A dark blue vertical bar runs down the left side of the page. A blue arrow points to the right from this bar, containing the date.

27/05/2022

Jeu De Dame

Projet du cour d'informatique

1	<i>Introduction</i>	2
2	<i>Le jeu de Dames</i>	2
2.1	Les règles du jeu	2
2.2	Le but du jeu	2
3	<i>Etapas de mon Jeu</i>	3
3.1	Construction de la matrice	3
3.2	Affichage de la matrice	4
3.3	Placement des pions blancs	5
3.4	Placement des pions noirs	6
3.5	Avancement des pions blancs	7
3.6	Avancement des pions noirs	12
4	<i>Team Gantt</i>	17
4.1	Lien	17
4.2	Diagramme	17
5	<i>lien vers d'autres outils</i>	18
5.1	Trello	18
5.2	GitHub	18
6	<i>Conclusion</i>	19
7	<i>Bibliographie</i>	19
7.1	Sources utilisées pour les règles et le buts du jeu	19
7.2	Sources utilisées pour le codage	19

1 Introduction

Lors de la réalisation de ce projet, j'ai appris et compris plus concrètement ce qu'est réellement la programmation, plus précisément en langage C#. D'abord, ça m'a permis de savoir quelles études supérieures je dois poursuivre après mes études secondaires. De plus, j'ai appris aussi à travailler de façon autonome du début jusqu'à la fin. Grâce notamment à des outils tels que « TeamGantt » mais aussi « Trello ». Ensuite, j'ai choisi ce jeu car dans mon enfance, j'ai énormément joué au jeu de dame. Enfin, c'est un jeu qui me parlait vraiment. Je connais les règles par cœur, ce qui m'a fait gagner du temps pour la réalisation de ce jeu. En conclusion, j'étais vraiment motivé à travailler dessus. Je vous invite vivement à lire la suite pour réellement comprendre comment le jeu a pu être réalisé. Bonne lecture.

2 Le jeu de Dames

2.1 Les règles du jeu

Le jeu de dames se joue en un contre un, sur un plateau de 10x10 et avec 20 pions par joueurs. Les pions sont placés sur les cases foncées des 4 premières rangées de part et d'autre du plateau. Les joueurs jouent chacun à leur tour. Par convention, le joueur avec les pions blanc commence toujours. Il a le droit d'avancer le pion uniquement en diagonale soit à droite, soit à gauche. Afin de marquer des points et gagner le joueur doit manger les pions du joueur adverse. Le pion du joueur voulant manger le pion de l'autre joueur, doit passer par-dessus le pion de l'autre joueur seulement si celui-ci est diagonalement collé en sachant qu'il est obligatoire que la case derrière soit libre. Si c'est le cas, le joueur enlève le pion adverse du damier.

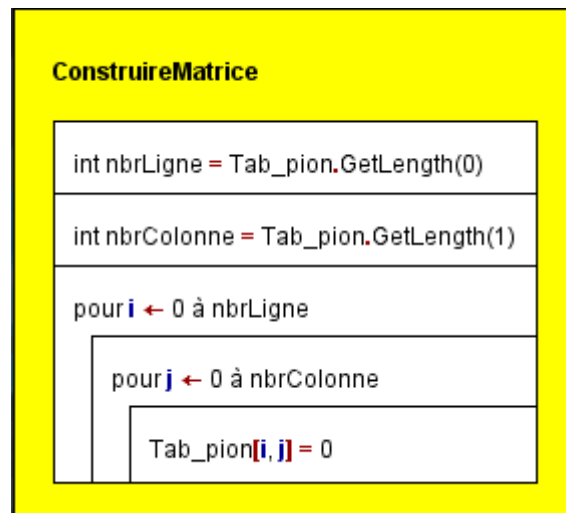
2.2 Le but du jeu

Le but du jeu est de manger tous les pions adverses pour qu'il ne lui en reste plus un seul sur le damier. Si un joueur ne peut plus bouger, même s'il lui reste des pions, il perd automatiquement la partie. Chaque pion peut se déplacer d'une case vers l'avant en diagonale. Un pion arrivant sur la dernière rangée et s'y arrêtant est promu en «dame». Il est alors doublé (on pose dessus un deuxième pion de sa couleur). La dame se déplace sur une même diagonale d'autant de cases qu'elle le désire, en avant et en arrière. Mais attention, Si vous pouvez manger un pion adverse alors vous devez absolument le manger! Et vous ne pouvez pas avancer avec un autre pion.

3 Etapes de mon Jeu

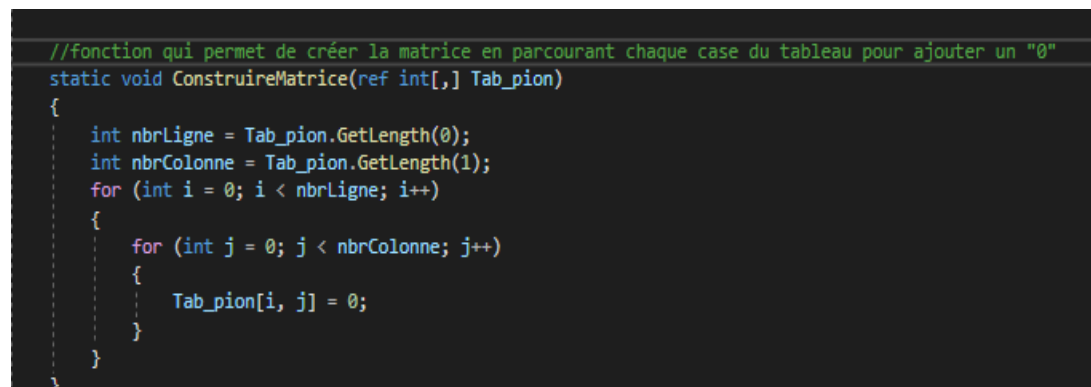
3.1 *Construction de la matrice*

3.1.1 Analyse



3.1.2 Le programme

```
//fonction qui permet de créer la matrice en parcourant chaque case du tableau pour ajouter un "0"
static void ConstruireMatrice(ref int[,] Tab_pion)
{
    int nbrLigne = Tab_pion.GetLength(0);
    int nbrColonne = Tab_pion.GetLength(1);
    for (int i = 0; i < nbrLigne; i++)
    {
        for (int j = 0; j < nbrColonne; j++)
        {
            Tab_pion[i, j] = 0;
        }
    }
}
```

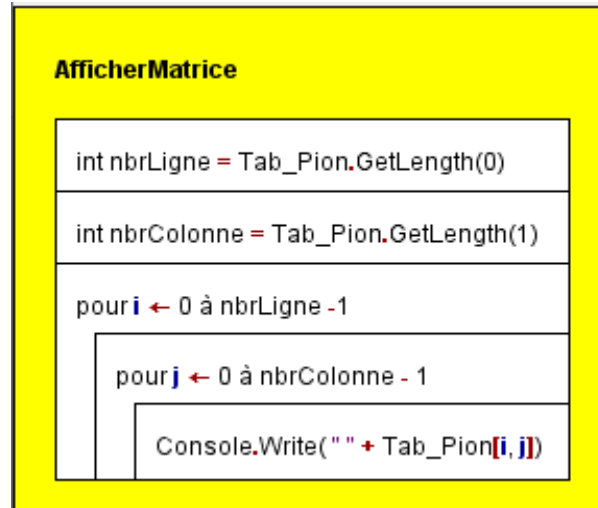


3.1.3 Explications du programme

La fonction ci-dessus va faire en sorte de parcourir toutes les cases du tableau et va venir y ajouter un « 0 » dans chaque case. Le 0 correspond aux cases du tableau.

3.2 Affichage de la matrice

3.2.1 Analyse



3.2.2 Le programme

```
//fonction qui permet de parcourir chaque case du tableau afin de les afficher et pouvoir ressortir tout le tableau
static void AfficherMatrice(int[,] Tab_Pion)
{
    int nbrLigne = Tab_Pion.GetLength(0);
    int nbrColonne = Tab_Pion.GetLength(1);

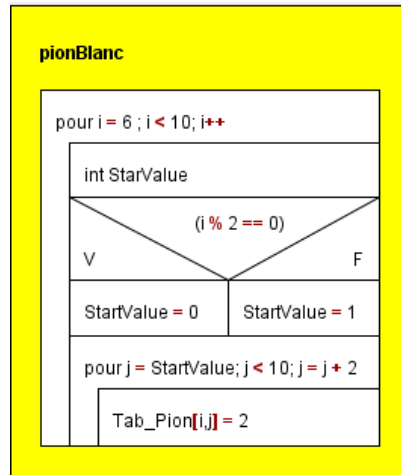
    for (int i = 0; i < nbrLigne; i++)
    {
        for (int j = 0; j < nbrColonne; j++)
        {
            Console.Write(" " + Tab_Pion[i, j]);
        }
        Console.WriteLine();
    }
}
```

3.2.3 Explications du programme

La fonction ci-dessus va faire en sorte de parcourir toutes les cases du tableau et va venir y récupérer l'information dans la case. Pour l'afficher dans le « cmd » et toutes les informations du tableau mis ensemble vont former le tableau. Dans mon cas un tableau rempli de « 0 »

3.3 Placement des pions blancs

3.3.1 Analyse



3.3.2 Le programme

```
//fonction qui permet de placer les pions noirs sur le plateau de jeu
static void PionBlanc(ref int[,] Tab_Pion)
{
    //pour les ligne de 6 => 9
    for (int i = 6; i < 10; i++)
    {
        int StartValue;
        //modulo= paire ou impaire
        if (i % 2 == 0)
        {
            StartValue = 0;
        }
        else
        {
            StartValue = 1;
        }

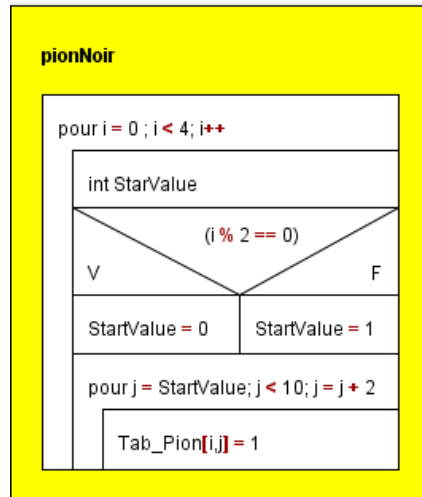
        for (int j = StartValue; j < 10; j = j + 2)
        {
            Tab_Pion[i, j] = 2;
        }
    }
}
```

3.3.3 Explications du programme

La fonction va placer les pions blancs (2) à l'aide de l'opération modulo afin de déterminer si c'est paire alors on mets 0 dans la case et si c'est impair alors on met 2, puis dans la boucle « for » viens se placer les « 2 », toutes les 2 cases d'intervalle.

3.4 Placement des pions noirs

3.4.1 Analyse



3.4.2 Le programme

```
//fonction qui permet de placer les pions blanc sur le plateau de jeu
static void PionNoir(ref int[,] Tab_Pion)
{
    //pour les ligne de 0 => 3
    for (int i = 0; i < 4; i++)
    {
        int StartValue;
        //modulo= paire ou impaire
        if (i % 2 == 0)
        {
            StartValue = 0;
        }

        else
        {
            StartValue = 1;
        }

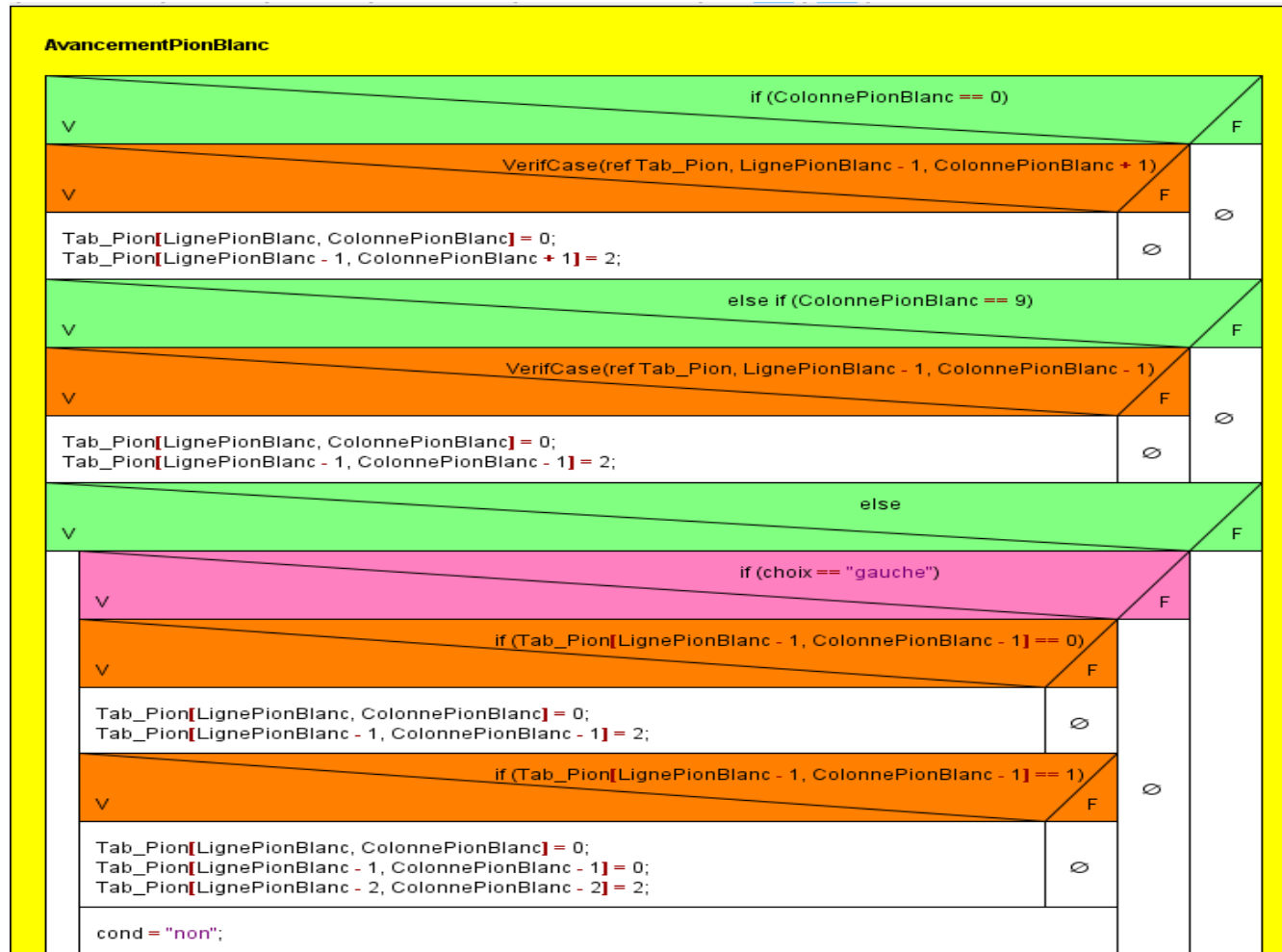
        for (int j = StartValue; j < 10; j = j + 2)
        {
            Tab_Pion[i, j] = 1;
        }
    }
}
```

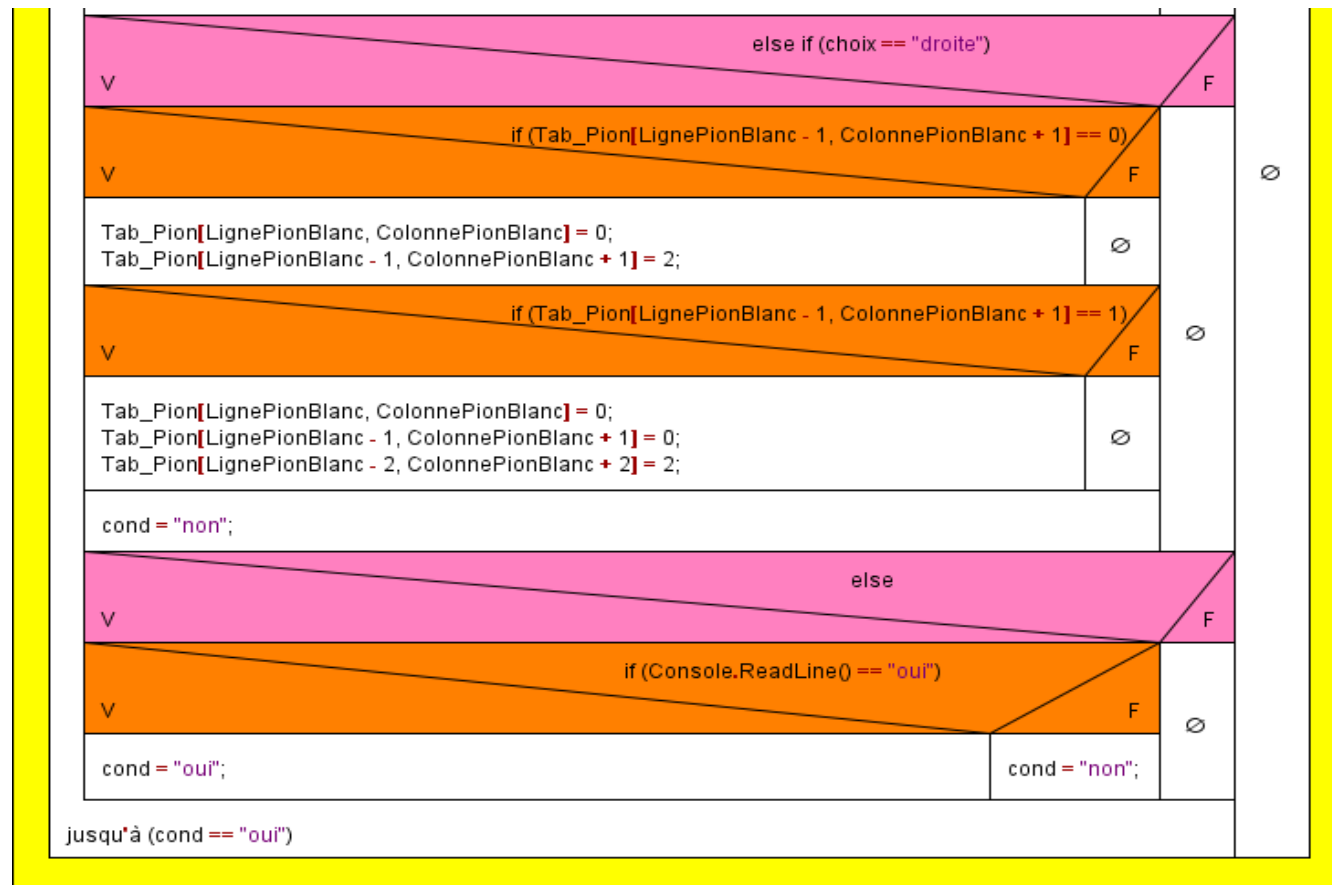
3.4.3 Explications du programme

La fonction va placer les pions blancs (1) à l'aide de l'opération modulo afin de déterminer si c'est paire, alors on mets 0 dans la case et si c'est impair, alors on mets 1, ensuite, dans la boucle « for » je viens placer les « 1 » toutes les 2 cases d'intervalle

3.5 Avancement des pions blancs

3.5.1 Analyse





3.5.2 Le programme

```
static void AvancementPionBlanc(ref int[,] Tab_Pion, string joueur1)
{
    int LignePionBlanc;
    int ColonnePionBlanc;

    Console.WriteLine(joueur1 + " " + "quelle numéro de ligne du pion que vous voulez avancer (Pion 2) ");
    LignePionBlanc = int.Parse(Console.ReadLine());

    Console.WriteLine(joueur1 + " " + "Quelle numéro de colonne du pion que vous voulez avancer (Pion 2) ");
    ColonnePionBlanc = int.Parse(Console.ReadLine());

    if (ColonnePionBlanc == 0)
    {
        //verif case haut droite
        if (VerifCase(ref Tab_Pion, LignePionBlanc - 1, ColonnePionBlanc + 1))
        {
            Tab_Pion[LignePionBlanc, ColonnePionBlanc] = 0;
            Tab_Pion[LignePionBlanc - 1, ColonnePionBlanc + 1] = 2;
        }
    }

    else if (ColonnePionBlanc == 9)
    {
        //verife case haut gauche
        if (VerifCase(ref Tab_Pion, LignePionBlanc - 1, ColonnePionBlanc - 1))
        {
            Tab_Pion[LignePionBlanc, ColonnePionBlanc] = 0;
            Tab_Pion[LignePionBlanc - 1, ColonnePionBlanc - 1] = 2;
        }
    }
}
```

```

else
{
    //Boucle pour rejouer si on se trompe sur le déplacement
    // string cond = "non";
    string cond = "oui";
    while (cond == "oui")
    {
        //demander si aller soit à Gauche ou soit à Droite sinon faux
        Console.WriteLine("Déplacer le pion gauche ou droite?");
        string choix = Console.ReadLine();

        if (choix == "gauche")
        {
            if (Tab_Pion[LignePionBlanc - 1, ColonnePionBlanc - 1] == 0)
            {
                Tab_Pion[LignePionBlanc, ColonnePionBlanc] = 0;
                Tab_Pion[LignePionBlanc - 1, ColonnePionBlanc - 1] = 2;
            }
            //Fonction pour manger les pion noir

            //si le pion noir avance sur une case avec un pion blanc dessus (2) alors le pion noir va directement passer 2 case plus loin
            if (Tab_Pion[LignePionBlanc - 1, ColonnePionBlanc - 1] == 1)
            {
                Tab_Pion[LignePionBlanc, ColonnePionBlanc] = 0;
                Tab_Pion[LignePionBlanc - 1, ColonnePionBlanc - 1] = 0;
                Tab_Pion[LignePionBlanc - 2, ColonnePionBlanc - 2] = 2;
            }
            cond = "non";
        }
        else if (choix == "droite")
        {
            if (Tab_Pion[LignePionBlanc - 1, ColonnePionBlanc + 1] == 0)
            {
                Tab_Pion[LignePionBlanc, ColonnePionBlanc] = 0;
                Tab_Pion[LignePionBlanc - 1, ColonnePionBlanc + 1] = 2;
            }
            //Fonction pour manger les pion noir

            //si le pion noir avance sur une case avec un pion blanc dessus (2) alors le pion noir va directement passer 2 case plus loin
            if (Tab_Pion[LignePionBlanc - 1, ColonnePionBlanc + 1] == 1)
            {
                Tab_Pion[LignePionBlanc, ColonnePionBlanc] = 0;
                Tab_Pion[LignePionBlanc - 1, ColonnePionBlanc + 1] = 0;
                Tab_Pion[LignePionBlanc - 2, ColonnePionBlanc + 2] = 2;
            }
            cond = "non";
        }
    }
}

```

```

else
{
    Console.WriteLine("commande erroné");
    Console.WriteLine("");

    Console.WriteLine("voulez-vous rejouer votre coup ? oui ou non ?");
    cond = Console.ReadLine();

    if (Console.ReadLine() == "oui")
    {
        cond = "oui";
    }
    else
    {
        cond = "non";
    }
}

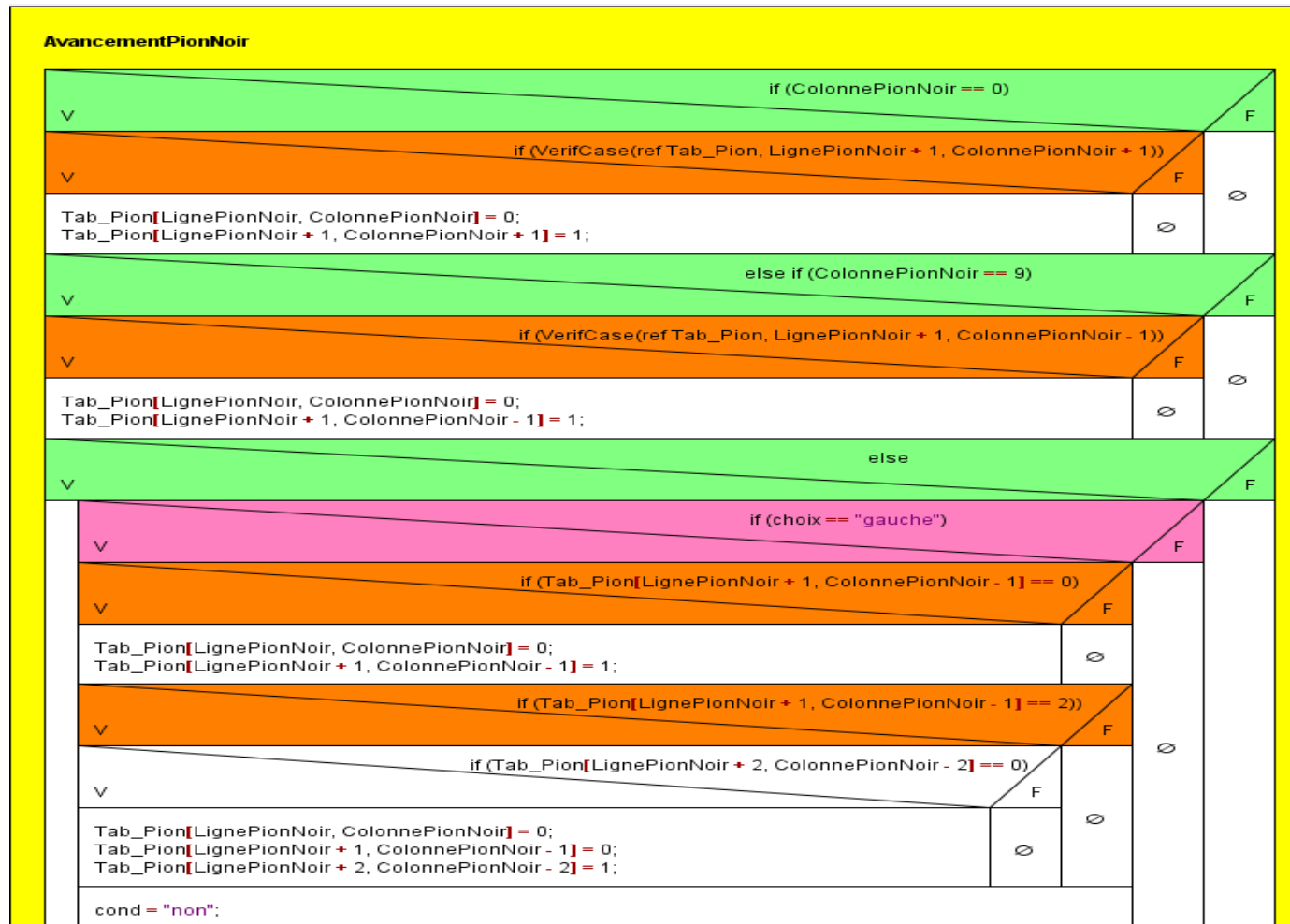
```

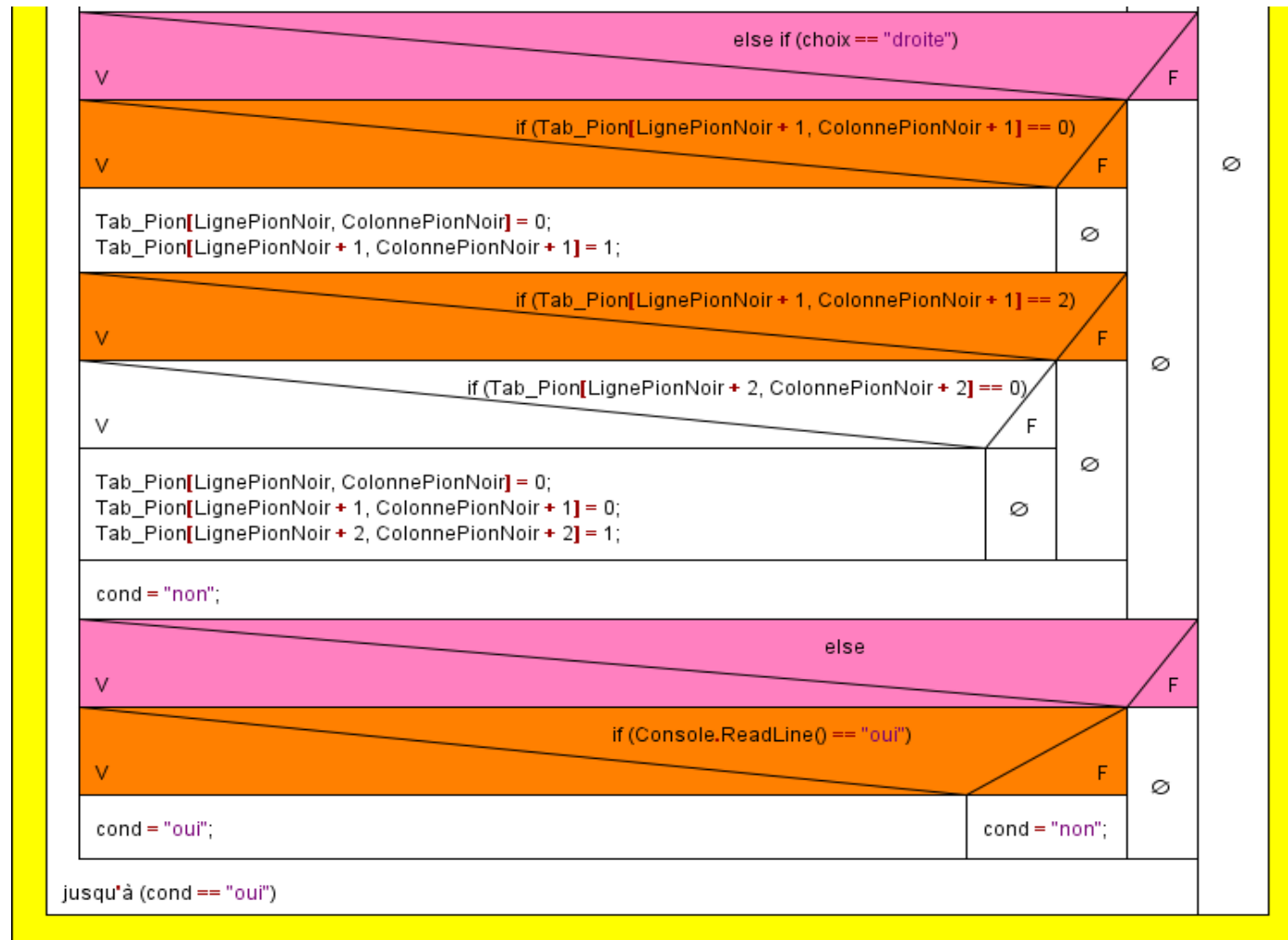
3.5.3 Explications du programme

L'avancement des pions blancs n'a pas que pour seule utilité de déplacer les pions sur le plateau mais aussi d'empêcher les pions de sortir du tableau, et la fonction contient aussi une fonction pour manger les pions noirs.

3.6 Avancement des pions noirs

3.6.1 Analyse





3.6.2 Le programme

```
static void AvancementPionNoir(ref int[,] Tab_Pion, string joueur2)
{
    int LignePionNoir;
    int ColonnePionNoir;

    Console.WriteLine(joueur2 + " " + "Quelle numéro de ligne du pion que vous voulez avancer (Pion 1) ");
    LignePionNoir = int.Parse(Console.ReadLine());

    Console.WriteLine(joueur2 + " " + "Quelle numéro de colonne du pion que vous voulez avancer (Pion 1) ");
    ColonnePionNoir = int.Parse(Console.ReadLine());

    if (ColonnePionNoir == 0)
    {
        //verif case haut droite
        if (VerifCase(ref Tab_Pion, LignePionNoir + 1, ColonnePionNoir + 1))
        {
            Tab_Pion[LignePionNoir, ColonnePionNoir] = 0;
            Tab_Pion[LignePionNoir + 1, ColonnePionNoir + 1] = 1;
        }
    }

    else if (ColonnePionNoir == 9)
    {
        //verife case haut gauche
        if (VerifCase(ref Tab_Pion, LignePionNoir + 1, ColonnePionNoir - 1))
        {
            Tab_Pion[LignePionNoir, ColonnePionNoir] = 0;
            Tab_Pion[LignePionNoir + 1, ColonnePionNoir - 1] = 1;
        }
    }
}
```

```

else
{
    string cond = "oui";
    while (cond == "oui")
    {
        //demander si aller soit à Gauche ou soit à Droite sinon faux
        Console.WriteLine("Déplacer le pion gauche ou droite?");
        string choix = Console.ReadLine();

        if (choix == "gauche")
        {
            if (Tab_Pion[LignePionNoir + 1, ColonnePionNoir - 1] == 0)
            {
                Tab_Pion[LignePionNoir, ColonnePionNoir] = 0;
                Tab_Pion[LignePionNoir + 1, ColonnePionNoir - 1] = 1;
            }
            //Fonction pour manger les pion blanc

            //si le pion noir avance sur une case avec un pion blanc dessus (2) alors le pion noir va directement passer 2 case plus loin
            if (Tab_Pion[LignePionNoir + 1, ColonnePionNoir - 1] == 2)
            {
                //ET si la case 2 ligne plus bas et 2 colonne à gauche est égale à 0 (alors je peux manger le pion adverse)
                if (Tab_Pion[LignePionNoir + 2, ColonnePionNoir - 2] == 0)
                {
                    Tab_Pion[LignePionNoir, ColonnePionNoir] = 0;
                    Tab_Pion[LignePionNoir + 1, ColonnePionNoir - 1] = 0;
                    Tab_Pion[LignePionNoir + 2, ColonnePionNoir - 2] = 1;
                }
            }
            cond = "non";
        }
        else if (choix == "droite")
        {
            if (Tab_Pion[LignePionNoir + 1, ColonnePionNoir + 1] == 0)
            {
                Tab_Pion[LignePionNoir, ColonnePionNoir] = 0;
                Tab_Pion[LignePionNoir + 1, ColonnePionNoir + 1] = 1;
            }
            //Fonction pour manger les pion blanc

            //si le pion noir avance sur une case avec un pion blanc dessus (2) alors le pion noir va directement passer 2 case plus loin
            if (Tab_Pion[LignePionNoir + 1, ColonnePionNoir + 1] == 2)
            {
                //sinon si la case 2 lignes plus bas et 2 colonne à droite est égale à 0 (alors je peux manger le pion adverse)
                if (Tab_Pion[LignePionNoir + 2, ColonnePionNoir + 2] == 0)
                {
                    Tab_Pion[LignePionNoir, ColonnePionNoir] = 0;
                    Tab_Pion[LignePionNoir + 1, ColonnePionNoir + 1] = 0;
                    Tab_Pion[LignePionNoir + 2, ColonnePionNoir + 2] = 1;
                }
            }
            cond = "non";
        }
    }
}

```



```

else
{
    Console.WriteLine("commande erroné");
    Console.WriteLine("");

    Console.WriteLine("voulez-vous rejouer votre coup ? oui ou non ?");
    cond = Console.ReadLine();

    if (Console.ReadLine() == "oui")
    {
        cond = "oui";
    }
    else
    {
        cond = "non";
    }
}

```

3.6.3 Explications du programme

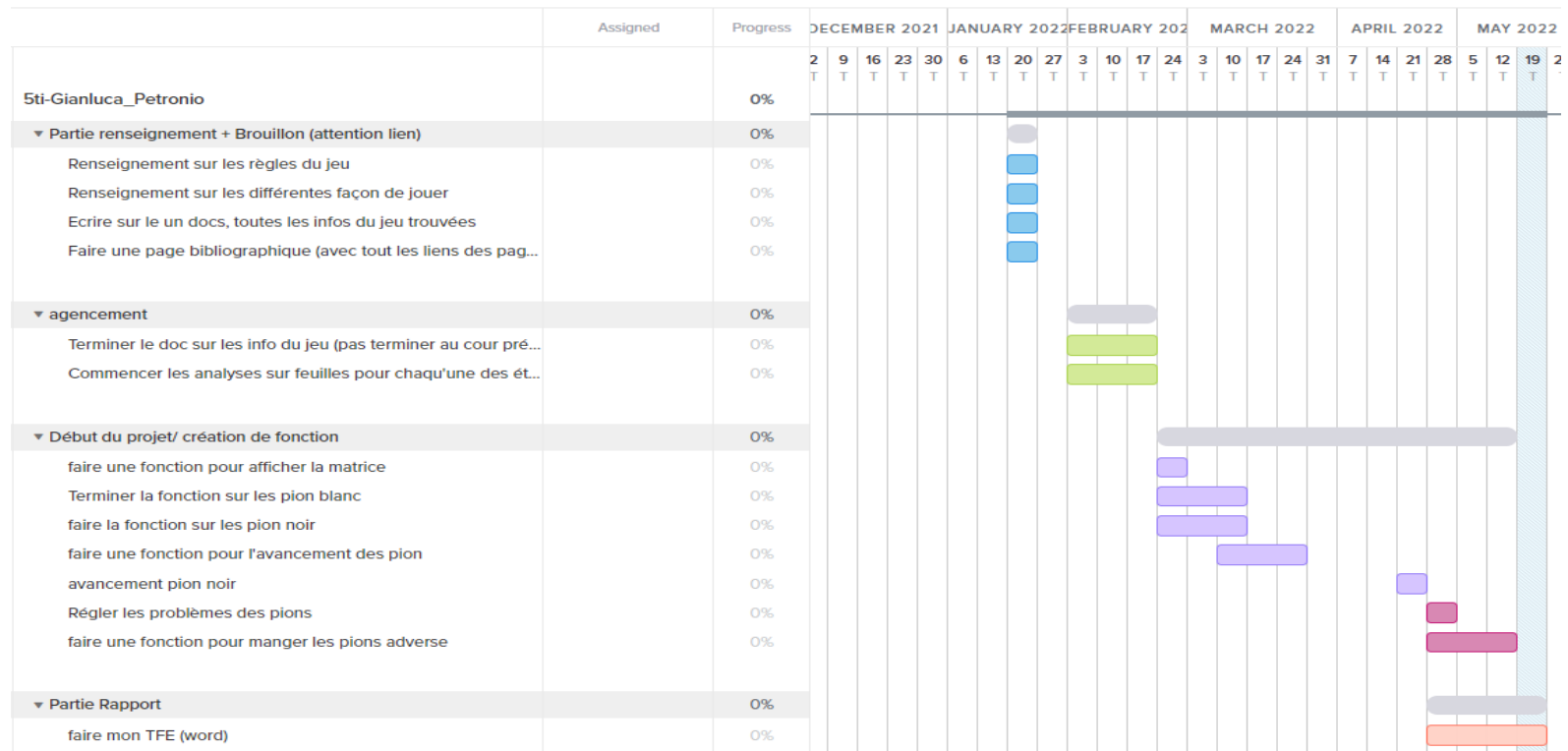
L'avancement des pions noirs n'a pas que pour seule utilité de déplacer les pions sur le plateau mais aussi d'empêcher les pions de sortir du tableau, et la fonction contient aussi une autre fonction pour manger les pions blancs. C'est la même fonction qui se trouve au-dessus mais inversée.

4 Team Gantt

4.1 Lien

https://prod.teamgantt.com/gantt/schedule/?ids=2933439#&ids=2933439&user=&custom=&company=&hide_completed=false&date_filter=&color_filter=

4.2 Diagramme



5 lien vers d'autres outils

5.1 *Trello*

<https://trello.com/b/jdz4WzEP/5ti-gianlucaPETRONIO>

5.2 *Git*Hub

https://github.com/GianlucaPtrn/JeuDeDames_GianlucaPetronio.git

6 Conclusion

En conclusion, c'est un projet qui m'a apporté beaucoup de plaisir à faire. Premièrement, il m'a réellement fait progresser et m'a appris à manier le langage c#. Ensuite, j'ai été obligé de faire des analyses avant chaque fonction (morceau de programme). De ce fait, ça rend le travail de programmation beaucoup plus facile et permet de mieux visualiser le morceau à coder, pour minimiser le nombre d'erreurs dans le code. Enfin, avant de commencer ce travail, mon niveau de programmation était très médiocre. Maintenant, je sais programmer correctement et reconnaître les erreurs indiqués. Pour finir, il me manque la fonction pour détecter les erreurs humaines (par exemple, se tromper sur le pions à avancer) et aussi faire la fonction pour un système de victoire. J'aurais dû m'y prendre un peu plus tôt. Afin de tout terminer au mieux. Merci pour votre lecture et votre attention.

7 Bibliographie

7.1 Sources utilisés pour les règles et le buts du jeu

<https://www.lecomptoirdesjeux.com/regle-jeu-dames.htm>

<http://www.ffjd.fr/Web/index.php?page=reglesdujeu#top>

<https://www.regles-de-jeux.com/regle-du-jeu-de-dames/>

7.2 Sources utilisés pour le codage

<https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/arrays/multidimensional-arrays>

<https://docs.microsoft.com/fr-fr/archive/msdn-magazine/2012/december/csharp-matrix-decomposition>

<https://waytolearnx.com/2017/04/programmation-en-c-les-boucles.html#:~:text=La%20boucle%20infinie&text=Vous%20pouvez%20avoir%20une%20expression,sur%20les%20touches%20Ctrl%20%2B%20C.>

<https://csharp.net-tutorials.com/fr/110/control-structures/linstruction-if/>