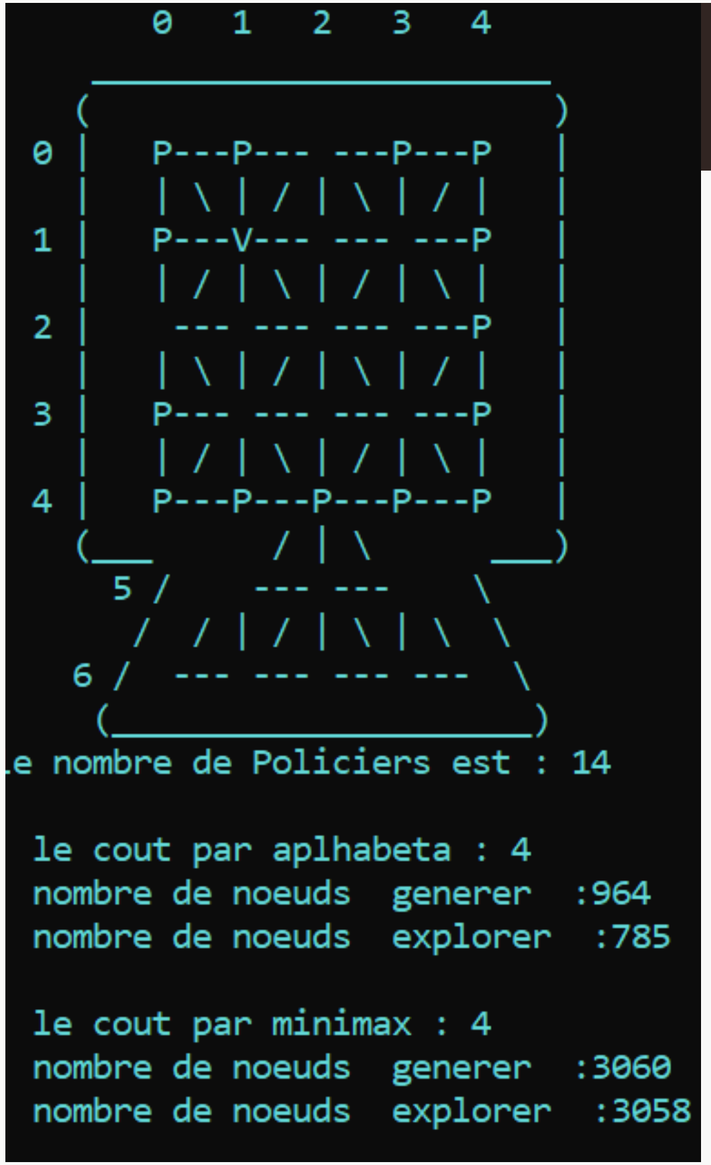
minimax vs alphabeta

Pour les deux stratégies, on a essayé de balayer plusieurs profondeurs pour tester la meilleure et pour comparer les deux algorithmes. Le slide suivant va présenter les captures d'écran pour différentes profondeurs.

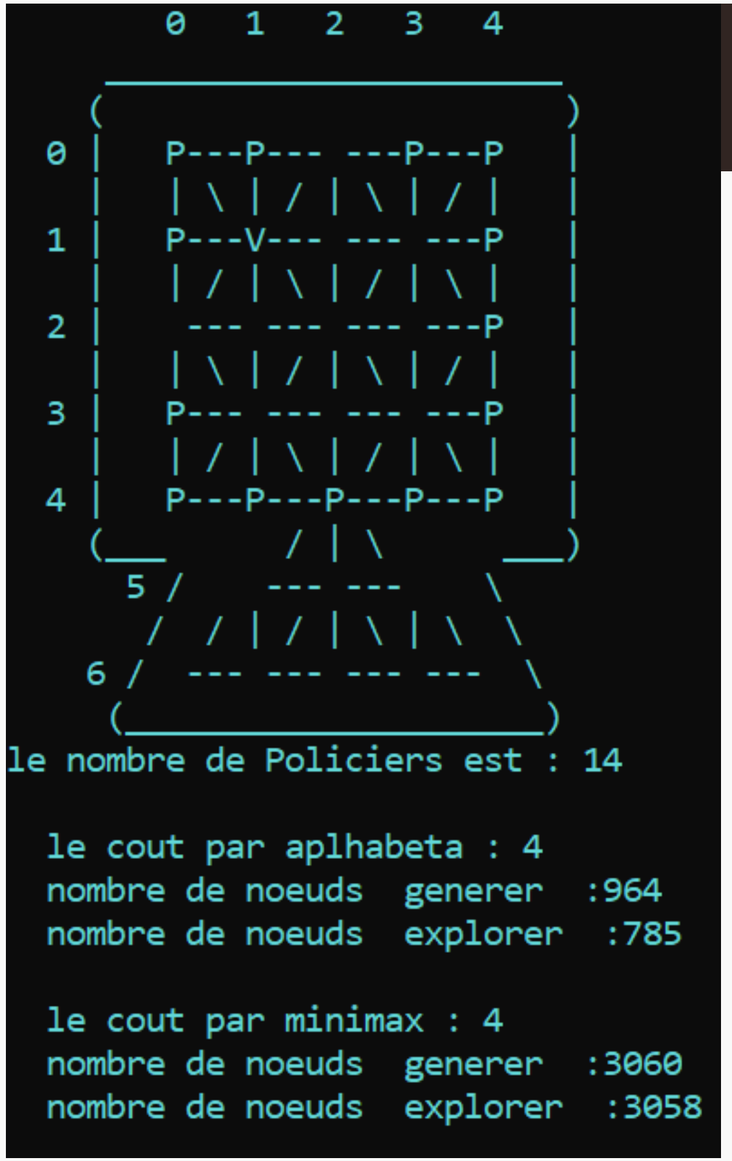
p=2


0 1 2 3 4
0
1
2
3
4
5
6
le nombre de Policiers est : 14
le cout par aplhabeta : 7
nombre de noeuds generer :130
nombre de noeuds explorer :45
le cout par minimax : 7
nombre de noeuds generer :130
nombre de noeuds explorer :129

p=3


0 1 2 3 4
0
1
2
3
4
5
6
e nombre de Policiers est : 14
le cout par aplhabeta : 4
nombre de noeuds generer :964
nombre de noeuds explorer :785
le cout par minimax : 4
nombre de noeuds generer :3060
nombre de noeuds explorer :3058

p=4

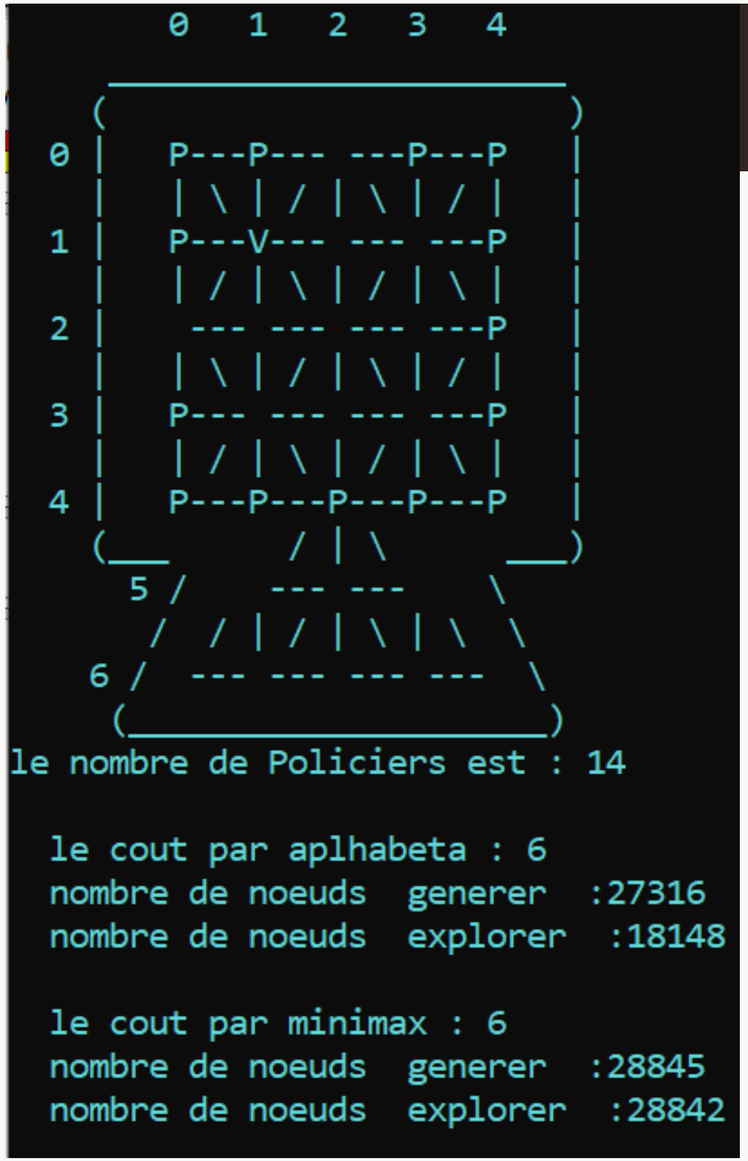



```
le nombre de Policiers est : 14

  le cout par aplhabeta : 4
  nombre de noeuds  generer  :964
  nombre de noeuds  explorer  :785

  le cout par minimax : 4
  nombre de noeuds  generer  :3060
  nombre de noeuds  explorer  :3058
```

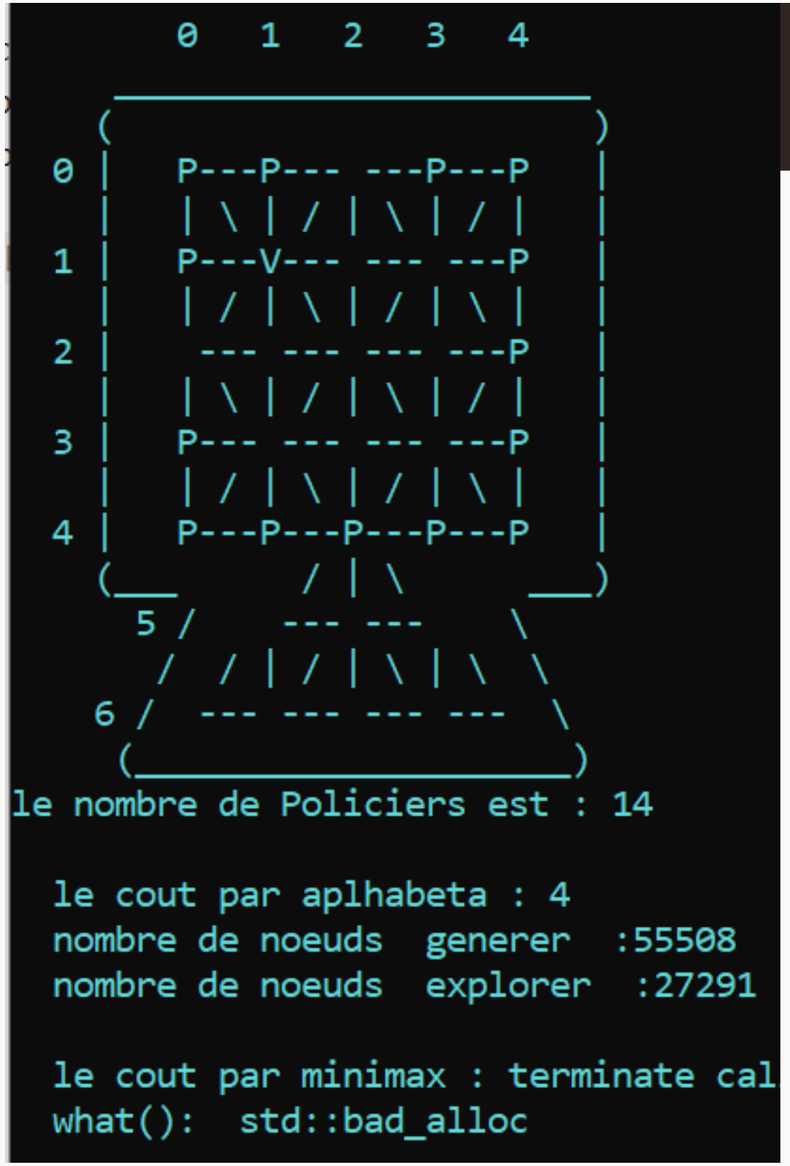
p=5




```
le nombre de Policiers est : 14

  le cout par aplhabeta : 6
  nombre de noeuds  generer  :27316
  nombre de noeuds  explorer  :18148

  le cout par minimax : 6
  nombre de noeuds  generer  :28845
  nombre de noeuds  explorer  :28842
```



p=6

le nombre des nœuds générés avec la méthode minimax dans ce cas à empêcher le système d'allouer la mémoire.

résumé

Le programme alphaBeta est optimal par rapport au programme minimax.

La profondeur qu'on peut atteindre par le programme alphaBeta est p = 6, alors que pour le programme minimax on peut atteindre la profondeur p=5.

Démo

Conclusion

Ce projet est une expérience très intéressante qui nous amène à faire face à un problème difficile et à nous initialiser avec le domaine d’intelligence artificielle.

De plus, il nous a permis d’appliquer les connaissances acquises dans cet élément de module, notamment les différentes stratégies de système de production à savoir le minimax ou l'alphabeta.

Difficultés rencontrées

Durant le progrès de notre projet on a subi comme difficultés d'une part l'optimisation des nœuds explorés. D'autre part, l’autoformation de la programmation sur une interface graphique (SDL).
Mais nous avons pu surmonter ces difficultés au final.

Merci pour votre attention