المدرسة الحسنية للأشغال العمومية
ECOLE HASSANIA DES TRAVAUX PUBLICS

RAPPORT DU PROJET 2023-2024

Réalisé par : Hajar OULABASSE

Mohamed KOUZOUD

Encadré par :Mme CHERRABI

المدرسة الحسنية للأشغال العمومية
ECOLE HASSANIA DES TRAVAUX PUBLICS

Plan :

I. **Présentation du Sujet** :
   a. Introduction au projet et explication de son contexte.

II. **Règles du Jeu** :
   a. Description détaillée du jeu Hex.
   b. Explication des règles et objectifs du jeu.

III. **Travail Demandé** :
   a. Explication des consignes fournies pour la réalisation du projet.

IV. **Codes et explication** :
   a. Présentation des parties principales du code source
   b. Explication du fonctionnement

V. **Conclusion et Perspectives**:
   a. Bilan des objectifs atteints.
   b. Réflexions sur les défis rencontrés

VI. **Remerciements** :

VII. **Annexes** :
   a. Code source complet du programme.
   b. Ressources d'apprentissage

المدرسة الحسنية للأشغال العمومية
ECOLE HASSANIA DES TRAVAUX PUBLICS

Présentation du projet

Ce projet porte sur le développement d'une implémentation du jeu Hex en langage C. Le jeu Hex est un jeu de stratégie abstrait à deux joueurs, inventé dans les années 1940. Son objectif est de relier les côtés opposés du plateau de jeu avec une chaîne continue de ses pièces, formant ainsi une connexion ininterrompue.

Dans le cadre de notre formation académique, nous avons entrepris ce projet pour mettre en pratique nos connaissances en programmation et notre compréhension des algorithmes. Notre objectif est de concevoir une implémentation fonctionnelle du jeu Hex, en mettant l'accent sur la modularité du code, la gestion efficace des ressources et la qualité de jeu contre un adversaire humain ou une intelligence artificielle.

Ce projet représente une occasion précieuse d'approfondir nos compétences en programmation, d'explorer les concepts de jeu et de stratégie, et de relever le défi de créer une application ludique et interactive.

Règles du jeu:

Description détailliée du jeu Hex :

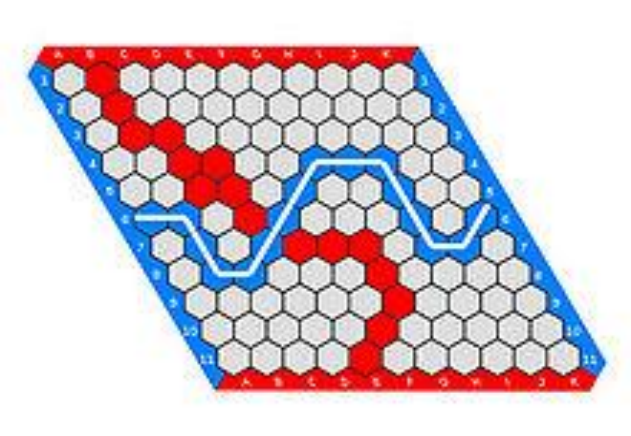
Le jeu Hex est un jeu de stratégie abstrait pour deux joueurs, inventé dans les années 1940 par Piet Hein et John Nash. Le plateau de jeu est composé de cellules hexagonales, généralement disposées en forme de losange. Chaque joueur se voit attribuer une couleur, généralement bleue ou rouge, et le but du jeu est de relier les côtés opposés du plateau avec une chaîne continue de ses pièces, formant ainsi une connexion ininterrompue.

Tour à tour, les joueurs placent une pièce de leur couleur sur une cellule vide du plateau. Une fois qu'une pièce est placée, elle ne peut pas être déplacée ou retirée du plateau. Le premier joueur à réussir à relier ses côtés opposés avec une chaîne continue de ses pièces gagne la partie.

Les règles du Hex sont simples, mais le jeu offre une profondeur stratégique significative. Les joueurs doivent anticiper les mouvements de leur adversaire tout en cherchant à créer leur propre chemin vers la victoire. Le Hex est

connu pour être un jeu équilibré, où la stratégie et la réflexion tactique sont essentielles pour remporter la partie.

En résumé, le Hex est un jeu captivant qui allie simplicité des règles et complexité stratégique, offrant ainsi une expérience de jeu enrichissante pour les joueurs de tous niveaux.



Règles du Jeu Hex :

1. Le Hex se joue à deux joueurs.
2. Chaque joueur se voit attribuer une couleur : l'un joue avec les pierres noires, l'autre avec les pierres blanches.
3. Les joueurs placent leurs pierres un par un à tour de rôle, sur les intersections du quadrillage du plateau. Les pierres peuvent également être placées sur les bords du plateau.
4. Une fois qu'une pierre est placée sur une intersection, elle ne peut plus être déplacée.
5. Si l'un des joueurs est considéré comme plus faible que l'autre, il peut placer des pierres supplémentaires sur le plateau pour compenser cette différence de niveau.
6. En cas d'égalité, le joueur ayant les pierres noires joue toujours en premier. Cependant, dans une partie à handicap, c'est le joueur avec les pierres blanches qui commence.

Travail Demandé :

Implémentation du Jeu Hex : Le projet doit inclure une implémentation complète et fonctionnelle du jeu Hex en langage C. Cela implique la création d'un plateau de jeu hexagonal, la gestion des coups des joueurs et la vérification des conditions de victoire.

Interaction Utilisateur : L'application doit permettre une interaction fluide avec l'utilisateur. Les joueurs doivent pouvoir choisir une case à remplir sur le plateau conformément aux règles du jeu.

Modes de Jeu : Le projet doit offrir au moins deux modes de jeu distincts : un mode joueur contre joueur et un mode joueur contre IA. Dans le mode joueur contre IA, l'IA doit être capable de jouer des coups stratégiques en fonction de la situation du jeu.

Affichage Graphique (Optionnel) : le projet doit inclure un affichage graphique du plateau de jeu pour améliorer l'expérience utilisateur.

Gestion des Erreurs : L'application doit être robuste et capable de gérer les erreurs et les entrées incorrectes de l'utilisateur de manière appropriée. Des messages d'erreur clairs doivent être fournis pour guider l'utilisateur en cas de problème.

Documentation et Rapport : En plus de l'implémentation du jeu, le projet doit être accompagné d'une documentation détaillée.

Code et explication

Partie 1 : Affichage du Plateau de Jeu

Cette partie du code est responsable de l'affichage du plateau de jeu Hex sur la console. Le plateau est représenté par une matrice 9x9, où chaque case peut contenir soit une pierre noire ('N'), une pierre blanche ('B'), ou être vide (' '). Les caractères %c sont remplacés par les valeurs correspondantes de la matrice m

La fonction **hex()** est conçue pour afficher un plateau de jeu Hex, où chaque case est alternativement colorée en rouge et bleu.

Définition des codes de couleur :

- Les directives **#define** sont utilisées pour définir des codes de couleur ANSI, notamment **RED** pour le rouge et **BLUE** pour le bleu, afin de représenter les deux joueurs.

```
#define RED         "\033[31m"
#define GREEN       "\033[32m"
#define RESET       "\033[0m"
#define BLUE        "\033[34m"
#define YELLOW        "\033[33m"
#define  Magenta     "\033[35m"
```

Affichage du titre :

- Un titre de bienvenue est affiché en utilisant la couleur jaune définie par **YELLOW**.

Affichage des colonnes :

- Les lettres A à I représentant les colonnes du plateau sont affichées en utilisant la couleur jaune définie par **YELLOW**.

Affichage des lignes du plateau :

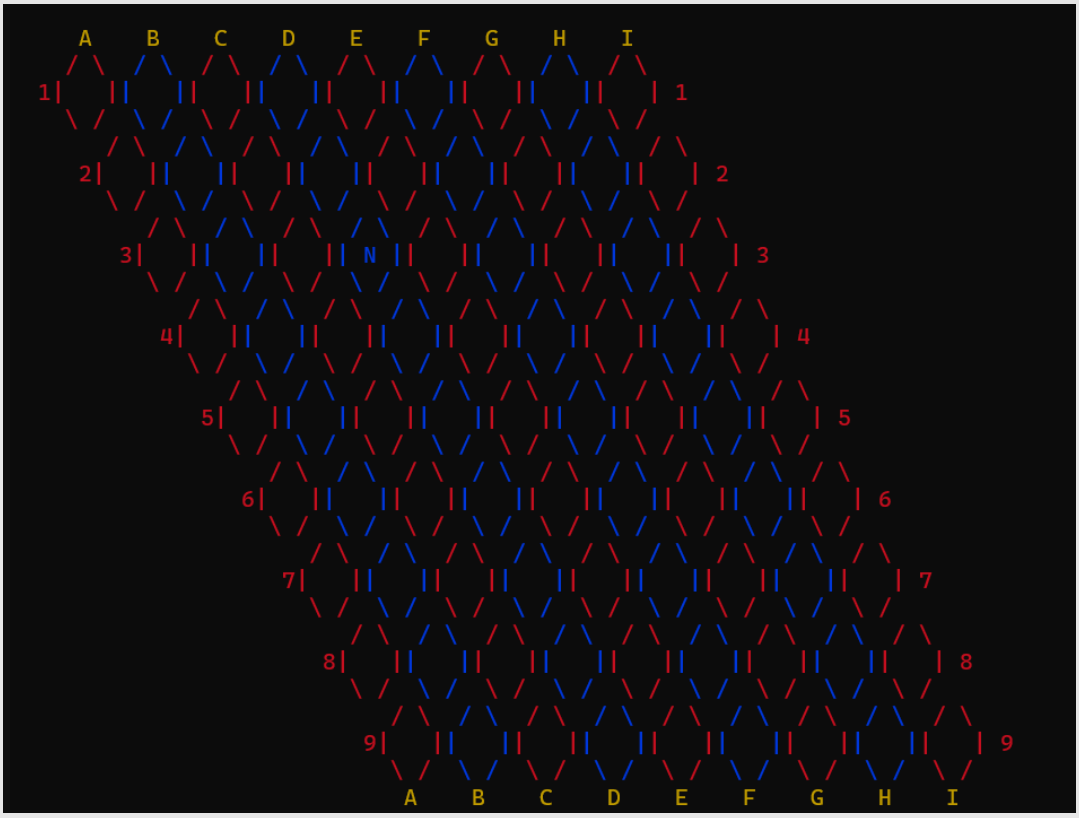
- Les numéros de 1 à 9 représentant les lignes du plateau sont colorées en rouge à l'aide du code de couleur **RED**.

Affichage des cases du plateau :

- Chaque case du plateau est affichée en utilisant les symboles contenus dans le tableau **m**.
- Les cases sont alternativement colorées en rouge (**RED**) et bleu (**BLUE**) .

```
void hex(char m[9][9]){
    system("color 05");
printf("     \n                WELCOME !  \n  \n");
printf("    %sA    %sB    %sC    %sD    %sE    %sF    %sG    %sH    %sI      \n",YELLOW,YELLOW,YELLOW,YELLOW,YELLOW,YELLOW,YELLOW,YELLOW,YELLOW,YELLOW
printf("   %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\    \n ",RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED);
printf(" 1| %c |%s| %c |%s| %c |%s| %c |%s| %c |%s| %c |%s| %c |%s| %c |%s| %c | 1\n ",m[0][0],BLUE,m[0][1],RED,m[0][2],BLUE,m[0][3],RED,m[0][4],BLUE,m
printf("  %s\\ /  %s\\ /  %s\\ /  %s\\ /  %s\\ /  %s\\ /  %s\\ /  %s\\ /  %s\\ /      ",RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED);
printf("  \n      %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\    \n ",RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED);
printf("   2| %c |%s| %c |%s| %c |%s| %c |%s| %c |%s| %c |%s| %c |%s| %c |%s| %c | 2\n ",m[1][0],BLUE,m[1][1],RED,m[1][2],BLUE,m[1][3],RED,m[1][4],BLU
printf("    %s\\ /  %s\\ /  %s\\ /  %s\\ /  %s\\ /  %s\\ /  %s\\ /  %s\\ /  %s\\ /     \n ",RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED);
printf("      %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\    \n ",RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED);
printf("     3| %c |%s| %c |%s| %c |%s| %c |%s| %c |%s| %c |%s| %c |%s| %c | 3\n ",m[2][0],BLUE,m[2][1],RED,m[2][2],BLUE,m[2][3],RED,m[2][4],
printf("      %s\\ /  %s\\ /  %s\\ /  %s\\ /  %s\\ /  %s\\ /  %s\\ /  %s\\ /  %s\\ /     \n ",RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED);
printf("        %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\    \n ",RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED);
printf("       4| %c |%s| %c |%s| %c |%s| %c |%s| %c |%s| %c |%s| %c |%s| %c | 4\n ",m[3][0],BLUE,m[3][1],RED,m[3][2],BLUE,m[3][3],RED,m[3][
printf("        %s\\ /  %s\\ /  %s\\ /  %s\\ /  %s\\ /  %s\\ /  %s\\ /  %s\\ /  %s\\ /     \n ",RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED);
printf("          %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\    \n ",RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED)
```

```
printf("          5| %c |%s| %c |%s| %c |%s| %c |%s| %c |%s| %c |%s| %c |%s| %c |%s| %c | 5\n ",m[4][0],BLUE,m[4][1],RED,m[4][2],BLUE,m[4][3],RED,m[
printf("           %s\\ /  %s\\ /  %s\\ /  %s\\ /  %s\\ /  %s\\ /  %s\\ /  %s\\ /  %s\\ /     \n ",RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED
printf("             %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\    \n ",RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,
printf("            6| %c |%s| %c |%s| %c |%s| %c |%s| %c |%s| %c |%s| %c |%s| %c |%s| %c | 6\n ",m[5][0],BLUE,m[5][1],RED,m[5][2],BLUE,m[5][3],RED
printf("             %s\\ /  %s\\ /  %s\\ /  %s\\ /  %s\\ /  %s\\ /  %s\\ /  %s\\ /  %s\\ /     \n ",RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,
printf("               %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\    \n ",RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED,BL
printf("              7| %c |%s| %c |%s| %c |%s| %c |%s| %c |%s| %c |%s| %c |%s| %c |%s| %c | 7\n ",m[6][0],BLUE,m[6][1],RED,m[6][2],BLUE,m[6][3],
printf("               %s\\ /  %s\\ /  %s\\ /  %s\\ /  %s\\ /  %s\\ /  %s\\ /  %s\\ /  %s\\ /   \n ",RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE
printf("                 %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\    \n ",RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE
printf("                8| %c |%s| %c |%s| %c |%s| %c |%s| %c |%s| %c |%s| %c |%s| %c |%s| %c | 8\n ",m[7][0],BLUE,m[7][1],RED,m[7][2],BLUE,m[7][
printf("                 %s\\ /  %s\\ /  %s\\ /  %s\\ /  %s\\ /  %s\\ /  %s\\ /  %s\\ /  %s\\ /     \n ",RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE
printf("                   %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\    \n ",RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,
printf("                  9| %c |%s| %c |%s| %c |%s| %c |%s| %c |%s| %c |%s| %c |%s| %c |%s| %c | 9\n ",m[8][0],BLUE,m[8][1],RED,m[8][2],BLUE,m[
printf("                   %s\\ /  %s\\ /  %s\\ /  %s\\ /  %s\\ /  %s\\ /  %s\\ /  %s\\ /  %s\\ /     \n ",RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,
printf("                     %sA    %sB    %sC    %sD    %sE    %sF    %sG    %sH    %sI      ",YELLOW,YELLOW,YELLOW,YELLOW,YELLOW,YELLOW,YELLOW
}
```



Partie 2 : Vérification des Lignes et Colonnes et les diagonales et anti-diagonales

Cette partie du code contient plusieurs fonctions chek_lin, check_row, check_diag, check_sec_diag, qui sont utilisées pour vérifier si une ligne une colonne ou une diagonale ou une anti-diagonale du plateau de jeu est entièrement occupée par la letter qui correspond au joueur ou la machine. Ces fonctions sont utilisées pour determiner qui a gagné

```
int chek_rowB(char m[9][9]) {

     int cmp=0;
          for (int j = 0; j < 9; j++){
          cmp=0;
          for (int i = 0; i < 9; i++) {
          if (m[j][i] == 'B') {
          cmp++;
          }
     }if(cmp==9){break;}
          } return cmp == 9;
}
```

```
int chek_linB(char m[9][9]) {

     int cmp=0;
     for (int j = 0; j < 9; j++){
          cmp=0;
          for (int i = 0; i < 9; i++) {
          if (m[i][j] == 'B') {
               cmp++;
          }
     } if(cmp==9){break;}

     } return cmp == 9;

}
```

```
int chek_rowN(char m[9][9]) {

     int cmp = 0;
     for (int j = 0; j < 9; j++){
               cmp = 0;
          for (int i = 0; i < 9; i++) {
          if (m[j][i] == 'N') {
               cmp++;
          }
     } if(cmp==9){break;}

     } return cmp == 9;
}
```

```c
int chek_linN(char m[9][9]) {

    int cmp = 0;
    for (int j = 0; j < 9; j++){
        cmp = 0;
        for (int i = 0; i < 9; i++) {
        if (m[i][j] == 'N') {
            cmp++;
        }
    } if(cmp==9){break;}
    } return cmp == 9;
}
```

```c
int chek_diagB(char m[9][9]) {
    int cmp = 0;
    for (int i = 0; i < 9; i++) {
        if (m[i][i] == 'B') {
            cmp++;
        }
    }
    return cmp == 9;
}
```

```c
int chek_diagN(char m[9][9]) {
    int cmp = 0;
    for (int i = 0; i < 9; i++) {
        if (m[i][i] == 'N') {
            cmp++;
        }
    }
    return cmp == 9;
}
```

```
int chek_sec_diagN(char m[9][9]) {
    int cmp = 0;
    for (int i = 0; i < 9; i++) {
        if (m[i][8 - i] == 'N') {
            cmp++;
        }
    }
    return cmp == 9;
}
```

la fonction de rafraîchissement:

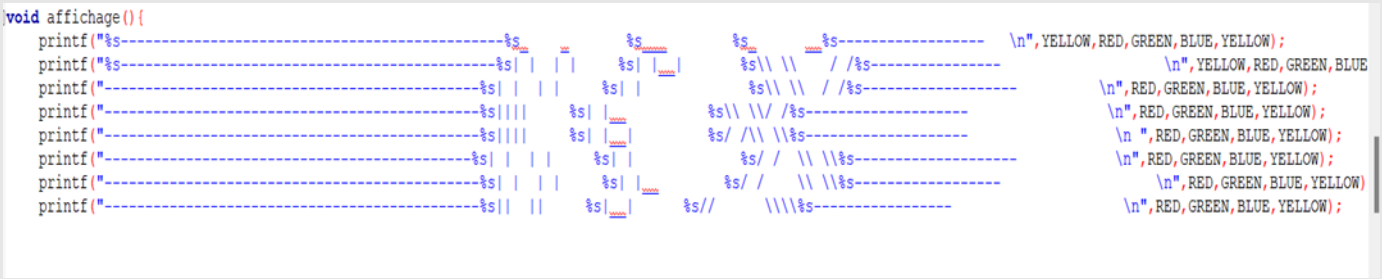
```
void actualiser(int n){
    sleep(n);
    system("cls");
}
```

La fonction actualiser est destinée à actualiser l'affichage du plateau après un certain délai n.

sleep(n) : Cette instruction suspend l'exécution du programme pendant n secondes. Cela signifie que le programme attend pendant n secondes avant de continuer à s'exécuter.

system("cls") : Cette fonction appelle une commande système pour effacer l'écran de la console. Sur les systèmes d'exploitation Windows, la commande cls efface l'écran. Cela permet de nettoyer la console avant d'afficher une nouvelle version mise à jour du plateau de jeu.

La fonction Affichage:



```
void affichage(){
    printf("%s----------%s  _   %s____   %s  __%s----------  \n",YELLOW,RED,GREEN,BLUE,YELLOW);
    printf("%s----------%s| |  | |  %s| |__|  %s\\ \\  / /%s----------  \n",YELLOW,RED,GREEN,BLUE
    printf("----------%s| |  | |  %s| |  %s\\ \\  / /%s----------  \n",RED,GREEN,BLUE,YELLOW);
    printf("----------%s||||  %s| |__  %s\\ \\/ /%s----------  \n",RED,GREEN,BLUE,YELLOW);
    printf("----------%s||||  %s| |__| %s/ /\\ \\%s----------  \n ",RED,GREEN,BLUE,YELLOW);
    printf("----------%s| |  | |  %s| |  %s/ /  \\ \\%s----------  \n",RED,GREEN,BLUE,YELLOW);
    printf("----------%s| |  | |  %s| |__ %s/ /  \\ \\%s----------  \n",RED,GREEN,BLUE,YELLOW)
    printf("----------%s||  ||  %s|___| %s//  \\\\%s----------  \n",RED,GREEN,BLUE,YELLOW);
```

La fonction affichage() ne prend aucun paramètre et ne retourne rien (void).

Corps de la fonction :

Utilisation des séquences d'échappement ANSI pour la couleur :

Les séquences d'échappement ANSI sont utilisées pour colorifier le texte dans la console.

Les couleurs utilisées sont définies auparavant dans le code par: YELLOW, RED, GREEN, et BLUE.

Chaque printf() affiche une ligne du motif. Les caractères spéciaux %s dans les chaînes sont remplacés par les séquences d'échappement ANSI correspondantes pour changer la couleur du texte.

Motif créé :

Le motif créé est une combinaison de lignes horizontales et verticales, formant la structure suivante:

Partie 3 : la fonction MODE_IA_1 :

```
char MODE_IA_1(char m[9][9]){
      int number_PL,number_IA;
      int k=0;
      char Letter_IA,Letter_PL;
      char name_player[20];
      printf("%s                                                    ~~~~~~~~~~",GREEN);
      printf("\n                                                  %s|| enter your name :       ||\n",YELLOW);
      printf("%s                                                   ~~~~~~~~~~%s\n",GREEN,RESET);
      printf(" %s                                                            ",RED);scanf("%s",&name_player);


         while(k<81){

            hajar:
           srand(time(NULL));
            number_IA=rand()%10;
            Letter_IA='A' + rand() % 9;
            Letter_IA=Letter_IA-'A';
            if((m[number_IA-1][Letter_IA]=='B')||(m[number_IA-1][Letter_IA]=='N')){ goto hajar;}

            m[number_IA-1][Letter_IA]='N';
            printf("\nAI's Move:\n");
            hex(m);
            if (chek_rowN(m) || chek_linN(m)||chek_diagN(m)||chek_sec_diagN(m)) {
            printf("\nIA wins!\n");
            break;}
```

```
        kouzoud:
        printf("\n %sPlayer %s 's turn to play :\n ",GREEN,name_player);
        printf("%s enter the letter\n",YELLOW);
        scanf(" %c", &Letter_PL);
        printf("%s enter the number\n",YELLOW);
        scanf(" %d",&number_PL);
        if(number_PL==0||Letter_PL=='Z'){printf("%sEND OF GAME!%s ",RED,RESET);exit(0);}
        if ((number_PL < 1 || number_PL > 9)||(Letter_PL<'A'||Letter_PL>'I')) {
            printf("%s Invalid input. Please enter valid values between %s 1 and 9,%s and letters between %s A and I\n",YELLOW,RED,YELLOW,RED);
            goto kouzoud;
            }
            Letter_PL = Letter_PL - 'A';
            if((m[number_PL-1][Letter_PL]=='B')||(m[number_PL-1][Letter_PL]=='N')){printf("%s is pline",RED);  goto kouzoud;}
            m[number_PL-1][Letter_PL]='B';
            hex(m);
            if (chek_rowB(m) || chek_linB(m)||chek_diagB(m)||chek_sec_diagB(m)) {
            printf("\n %s congratulation %s %s %s you are the winner!\n%s",RED,GREEN,name_player,RED,RESET);
            break;}


k++;
```

- char MODE_IA_1(char m[9][9]): Définition de la fonction MODE_IA_1 prenant en paramètre une matrice m représentant le plateau de jeu.
- int number_PL,number_IA: Déclaration des variables ‘number_PL’ et ‘number_IA` pour stocker les numéros de ligne du joueur et de la machine.
- int k=0: Initialisation d'une variable k à zéro pour compter le nombre de tours.
- char Letter_IA,Letter_PL: Déclaration des variables Letter_IA et Letter_PL pour stocker les lettres des mouvements des joueurs.
- char name_player[20]: Déclaration d'un tableau de caractères ‘name_player’ pour stocker le nom du joueur.
- printf(...): Affichage d'un message pour demander le nom du joueur.
- scanf("%s",&name_player): Lecture du nom du joueur à partir de l'entrée utilisateur.
- while(k<81)Début d'une boucle ‘while’ pour exécuter le jeu jusqu'à ce que le plateau soit rempli ou qu'un joueur gagne.(81 le nombre maximum des iterations possible avant que je je finisse)
- ‘hajar’: Marqueur pour la boucle `while`.
- srand(time(NULL)): utilisé pour initialiser le générateur de nombres pseudo-aléatoires en fonction du temps actuel, afin d'éviter la répétition des séquences de nombres aléatoires lors de l'exécution du programme.
- number_IA=rand()%10: Génération aléatoire d'un nombre pour la ligne de l'IA.
- Letter_IA='A' + rand() % 9: Génération aléatoire d'une lettre pour la colonne de l'IA.
- Letter_IA=Letter_IA-'A': Conversion de la lettre en un indice numérique.
- if((m[number_IA-1][Letter_IA]=='B')||(m[number_IA-1][Letter_IA]=='N')){ goto simo;}`: Vérification si la case choisie par l'IA est déjà occupée. Si c'est le cas, elle recommence à choisir une autre case.
- m[number_IA-1][Letter_IA]='N';`: Placement du symbole de l'IA (par exemple, 'N') sur le plateau.
- printf("\nAI's Move:\n"): Affichage du mouvement de l'IA.

- hex(m): Affichage du plateau de jeu.
- if (chek_rowN(m) || chek_linN(m)||chek_diagN(m)||chek_sec_diagN(m)) { printf("\nIA wins!\n"); break;}`: Vérification si l'IA a gagné après son mouvement.
- 'kouzoud': Marqueur pour la boucle `do...while` du joueur.

⟹ De meme la saisie et la validation du mouvement du joueur.

Cette partie du code gère essentiellement le déroulement du jeu en alternant entre les mouvements de l'IA et du joueur, avec des vérifications pour déterminer s'il y a un gagnant ou si le jeu est terminé.

⟹ mettre à jour l'affichage du jeu après chaque tour,en assurant une expérience de jeu fluide et permettant au joueur de visualiser les derniers mouvements effectués.

```
k++;
actualiser(0);
      }

return 0;
}
```

Partie 4 :la fonction MODE_TWO_PLAYERS :

```
char MODE_TWO_PLAYERS(char m[9][9]){
int x=0;
srand(time(NULL));
int number_player2,number_player1;
int random_start= rand() % 2;
char letter_player1,letter_player2;
char name_player1[20];
char name_player2[20];
printf("                                        %s ---------------------\n",GREEN);
printf("                                        %s name of first player: \n",YELLOW);
printf("                                        %s ---------------------\n",GREEN);
printf("                                                  ");
scanf(" %s",&name_player1);
printf("                                        %s ********************\n",Magenta);
printf("                                        %s ---------------------\n",GREEN);
printf("                                        %s name of second player: \n",YELLOW);
printf("                                        %s ---------------------\n",GREEN);
printf("                                                  ");
scanf(" %s",&name_player2);
//hex(m);
 if (random_start == 0) {
        printf("\n%s goes first!\n", name_player1);
    } else {
        printf("\n%s goes first!\n", name_player2);
    }
        while(x<81){
        hex(m);
        if (random_start == 0){
```

```
            hajar:
        printf("\n %s Player %s 's turn to play :\n ",GREEN,name_player1);
        printf("%s Enter the letter (A-I):\n",YELLOW);
        scanf(" %s", &letter_player1);
        printf("%s Enter the number (1-9):\n",YELLOW);
        scanf(" %d",&number_player1);
        if(number_player1==0||letter_player1=='Z'){printf("%sEND OF GAME!%s ",RED,RESET);exit(0);}
        if ((number_player1 < 1 || number_player1 > 9)||(letter_player1<'A' || letter_player1>'I')) {
            printf("%s Invalid input. Please enter valid values between %s 1 and 9,%s and letters between %s A and I\n",YELLOW,RED,YELLOW,RED);
            goto hajar;

            }
            letter_player1 = letter_player1 - 'A';
            if((m[number_player1-1][letter_player1]=='N')||(m[number_player1-1][letter_player1]=='B')){printf("is pline");goto hajar;}
            m[number_player1-1][letter_player1]='N';
            //hex(m);
            if (chek_rowN(m) || chek_linN(m)||chek_diagN(m)||chek_sec_diagN(m)) {
            printf("\n %s congratulation %s %s  %s you are the winner!\n%s",RED,GREEN,name_player1,RED,RESET);
            break;}
        }

         else{
        kouzoud:
        printf("\n %s Player %s's turn to play\n",GREEN,name_player2);
        printf("%s Enter the letter (A-I):\n",YELLOW);
        scanf(" %c", &letter_player2);
        printf("%s Enter the number (1-9):\n",YELLOW);
        scanf(" %d",&number_player2);
        if(number_player2==0||letter_player2=='Z'){exit(0);}
        if ((number_player2 < 1 || number_player2 > 9)||(letter_player2<'A' || letter_player2>'I')) {
```

```
        if ((number_player2 < 1 || number_player2 > 9)||(letter_player2<'A' || letter_player2>'I')) {
            printf("%s Invalid input. Please enter valid values between %s 1 and 9,%s and letters between %s A and I\n",YELLOW,RED,YELLOW,RED);
            goto kouzoud;
            }
            letter_player2 = letter_player2 - 'A';
            if((m[number_player2-1][letter_player2]=='B')||(m[number_player2-1][letter_player2]=='N')){printf("is pline");  goto kouzoud;}
            m[number_player2-1][letter_player2]='B';
            if (chek_rowB(m) || chek_linB(m)||chek_diagB(m)||chek_sec_diagB(m)) {
            printf("\n %s congratulation %s %s  %s you are the winner!\n%s",RED,GREEN,name_player2,RED,RESET);
            break;}
         }


x++;
```

- La fonction **MODE_TWO_PLAYERS** est définie pour permettre le déroulement du jeu en mode deux joueurs, prenant en paramètre une matrice **m** représentant le plateau de jeu.
- Les variables **x**, **number_player2**, **number_player1**, **random_start**, **letter_player1**, **letter_player2**, **name_player1**, et **name_player2** sont déclarées pour stocker diverses informations nécessaires au jeu.
- La fonction **rand()** est utilisée en conjonction avec **srand(time(NULL))** pour initialiser le générateur de nombres pseudo-aléatoires en fonction du temps actuel.
- Le joueur qui commencera la partie est choisi de manière aléatoire à l'aide de **rand() % 2**.
- Les noms des deux joueurs sont demandés à l'utilisateur à l'aide de **scanf**.
- Le jeu se déroule dans une boucle **while** qui continue tant que le plateau n'est pas rempli ou qu'un joueur ne gagne pas.
- À chaque itération, le plateau est affiché à l'aide de la fonction **hex(m)**.
- Le joueur dont c'est le tour est déterminé en fonction de la valeur de **random_start**.
- Chaque joueur est invité à entrer une lettre et un nombre correspondant à la case où il souhaite placer son symbole ('N' pour le premier joueur et 'B' pour le deuxième).
- Les coordonnées entrées par les joueurs sont vérifiées pour s'assurer qu'elles sont valides et que la case n'est pas déjà occupée.
- Si les conditions de victoire sont remplies pour l'un des joueurs, un message de félicitations est affiché, et le jeu se termine.
- À chaque itération de la boucle, la valeur de **random_start** est mise à jour pour alterner entre les joueurs; random_start = (random_start + 1) % 2; est utilisée pour alterner le joueur qui commence à jouer à chaque tour du jeu elle increment le variable random_start à chaque fois .

- La fonction **actualiser(0)** est appelée pour mettre à jour l'affichage du plateau après chaque tour.

Partie 4 :la fonction menu

```
void menu(){
    system("color 02");
    char m[9][9];
    char ok;
    for(int i=0;i<9;i++){
    for(int j=0;j<9;j++)(m[i][j]=' ';})
     int i,j,n;
     affichage();
     printf("      \n\n\n \n
     printf("
     do{printf("
     printf("
     printf("
     printf("
     printf("
     printf("
     printf("
     printf("
     printf("
     }while((i!=1)&&(i!=2)&&(i!=3));
     actualiser(1);
```

```
                %s_____\n",RED);
%s\\_____|_____|_____|_____|_____|_____/\n",RED);
    %s\\ \\          %s menu:                 %s\\ \\ \n",GREEN,BLUE,GREEN);
  %s\\ \\          %s ~~~~~                %s\\ \\\n",GREEN,BLUE,GREEN);
   %s\\ \\          %s  1.Play              %s\\ \\\n",GREEN,BLUE,GREEN);
    %s\\ \\          %s 2.rules of game      %s\\ \\\n",GREEN,BLUE,GREEN);
     %s\\ \\          %s 3.exit               %s\\ \\\n",GREEN,BLUE,GREEN);
      %s\\ \\                                 \\ \\\n",GREEN);
       %s\\ \\                                 \\ \\\n",GREEN);
        %s\\ \\                                 \\ \\\n",GREEN);
          \t\t\t");scanf("%d",&i);
```

La fonction **menu()** est une fonction principale de notre programme, qui agit comme un menu interactif pour l'utilisateur. Elle permet à l'utilisateur de choisir parmi différentes options, comme jouer à un jeu, afficher les règles du jeu ou quitter le programme.

1-**Initialisation du tableau m** :

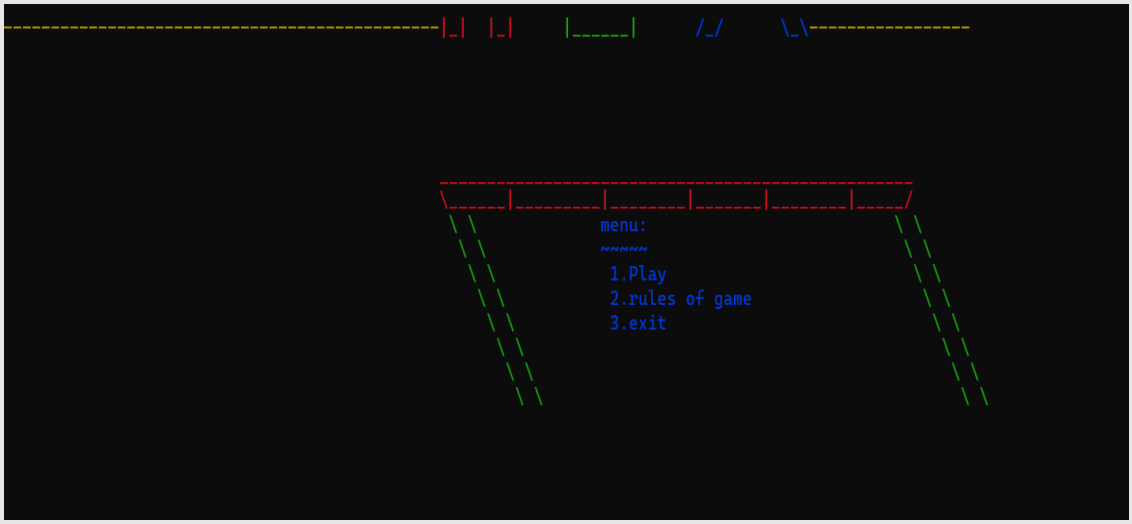
- Au début de la fonction, on initialise un tableau **m** de taille 9x9 avec des espaces. Ce tableau représente le plateau de jeu hexagonal.

2. **Affichage du menu principal** :

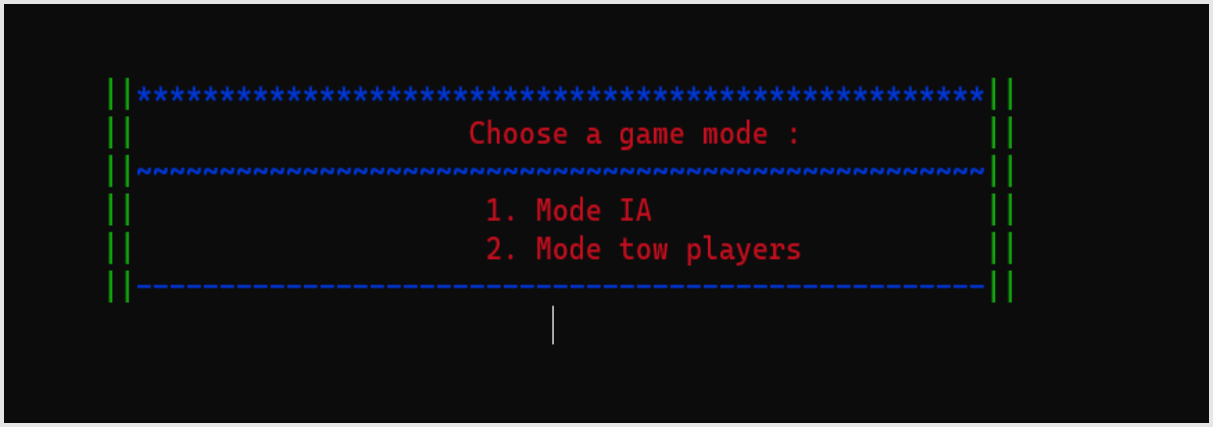
- La fonction affiche un menu principal interactif à l'utilisateur. Elle utilise des séquences d'échappement ANSI pour colorer et mettre en forme le texte affiché à l'écran.

3. **Gestion du choix de l'utilisateur** :

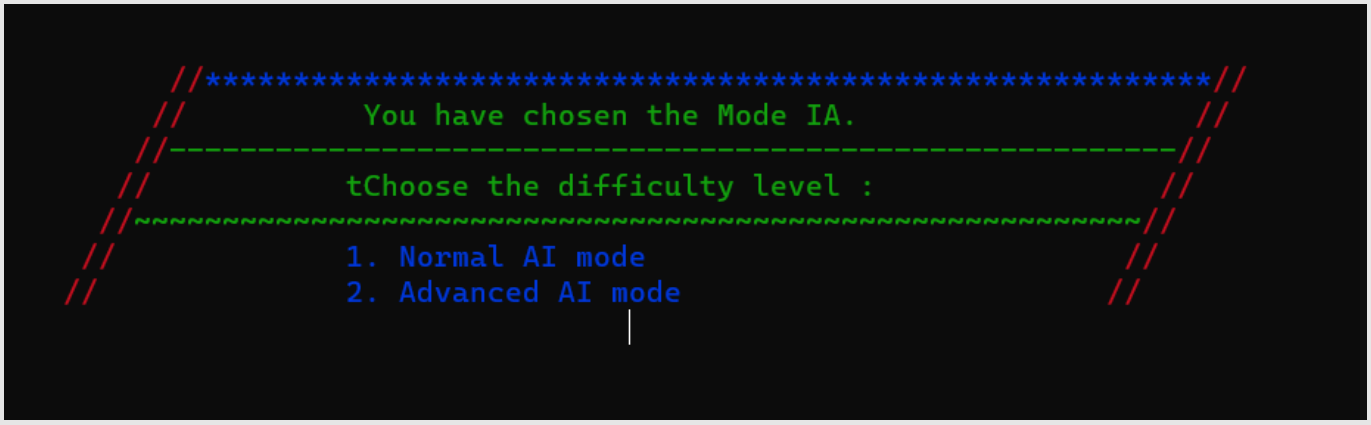
- La fonction utilise une boucle **do-while** pour permettre à l'utilisateur de choisir une option valide. L'utilisateur peut choisir entre jouer, afficher les règles du jeu ou quitter le programme.



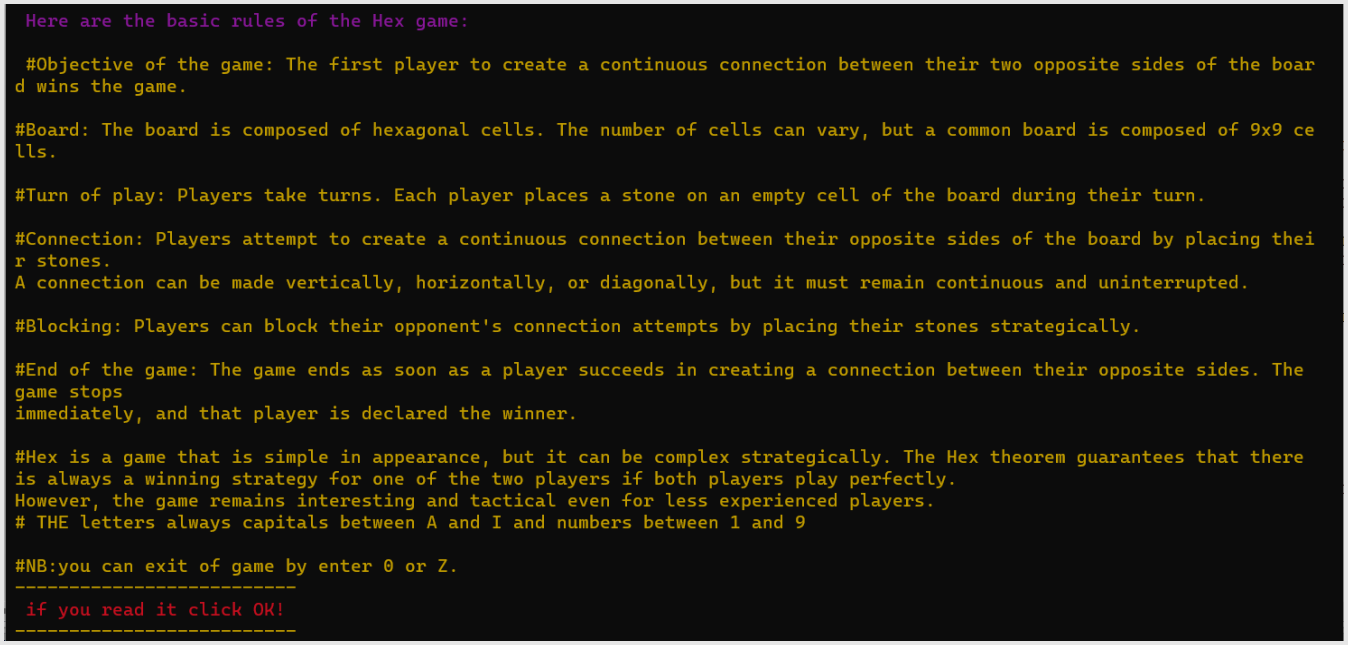
4. **Sous-menu pour le choix de jeu** :
    - Si l'utilisateur choisit de jouer, la fonction affiche un sous-menu interactif lui permettant de choisir entre différents modes de jeu, comme le mode IA ou le mode à deux joueurs.



Si le joueur choisit le mode IA la fonction affiche un sous menu qui lui permet de choisir entre le niveau de difficulté normal ou avancée.



5. **Gestion des choix de jeu** :
    - En fonction du choix de l'utilisateur dans le sous-menu, la fonction appelle les fonctions correspondantes pour lancer le jeu dans le mode sélectionné. Par exemple, si l'utilisateur choisit le mode IA, la fonction appelle **MODE_IA**_pour démarrer le jeu en mode IA.
6. **Affichage des règles du jeu** :
    - Si l'utilisateur choisit d'afficher les règles du jeu, la fonction affiche un ensemble de règles détaillées expliquant le fonctionnement du jeu Hex. Elle utilise des séquences d'échappement ANSI pour colorer le texte et rend l'affichage plus attrayant.



7. **Sortie du programme** :
    - Si l'utilisateur choisit de quitter le programme, la fonction affiche un message de sortie et utilise la fonction **exit(0)** pour quitter le programme proprement.
8. **Gestion de la récursion** :
    - Après chaque choix effectué par l'utilisateur, la fonction appelle à nouveau **menu()** pour afficher à nouveau le menu principal et permettre à l'utilisateur de choisir une autre option. Cela crée une boucle de menu récursive jusqu'à ce que l'utilisateur choisisse de quitter le programme.

```
void menu(){
    system("color 02");
    char m[9][9];
    char ok;
    for(int i=0;i<9;i++){
    for(int j=0;j<9;j++){m[i][j]=' ';}}
     int i,j,n;
     affichage();
     printf("     \n\n\n \n                                    %s_____\n",RED);
     printf("                                              %s\\_____|_____|_____|_____|_____|_____/\n",RED);
     do{printf("                                             %s\\ \\           %s menu:                %s\\ \\ \n",GREEN,BLUE,GREEN);
     printf("                                            %s\\ \\          %s ~~~~~                 %s\\ \\\n",GREEN,BLUE,GREEN);
     printf("                                            %s\\ \\          %s  1.Play              %s\\ \\\n",GREEN,BLUE,GREEN);
     printf("                                            %s\\ \\          %s 2.rules of game      %s\\ \\\n",GREEN,BLUE,GREEN);
     printf("                                            %s\\ \\          %s 3.exit               %s\\ \\\n",GREEN,BLUE,GREEN);
     printf("                                            %s\\ \\                                  \\ \\\n",GREEN);
     printf("                                            %s\\ \\                                  \\ \\\n",GREEN);
     printf("                                            %s\\ \\                                  \\ \\\n",GREEN);
     printf("                                              \t\t\t");scanf("%d",&i);
    }while((i!=1)&&(i!=2)&&(i!=3));
    actualiser(1);          int menu::i
```

```
if(i==1){
      printf("\n\n");
   printf("                    %s||%s**************************************************%s|||\n",GREEN,BLUE,GREEN);
      do{
   printf("                    %s||                 %sChoose a game mode :          %s||          \n",GREEN,RED,GREEN);
   printf("                    %s||%s~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~%s|||\n",GREEN,BLUE,GREEN);
   printf("                    %s||                 %s 1. Mode IA                   %s|||\n",GREEN,RED,GREEN);
   printf("                    %s||                 %s 2. Mode tow players          %s||  \n",GREEN,RED,GREEN);
   printf("                    %s||%s-------------------------------------------------%s|||\n",GREEN,BLUE,GREEN);
   printf("                         \t\t\t");scanf("%d",&j);
      }while((j!=1)&&(j!=2));
actualiser(1);
   if(j==1){
         printf("\n\n");
         printf("                      %s//%s************************************************************%s//\n",RED,BLUE,RED);
        do{
         printf("                      %s//%s         You have chosen the Mode IA.                %s//\n",RED,GREEN,RED);
         printf("                      %s//%s-------------------------------------------------------%s// \n",RED,GREEN,RED);
         printf("                      %s//%s          tChoose the difficulty level :             %s//\n",RED,GREEN,RED);
         printf("                      %s//%s~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~%s//\n",RED,GREEN,RED);
         printf("                      %s//%s            1. Normal AI mode                    %s//\n",RED,BLUE,RED);
         printf("                      %s//%s            2. Advanced AI mode                  %s//\n",RED,BLUE,RED);
         printf("                              \t\t\t");scanf("%d",&n);
        }while((n!=1)&&(n!=2));
        actualiser(1);
         if(n==1){
         printf("                                      %s ~~~~~~~~~~~~~~~~~~~~~~~\n",GREEN);
         printf("                                     %s ||Normal AI mode apply:   ||\n",YELLOW);
         printf("                                      %s ~~~~~~~~~~~~~~~~~~~~~~~\n",GREEN);
         printf("                                       %s************************\n",BLUE);
         MODE_IA_1(m);}
```

```
        else if(n==2){printf("~~~~~~~~~~~~~~~~~~~~~~~~\n");
        printf("Advanced AI mode apply: \n");
        printf("~~~~~~~~~~~~~~~~~~~~~~~~\n");}
        }
   else if(j==2){
   printf("                                    %s~~~~~~~~~~~~~~~~~~~~\n",RED);
   printf("                                   %s Two player mode apply: \n",YELLOW);
   printf("                                  %s  ~~~~~~~~~~~~~~~~~~~~\n",RED);
   printf("                                   %s ********************\n",Magenta);
   MODE_TWO_PLAYERS(m);}
 }
 else if(i==2){
printf("%s Here are the basic rules of the Hex game:\n\n",Magenta);
printf("%s #Objective of the game: The first player to create a continuous connection between their two opposite sides of the board wins the game.\n\n
printf("#Board: The board is composed of hexagonal cells. The number of cells can vary, but a common board is composed of 9x9 cells.\n\n");
printf("#Turn of play: Players take turns. Each player places a stone on an empty cell of the board during their turn.\n\n");
printf("#Connection: Players attempt to create a continuous connection between their opposite sides of the board by placing their stones. \nA connecti
printf("#Blocking: Players can block their opponent's connection attempts by placing their stones strategically.\n\n");
printf("#End of the game: The game ends as soon as a player succeeds in creating a connection between their opposite sides. The game stops \nimmediate
printf("#Hex is a game that is simple in appearance, but it can be complex strategically. The Hex theorem guarantees that there \nis always a winning
printf("# THE letters always capitals between A and I and numbers between 1 and 9 \n");
printf("\n#NB:you can exit of game by enter 0 or Z. ");
printf("\n%s------------------------\n",YELLOW);
printf("%s if you read it click OK!\n",RED);
printf("%s------------------------\n",YELLOW);
scanf("%s",&ok);
actualiser(1);
menu();

 }
```

```
        }
        else if (i == 3) {
            printf("%s\n Exiting the program.\n\n%s",RED,RESET);
            exit(0);
        }

    }


int main(){

menu();


}
```

Annexe :

Vue d'ensemble du code :

```c
#include<stdio.h>
#include<stdlib.h>
#include <unistd.h>
#include <time.h>
#define RED     "\033[31m"
#define GREEN   "\033[32m"
#define RESET   "\033[0m"
#define BLUE    "\033[34m"
#define YELLOW    "\033[33m"
#define  Magenta  "\033[35m"
void hex(char m[9][9]){
    system("color 05");
printf("      \n                WELCOME !  \n  \n");
printf("    %sA    %sB    %sC    %sD    %sE    %sF    %sG    %sH    %sI     \n",YELLOW,YELLOW,YELLOW,YELLOW,YELLOW,YELLOW,YELLOW,YELLOW,YELLOW,YELLOW);
printf("    %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\    \n ",RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED);
printf(" 1| %c |%s| %c |%s| %c |%s| %c |%s| %c |%s| %c |%s| %c |%s| %c | 1\n ",m[0][0],BLUE,m[0][1],RED,m[0][2],BLUE,m[0][3],RED,m[0][4],BLUE,m[0][5],RED,m[0][6],BLUE,m[0][7],RED,m[0][8]);
printf("   %s\\ / %s\\ / %s\\ / %--s\\ / %s\\ / %s\\ / %s\\ / %s\\ /      ",RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED);
printf("   \n      %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\    \n ",RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED);
printf("    2| %c |%s| %c |%s| %c |%s| %c |%s| %c |%s| %c |%s| %c |%s| %c | 2\n ",m[1][0],BLUE,m[1][1],RED,m[1][2],BLUE,m[1][3],RED,m[1][4],BLUE,m[1][5],RED,m[1][6],BLUE,m[1][7],RED,m[1][8]);
printf("     %s\\ / %s\\ / %s\\ / %s\\ / %s\\ / %s\\ / %s\\ / %s\\ /     \n ",RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED);
printf("        %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\     \n ",RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED);
printf("       3| %c |%s| %c |%s| %c |%s| %c |%s| %c |%s| %c |%s| %c |%s| %c | 3\n ",m[2][0],BLUE,m[2][1],RED,m[2][2],BLUE,m[2][3],RED,m[2][4],BLUE,m[2][5],RED,m[2][6],BLUE,m[2][7],RED,m[2][
printf("        %s\\ / %s\\ / %s\\ / %s\\ / %s\\ / %s\\ / %s\\ / %s\\ /     \n ",RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED);
printf("           %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\     \n ",RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED);
printf("         4| %c |%s| %c |%s| %c |%s| %c |%s| %c |%s| %c |%s| %c |%s| %c | 4\n ",m[3][0],BLUE,m[3][1],RED,m[3][2],BLUE,m[3][3],RED,m[3][4],BLUE,m[3][5],RED,m[3][6],BLUE,m[3][7],RED,m[
printf("          %s\\ / %s\\ / %s\\ / %s\\ / %s\\ / %s\\ / %s\\ / %s\\ /     \n ",RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED);
printf("             %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\     \n ",RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED);
printf("           5| %c |%s| %c |%s| %c |%s| %c |%s| %c |%s| %c |%s| %c |%s| %c | 5\n ",m[4][0],BLUE,m[4][1],RED,m[4][2],BLUE,m[4][3],RED,m[4][4],BLUE,m[4][5],RED,m[4][6],BLUE,m[4][7],RED
printf("            %s\\ / %s\\ / %s\\ / %s\\ / %s\\ / %s\\ / %s\\ / %s\\ /     \n ",RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED);
printf("               %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\     \n ",RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED);
printf("             6| %c |%s| %c |%s| %c |%s| %c |%s| %c |%s| %c |%s| %c |%s| %c | 6\n ",m[5][0],BLUE,m[5][1],RED,m[5][2],BLUE,m[5][3],RED,m[5][4],BLUE,m[5][5],RED,m[5][6],BLUE,m[5][7],
printf("              %s\\ / %s\\ / %s\\ / %s\\ / %s\\ / %s\\ / %s\\ / %s\\ /     \n ",RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED);
printf("                 %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\     \n ",RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED);
printf("               7| %c |%s| %c |%s| %c |%s| %c |%s| %c |%s| %c |%s| %c |%s| %c | 7\n ",m[6][0],BLUE,m[6][1],RED,m[6][2],BLUE,m[6][3],RED,m[6][4],BLUE,m[6][5],RED,m[6][6],BLUE,m[6][
printf("                %s\\ / %s\\ / %s\\ / %s\\ / %s\\ / %s\\ / %s\\ / %s\\ /   \n ",RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED);
printf("                   %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\     \n ",RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED);
printf("                 8| %c |%s| %c |%s| %c |%s| %c |%s| %c |%s| %c |%s| %c |%s| %c | 8\n ",m[7][0],BLUE,m[7][1],RED,m[7][2],BLUE,m[7][3],RED,m[7][4],BLUE,m[7][5],RED,m[7][6],BLUE,m[
printf("                  %s\\ / %s\\ / %s\\ / %s\\ / %s\\ / %s\\ / %s\\ / %s\\ /     \n ",RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED);
printf("                     %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\  %s/ \\     \n ",RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED);
printf("                   9| %c |%s| %c |%s| %c |%s| %c |%s| %c |%s| %c |%s| %c |%s| %c | 9\n ",m[8][0],BLUE,m[8][1],RED,m[8][2],BLUE,m[8][3],RED,m[8][4],BLUE,m[8][5],RED,m[8][6],BLUE
printf("                    %s\\ / %s\\ / %s\\ / %s\\ / %s\\ / %s\\ / %s\\ / %s\\ /     \n ",RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED,BLUE,RED);
```

```
45
46  int chek_rowB(char m[9][9]) {
47
48      int cmp=0;
49          for (int j = 0; j < 9; j++){
50          cmp=0;
51          for (int i = 0; i < 9; i++) {
52          if (m[j][i] == 'B') {
53          cmp++;
54          }
55      }if(cmp==9){break;}
56          } return cmp == 9;
57  }
58
59
60  int chek_linB(char m[9][9]) {
61
62      int cmp=0;
63      for (int j = 0; j < 9; j++){
64          cmp=0;
65          for (int i = 0; i < 9; i++) {
66          if (m[i][j] == 'B') {
67              cmp++;
68          }
69      } if(cmp==9){break;}
70
71      } return cmp == 9;
72
73  }
74
75
76  int chek_rowN(char m[9][9]) {
77
78      int cmp = 0;
79      for (int j = 0; j < 9; j++){
80              cmp = 0;
81          for (int i = 0; i < 9; i++) {
82          if (m[j][i] == 'N') {
83              cmp++;
84          }
```

```
85          } if(cmp==9){break;}
86
87          } return cmp == 9;
88      }
89
90
91      int chek_linN(char m[9][9]) {
92
93          int cmp = 0;
94          for (int j = 0; j < 9; j++){
95              cmp = 0;
96              for (int i = 0; i < 9; i++) {
97              if (m[i][j] == 'N') {
98                  cmp++;
99              }
100         } if(cmp==9){break;}
101         } return cmp == 9;
102     }
103
104
105     int chek_diagB(char m[9][9]) {
106         int cmp = 0;
107         for (int i = 0; i < 9; i++) {
108             if (m[i][i] == 'B') {
109                 cmp++;
110             }
111         }
112         return cmp == 9;
113     }
114
115     int chek_diagN(char m[9][9]) {
116         int cmp = 0;
117         for (int i = 0; i < 9; i++) {
118             if (m[i][i] == 'N') {
119                 cmp++;
120             }
121         }
122         return cmp == 9;
123     }
```

```c
int chek_sec_diagN(char m[9][9]) {
    int cmp = 0;
    for (int i = 0; i < 9; i++) {
        if (m[i][8 - i] == 'N') {
            cmp++;
        }
    }
    return cmp == 9;
}

int chek_sec_diagB(char m[9][9]) {
    int cmp = 0;
    for (int i = 0; i < 9; i++) {
        if (m[i][8 - i] == 'B') {
            cmp++;
        }
    }
    return cmp == 9;
}
void actualiser(int n){
    sleep(n);
    system("cls");
}
```

```c
void affichage(){
    printf("%s----------%s_  _  %s____  %s_  _%s---------- \n",YELLOW,RED,GREEN,BLUE,YELLOW);
    printf("%s----------%s| |_| |%s|__| %s\\ \\/ /%s---------- \n",YELLOW,RED,GREEN,BLUE,YELLOW);
    printf("----------%s| | | |%s|    %s\\ \\/ /%s---------- \n",RED,GREEN,BLUE,YELLOW);
    printf("----------%s|_|_|_|%s|___ %s\\ \\/ /%s---------- \n",RED,GREEN,BLUE,YELLOW);
    printf("----------%s|_|_|_|%s|__| %s/ /\\ \\%s---------- \n ",RED,GREEN,BLUE,YELLOW);
    printf("----------%s| | | |%s|    %s/ /  \\ \\%s---------- \n",RED,GREEN,BLUE,YELLOW);
    printf("----------%s| | | |%s|___ %s/ /    \\ \\%s---------- \n",RED,GREEN,BLUE,YELLOW);
    printf("----------%s|_| |_|%s|____|%s/_/      \\_\\%s---------- \n",RED,GREEN,BLUE,YELLOW);
```

```
164 char MODE_IA_1(char m[9][9]){
165       int number_PL,number_IA;
166       int k=0;
167       char Letter_IA,Letter_PL;
168       char name_player[20];
169       printf("%s                                                ~~~~~~~~~~~~~~~~~~~~~~~~~~",GREEN);
170       printf("\n                                              %s|| enter your name :      ||\n",YELLOW);
171       printf("%s                                               ~~~~~~~~~~~~~~~~~~~~~~~~~~%s\n",GREEN,RESET);
172       printf(" %s                                                        ",RED);scanf("%s",&name_player);
173
174
175         while(k<81){
176
177           simo:
178          srand(time(NULL));
179           number_IA=rand()%10;
180           Letter_IA='A' + rand() % 9;
181           Letter_IA=Letter_IA-'A';
182           if((m[number_IA-1][Letter_IA]=='B')||(m[number_IA-1][Letter_IA]=='N')){ goto simo;}
183
184           m[number_IA-1][Letter_IA]='N';
185           printf("\nAI's Move:\n");
186           hex(m);
187           if (chek_rowN(m) || chek_linN(m)||chek_diagN(m)||chek_sec_diagN(m)) {
188           printf("\nIA wins!\n");
189           break;}
```

```
191         kouzoud:
192         printf("\n %sPlayer %s 's turn to play :\n ",GREEN,name_player);
193         printf("%s enter the letter\n",YELLOW);
194         scanf(" %c", &Letter_PL);
195         printf("%s enter the number\n",YELLOW);
196         scanf(" %d",&number_PL);
197         if(number_PL==0||Letter_PL=='Z'){printf("%sEND OF GAME!%s ",RED,RESET);exit(0);}
198         if ((number_PL < 1 || number_PL > 9)||(Letter_PL<'A'||Letter_PL>'I')) {
199             printf("%s Invalid input. Please enter valid values between %s 1 and 9,%s and letters between %s A and I\n",YELLOW,RED,YELLOW,RED);
200             goto kouzoud;
201             }
202             Letter_PL = Letter_PL - 'A';
203             if((m[number_PL-1][Letter_PL]=='B')||(m[number_PL-1][Letter_PL]=='N')){printf("%s is pline",RED);  goto kouzoud;}
204             m[number_PL-1][Letter_PL]='B';
205             hex(m);
206             if (chek_rowB(m) || chek_linB(m)||chek_diagB(m)||chek_sec_diagB(m)) {
207             printf("\n %s congratulation %s %s %s you are the winner!\n%s",RED,GREEN,name_player,RED,RESET);
208             break;}
209
210
211
212 k++;
213
214 actualiser(0);
215     }
216
217
218 return 0;
219 }
```

```c
char MODE_TWO_PLAYERS(char m[9][9]){
int x=0;
srand(time(NULL));
int number_player2,number_player1;
int random_start= rand() % 2;
char letter_player1,letter_player2;
char name_player1[20];
char name_player2[20];
printf("                                    %s --------------------\n",GREEN);
printf("                                    %s name of first player: \n",YELLOW);
printf("                                    %s --------------------\n",GREEN);
printf("                                               ");
scanf(" %s",&name_player1);
printf("                                    %s ********************\n",Magenta);
printf("                                    %s --------------------\n",GREEN);
printf("                                    %s name of second player: \n",YELLOW);
printf("                                    %s --------------------\n",GREEN);
printf("                                               ");
scanf(" %s",&name_player2);
//hex(m);
 if (random_start == 0) {
        printf("\n%s goes first!\n", name_player1);
    } else {
        printf("\n%s goes first!\n", name_player2);
    }
        while(x<81){
        hex(m);
        if (random_start == 0){
            hajar:
        printf("\n %s Player %s 's turn to play :\n ",GREEN,name_player1);
        printf("%s Enter the letter (A-I):\n",YELLOW);
        scanf(" %s", &letter_player1);
        printf("%s Enter the number (1-9):\n",YELLOW);
        scanf(" %d",&number_player1);
        if(number_player1==0||letter_player1=='Z'){printf("%sEND OF GAME!%s ",RED,RESET);exit(0);}
        if ((number_player1 < 1 || number_player1 > 9)||(letter_player1<'A' || letter_player1>'I')) {
            printf("%s Invalid input. Please enter valid values between %s 1 and 9,%s and letters between %s A and I\n",YELLOW,RED,YELLOW,RED);
            goto hajar;
```

```
            }
            letter_player1 = letter_player1 - 'A';
            if((m[number_player1-1][letter_player1]=='N')||(m[number_player1-1][letter_player1]=='B')){printf("is pline");goto hajar;}
            m[number_player1-1][letter_player1]='N';
            //hex(m);
            if (chek_rowN(m) || chek_linN(m)||chek_diagN(m)||chek_sec_diagN(m)) {
            printf("\n %s congratulation %s %s  %s you are the winner!\n%s",RED,GREEN,name_player1,RED,RESET);
            break;}
        }

         else{
        oulabasse:
        printf("\n %s Player %s's turn to play\n",GREEN,name_player2);
        printf("%s Enter the letter (A-I):\n",YELLOW);
        scanf(" %c", &letter_player2);
        printf("%s Enter the number (1-9):\n",YELLOW);
        scanf(" %d",&number_player2);
        if(number_player2==0||letter_player2=='Z'){exit(0);}
        if ((number_player2 < 1 || number_player2 > 9)||(letter_player2<'A' || letter_player2>'I')) {
            printf("%s Invalid input. Please enter valid values between %s 1 and 9,%s and letters between %s A and I\n",YELLOW,RED,YELLOW,RED);
            goto oulabasse;
            }
            letter_player2 = letter_player2 - 'A';
            if((m[number_player2-1][letter_player2]=='B')||(m[number_player2-1][letter_player2]=='N')){printf("is pline");  goto oulabasse;}
            m[number_player2-1][letter_player2]='B';
            if (chek_rowB(m) || chek_linB(m)||chek_diagB(m)||chek_sec_diagB(m)) {
            printf("\n %s congratulation %s %s  %s you are the winner!\n%s",RED,GREEN,name_player2,RED,RESET);
            break;}
         }


x++;
random_start = (random_start + 1) % 2;
actualiser(0);
}

}
```

```c
void menu(){
    system("color 02");
    char m[9][9];
    char ok;
    for(int i=0;i<9;i++){
    for(int j=0;j<9;j++){m[i][j]=' ';}}
     int i,j,n;
     affichage();
     printf("     \n\n\n \n                                        %s_____________________________________\n",RED);
     printf("                                                %s\\_____|_____|_____|_____|_____|___/\n",RED);
     do{printf("                                                  %s\\ \\          %s menu:                  %s\\ \\ \n",GREEN,BLUE,GREEN);
     printf("                                                  %s\\ \\          %s ~~~~~                  %s\\ \\\n",GREEN,BLUE,GREEN);
     printf("                                                  %s\\ \\          %s  1.Play                %s\\ \\\n",GREEN,BLUE,GREEN);
     printf("                                                  %s\\ \\          %s 2.rules of game        %s\\ \\\n",GREEN,BLUE,GREEN);
     printf("                                                  %s\\ \\          %s 3.exit                 %s\\ \\\n",GREEN,BLUE,GREEN);
     printf("                                                  %s\\ \\                                    \\ \\\n",GREEN);
     printf("                                                  %s\\ \\                                    \\ \\\n",GREEN);
     printf("                                                  %s\\ \\                                    \\ \\\n",GREEN);
     printf("                                                    \t\t\t");scanf("%d",&i);
     }while((i!=1)&&(i!=2)&&(i!=3));
     actualiser(1);

     if(i==1){
          printf("\n\n");
        printf("                        %s||%s**************************************************%s||\n",GREEN,BLUE,GREEN);
          do{
        printf("                        %s||                  %sChoose a game mode :          %s||          \n",GREEN,RED,GREEN);
        printf("                        %s||%s~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~%s||\n",GREEN,BLUE,GREEN);
        printf("                        %s||                  %s 1. Mode IA                   %s||\n",GREEN,RED,GREEN);
        printf("                        %s||                  %s 2. Mode tow players          %s||  \n",GREEN,RED,GREEN);
        printf("                        %s||%s--------------------------------------------------%s||\n",GREEN,BLUE,GREEN);
        printf("                              \t\t\t");scanf("%d",&j);
          }while((j!=1)&&(j!=2));
     actualiser(1);
```

```
if(j==1){
        printf("\n\n");
        printf("                    %s//%s*****************************************************%s//\n",RED,BLUE,RED);
      do{
        printf("                    %s//%s        You have chosen the Mode IA.                 %s//\n",RED,GREEN,RED);
        printf("                    %s//%s-------------------------------------------------%s// \n",RED,GREEN,RED);
        printf("                    %s//%s        tChoose the difficulty level :               %s//\n",RED,GREEN,RED);
        printf("                    %s//%s~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~%s//\n",RED,GREEN,RED);
        printf("                    %s//%s          1. Normal AI mode                          %s//\n",RED,BLUE,RED);
        printf("                    %s//%s          2. Advanced AI mode                        %s//\n",RED,BLUE,RED);
        printf("                              \t\t\t");scanf("%d",&n);
      }while((n!=1)&&(n!=2));
      actualiser(1);
        if(n==1){
        printf("                                        %s ~~~~~~~~~~~~~~~~~~~~~~~~\n",GREEN);
        printf("                                       %s ||Normal AI mode apply:   ||\n",YELLOW);
        printf("                                        %s ~~~~~~~~~~~~~~~~~~~~~~~~\n",GREEN);
        printf("                                         %s************************\n",BLUE);
        MODE_IA_1(m);}

        else if(n==2){printf("~~~~~~~~~~~~~~~~~~~~~~~~~~\n");
        printf("Advanced AI mode apply: \n");
        printf("~~~~~~~~~~~~~~~~~~~~~~~~~~\n");}
        }
else if(j==2){
printf("                                  %s~~~~~~~~~~~~~~~~~~~\n",RED);
printf("                                 %s Two player mode apply: \n",YELLOW);
printf("                                %s  ~~~~~~~~~~~~~~~~~~~\n",RED);
printf("                                 %s ********************\n",Magenta);
```

```
        MODE_TWO_PLAYERS(m);}
    }
    else if(i==2){
    printf("%s Here are the basic rules of the Hex game:\n\n",Magenta);
    printf("%s #Objective of the game: The first player to create a continuous connection between their two opposite sides of the
    printf("#Board: The board is composed of hexagonal cells. The number of cells can vary, but a common board is composed of 9x9
    printf("#Turn of play: Players take turns. Each player places a stone on an empty cell of the board during their turn.\n\n");
    printf("#Connection: Players attempt to create a continuous connection between their opposite sides of the board by placing th
    printf("#Blocking: Players can block their opponent's connection attempts by placing their stones strategically.\n\n");
    printf("#End of the game: The game ends as soon as a player succeeds in creating a connection between their opposite sides. Th
    printf("#Hex is a game that is simple in appearance, but it can be complex strategically. The Hex theorem guarantees that ther
    printf("# THE letters always capitals between A and I and numbers between 1 and 9 \n");
    printf("\n#NB:you can exit of game by enter 0 or Z. ");
    printf("\n%s------------------------\n",YELLOW);
    printf("%s if you read it click OK!\n",RED);
    printf("%s------------------------\n",YELLOW);
    scanf("%s",&ok);
    actualiser(1);
   menu();

    }
    else if (i == 3) {
        printf("%s\n Exiting the program.\n\n%s",RED,RESET);
        exit(0);
    }

}



int main(){

menu();


}
```

Conclusion et perspectives:

Objectifs réalisés :

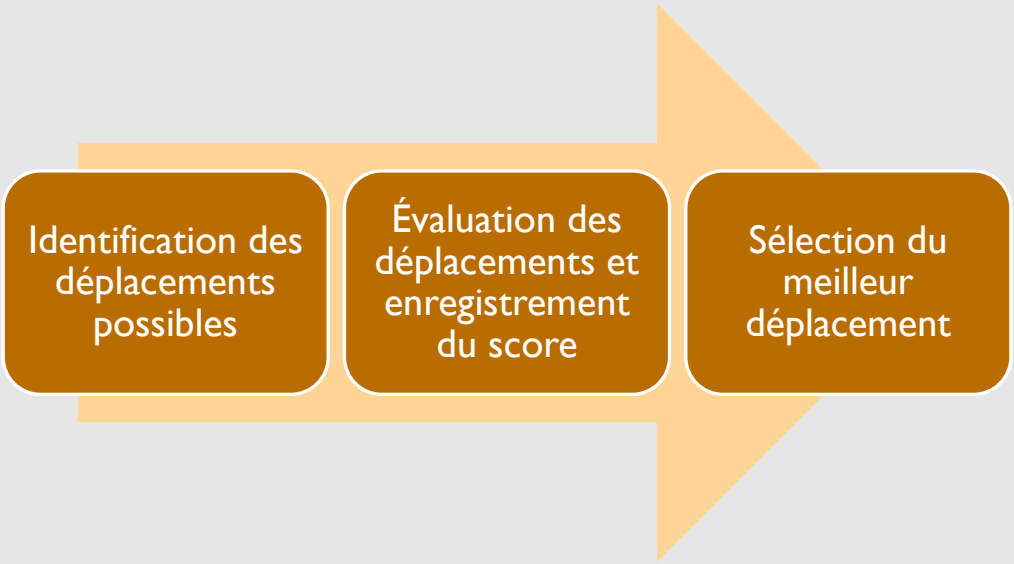
1. Affichage du plateau de jeu : On a réussi à afficher le plateau de jeu hexagonal avec les bonnes dimensions et les cellules vides représentées par des espaces.
2. Mode IA : On a pu développé un mode de jeu où l'ordinateur joue contre le joueur. L'IA peut prendre des décisions en plaçant la letter ‘N’ sur le plateau.
3. Mode Deux Joueurs : On a pu implémenté un mode où deux joueurs humains peuvent s'affronter en prenant alternativement des tours pour placer la letter ‘N’ ou ‘B’.
4. Validation des entrées utilisateur : Nous avons géré les entrées utilisateur en vérifiant si elles sont valides et en demandant à l'utilisateur de saisir à nouveau si nécessaire.
5. Détection de la victoire : Nous avons pu inclure la logique pour détecter lorsque l'un des joueurs a gagné en reliant deux côtés différents du plateau.

6. Sortie du jeu : Nous avons pu ajouter la possibilité pour les joueurs de quitter le jeu en entrant une commande spécifique (comme 0 ou Z).
7. Affichage des règles du jeu : Nous avons inclus une fonctionnalité qui permet aux joueurs de consulter les règles du jeu Hex.
8. Interface utilisateur agréable : Nous avons mis en place une interface utilisateur conviviale avec des couleurs attrayantes et des mises en forme appropriées pour rendre l'expérience de jeu plus agréable.
9. Gestion de l'écran : Nous avons ajouté des fonctionnalités pour effacer et actualiser l'écran afin de fournir une meilleure expérience utilisateur pendant le jeu.

Défis rencontrés:

En résumé, bien que nous ayons réussi à implémenter les modes AI normal de notre jeu avec succès, nous avons été confrontés à des difficultés lors de la mise en œuvre du mode AI avancé et de la fonction de test. Le mode AI avancé, qui aurait enrichi l'expérience de jeu en offrant une intelligence artificielle plus sophistiquée, n'a malheureusement pas pu être achevé en raison de contraintes de temps et de ressources. De plus, nous avons constaté des limitations dans la fonction de test, où seules les lignes droites pouvaient être suivies, alors que les cas où le trajet passe entre deux lettres similaires n'étaient pas correctement comptabilisés.

Réflexion sur l'IA avancé:



voici notre conception pour l'IA avancée dans notre jeu Hex, basée sur les idées suivantes :

Parcours du tableau :

Identification des déplacements possibles :

Pour chaque case , l'algorithme identifie tous les déplacements possibles. Ces déplacements peuvent inclure :

- pointage sur une case où il peut toucher la case remplie par ladversair.
- pointage sur une case vers une case limite .
- poingage simple pour lier.

Évaluation des mouvements et selection du Meilleur choix :

- Pour chaque case possible l'IA simule son exécution et enregistre le score associé. Le score est déterminé par le nombre de conditions de vérifiées.

Cette approche permet à l'IA de prendre des décisions informées en analysant les déplacements possibles et en choisissant celui qui maximise ses chances de succès.

المدرسة الحسنية للأشغال العمومية
ECOLE HASSANIA DES TRAVAUX PUBLICS

Remerciement :

Nous tenons à exprimer notre sincère gratitude envers Madame Cherrabi, notre professeur encadrante, pour son soutien et ses précieux conseils tout au long de ce projet. Sa guidance éclairée a été d'une importance capitale pour surmonter les obstacles rencontrés et pour mener à bien cette étude. Son expertise et son dévouement ont grandement enrichi notre expérience et nous ont permis de progresser dans notre compréhension du langage C. Nous lui sommes sincèrement reconnaissants pour son engagement et sa disponibilité.

Ressources d'apprentissage :

Tutoriel de jeu Hex sur YouTube

- **Lien** : https://www.youtube.com/watch?v=NL9imSjJi7o&list=PLNefH6S6myiMPc5-9XAeUPfxe21Hxso-s
- **Description** : Ce tutoriel vidéo m'a donné un aperçu des règles du jeu Hex et m'a aidé à comprendre les bases du développement de mon programme.

Générateur de Texte Stylisé :

- Lors de la conception de l'interface utilisateur du jeu Hex, nous avons utilisé un générateur de texte stylisé pour créer des titres et des menus attrayants. Cet outil en ligne nous a permis d'expérimenter avec différentes polices et styles de texte pour améliorer la présentation visuelle du jeu.
- Lien : http://patorjk.com/software/taag/#p=display&f=Graffiti&t=Type%20Something
- Description : Cet outil en ligne propose une large gamme de polices créatives pour styliser du texte. Il offre une interface conviviale où les utilisateurs peuvent saisir leur texte et prévisualiser instantanément le rendu dans différentes polices. Une fois satisfait du résultat, le texte stylisé peut être copié et collé dans d'autres applications ou documents.

Tutoriel YouTube sur le Développement de Jeux en C :

- Pendant la phase de développement du jeu Hex, nous avons trouvé un tutoriel vidéo sur YouTube qui nous a guidés à travers les principes de base du développement de jeux en langage C. Ce tutoriel nous a aidés à comprendre les concepts fondamentaux de la programmation de jeux, tels que la gestion de la logique du jeu, les boucles de jeu et la manipulation des données.
- Titre : **Coder des jeux en C**
- Chaîne YouTube : **Gamecodeur - L'école de jeux vidéo en Ligne**
- Lien :

https://www.youtube.com/watch?v=2gLaujOAnD8

- Description : Ce tutoriel vidéo fournit une introduction complète au développement de jeux en utilisant le langage C. Il couvre des sujets tels que la gestion des entrées utilisateur, le rendu graphique simple, la logique du jeu et la gestion de la mémoire. Les exemples pratiques et les explications claires ont été extrêmement bénéfiques pour notre équipe lors de la création du jeu Hex.

المدرسة الحسنية للأشغال العمومية
ECOLE HASSANIA DES TRAVAUX PUBLICS