Couleur avec flèches

using System;

namespace MatriceColor

{

class Program

{

static void Main(string[] args)

{

// Initialisation de la matrice

int[,] matrice = new int[5, 5];

int currentRow = 0;

int currentCol = 0;

// Boucle principale

while (true)

{

Console.Clear();

AfficherMatrice(matrice, currentRow, currentCol);

ConsoleKeyInfo keyInfo = Console.ReadKey(true);

// Déplacement dans la matrice

if (keyInfo.Key == ConsoleKey.LeftArrow && currentCol > 0)

{

currentCol--;

}

else if (keyInfo.Key == ConsoleKey.RightArrow && currentCol < matrice.GetLength(1) - 1)

{

currentCol++;

}

else if (keyInfo.Key == ConsoleKey.UpArrow && currentRow > 0)

{

currentRow--;

}

else if (keyInfo.Key == ConsoleKey.DownArrow && currentRow < matrice.GetLength(0) - 1)

{

currentRow++;

}

// Sélection de la case avec la touche "Entrée"

else if (keyInfo.Key == ConsoleKey.Enter)

{

Console.ForegroundColor = ConsoleColor.Green;

matrice[currentRow, currentCol] = 1;

Console.ForegroundColor = ConsoleColor.White;

}

}

}

static void AfficherMatrice(int[,] matrice, int currentRow, int currentCol)

{

for (int i = 0; i < matrice.GetLength(0); i++)

{

for (int j = 0; j < matrice.GetLength(1); j++)

{

if (i == currentRow && j == currentCol)

{

Console.BackgroundColor = ConsoleColor.Blue;

}

if (matrice[i, j] == 1)

{

Console.ForegroundColor = ConsoleColor.Green;

Console.Write(matrice[i, j]);

}

else

{

Console.ForegroundColor = ConsoleColor.White;

Console.Write(matrice[i, j]);

}

Console.BackgroundColor = ConsoleColor.Black;

Console.Write(" ");

}

Console.WriteLine();

}

}

}

}

Modèle d’analyse

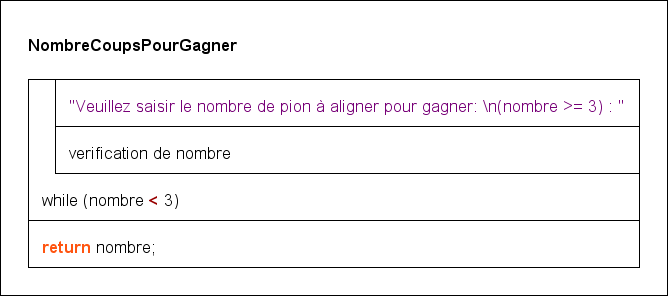
Enoncé

spécification

| NombreCoupsPourGagner | Paramètres | | | |
| --- | --- | --- | --- | --- |
| Nom | Type | Description | In/Out |
| nombre | int | nombre de pions à aligner | out |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
| Hypothèses (conditions sur paramètres IN) | on suppose que l’utilisateur entrera une valeur >= au nombre | | | |
| But | qu’il puisse demander combien de pion il veut aligner | | | |

Variables locales

GNS



Modèle d’analyse

Enoncé

spécification

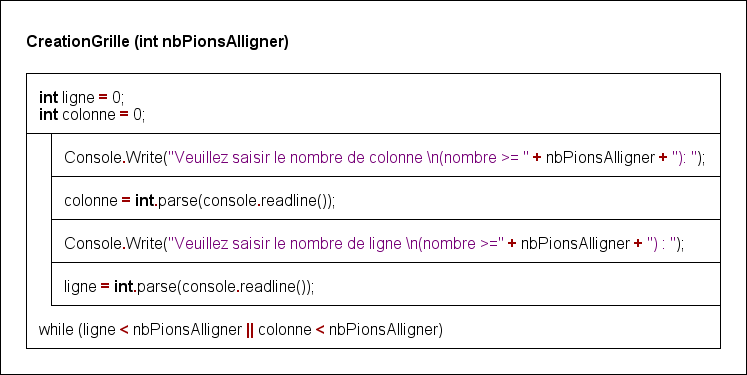
| CreationGrille | Paramètres | | | |
| --- | --- | --- | --- | --- |
| Nom | Type | Description | In/Out |
| nbPionsAlligner | int | nombre de pions à aligner | in |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
| Hypothèses (conditions sur paramètres IN) | on suppose que l’utilisateur entrera des valeurs de ligne et de colonne >= au nbPionsAlligner | | | |
| But | que les lignes et colonne soit > que nbPionsAlligner | | | |

Variables locales

int ligne

int colonne

GNS



using System;

namespace ConsoleApp1

{

class Program

{

static void Main(string[] args)

{

int nbPionsAlligner = 4;

string[,] grille;

int colonne;

int ligne;

CreationGrille(nbPionsAlligner, out grille, out ligne, out colonne);

for (int i = 0; i < ligne; i++)

{

for (int y = 0; y < colonne; y++)

{

Console.Write(grille[i, y]);

}

Console.WriteLine();

}

}

static void CreationGrille(int nbPionsAlligner, out string[,] grille, out int ligne, out int colonne)

{

Console.WriteLine("Création de la grille:");

Console.WriteLine();

ligne = 0;

colonne = 0;

do

{

Console.Write("Veuillez saisir le nombre de ligne \n(nombre >=" + nbPionsAlligner + ") : ");

ligne = int.Parse(Console.ReadLine());

Console.WriteLine();

Console.Write("Veuillez saisir le nombre de colonne \n(nombre >= " + nbPionsAlligner + "): ");

colonne = int.Parse(Console.ReadLine());

Console.WriteLine();

} while (ligne < nbPionsAlligner || colonne < nbPionsAlligner);

grille = new string[ligne, colonne];

for (int i = 0; i < ligne; i++)

{

for (int y = 0; y < colonne; y++)

{

grille[i, y] = "|0|";

}

}

}

}

}

Modèle d’analyse

Enoncé

spécification

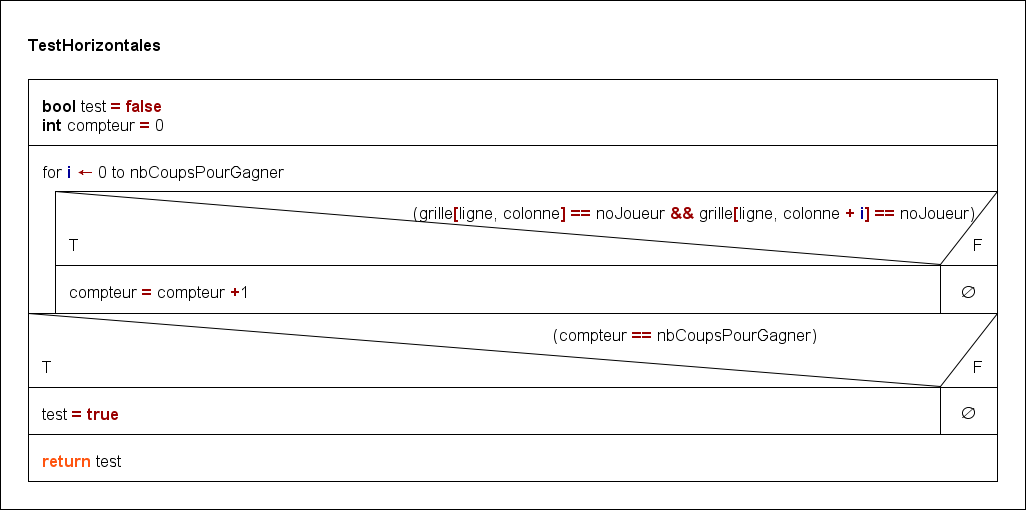
| TestHorizontales | Paramètres | | | |
| --- | --- | --- | --- | --- |
| Nom | Type | Description | In/Out |
| grille | int[,] | la matrice | out |
| ligne | int | nombre de ligne | in |
| colonne | int | nombre de colonne | in |
| noJoueur | int | verification pour voir si le test est bien passer | in |
| nbCoupsPourGagner | int | le nombre de coup pour gagner la partie | in |
| Hypothèses (conditions sur paramètres IN) | la grille ne doit pas être rempli au maximum | | | |
| But | tester le programme horizontalement | | | |

Variables locales

test bool

compteur int

GNS



Modèle d’analyse

Enoncé

spécification

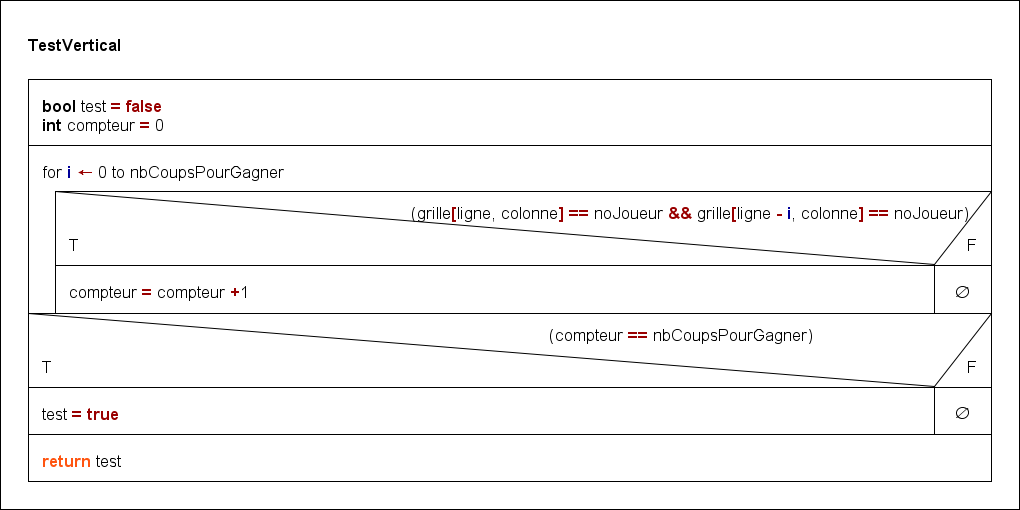
| TestVertical | Paramètres | | | |
| --- | --- | --- | --- | --- |
| Nom | Type | Description | In/Out |
| grille | int[,] | la matrice | out |
| ligne | int | nombre de ligne | in |
| colonne | int | nombre de colonne | in |
| noJoueur | int | verification pour voir si le test est bien passer | in |
| nbCoupsPourGagner | int | le nombre de coup pour gagner la partie | in |
| Hypothèses (conditions sur paramètres IN) | la grille ne doit pas être rempli au maximum | | | |
| But | tester le programme horizontalement | | | |

Variables locales

test bool

compteur int

GNS



Modèle d’analyse

Enoncé

spécification

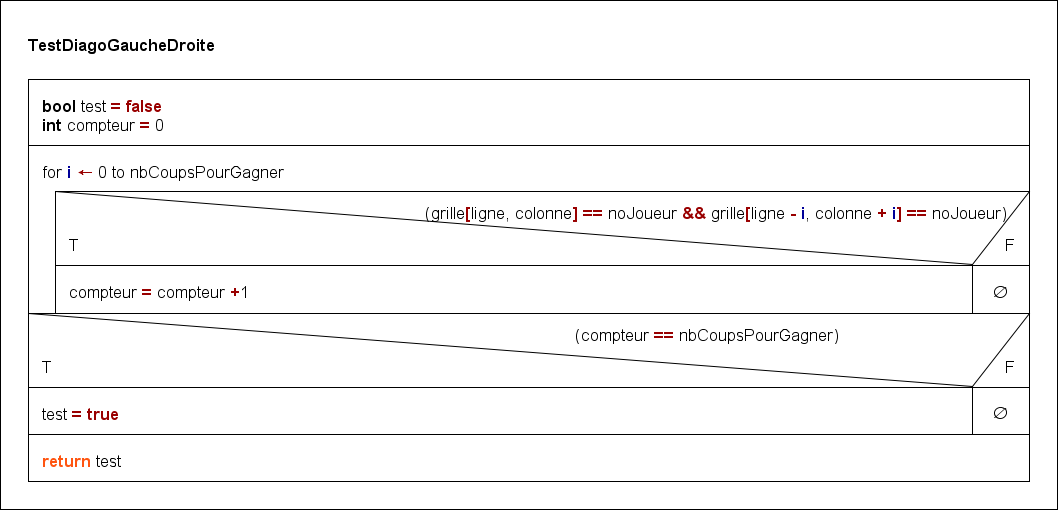
| TestDiagoGaucheDroite | Paramètres | | | |
| --- | --- | --- | --- | --- |
| Nom | Type | Description | In/Out |
| grille | int[,] | la matrice | out |
| ligne | int | nombre de ligne | in |
| colonne | int | nombre de colonne | in |
| noJoueur | int | verification pour voir si le test est bien passer | in |
| nbCoupsPourGagner | int | le nombre de coup pour gagner la partie | in |
| Hypothèses (conditions sur paramètres IN) | la grille ne doit pas être rempli au maximum | | | |
| But | tester le programme horizontalement | | | |

Variables locales

test bool

compteur int

GNS



Modèle d’analyse

Enoncé

spécification

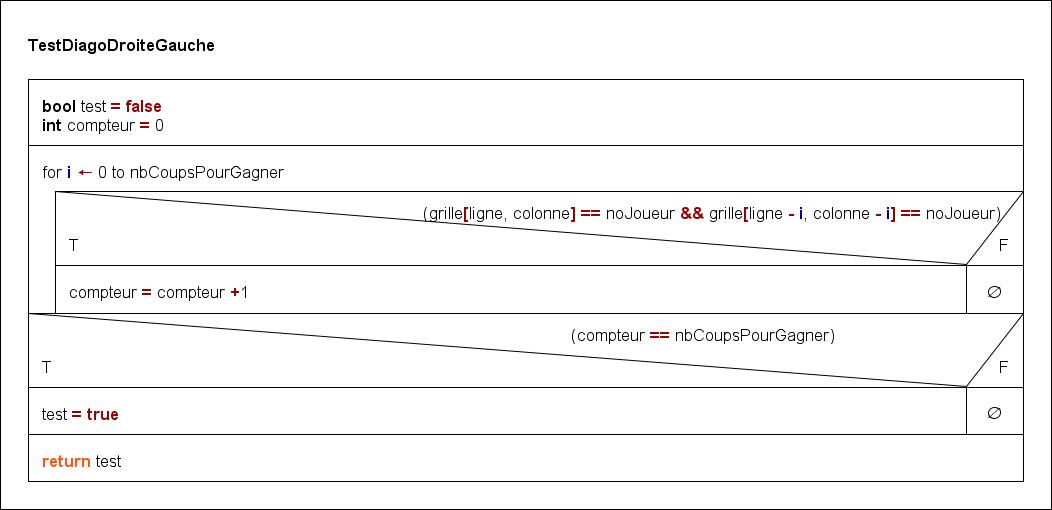
| TestDiagoDroiteGauche | Paramètres | | | |
| --- | --- | --- | --- | --- |
| Nom | Type | Description | In/Out |
| grille | int[,] | la matrice | out |
| ligne | int | nombre de ligne | in |
| colonne | int | nombre de colonne | in |
| noJoueur | int | verification pour voir si le test est bien passer | in |
| nbCoupsPourGagner | int | le nombre de coup pour gagner la partie | in |
| Hypothèses (conditions sur paramètres IN) | la grille ne doit pas être rempli au maximum | | | |
| But | tester le programme horizontalement | | | |

Variables locales

test bool

compteur int

GNS



Modèle d’analyse

Enoncé

faire une analyse de la victoire

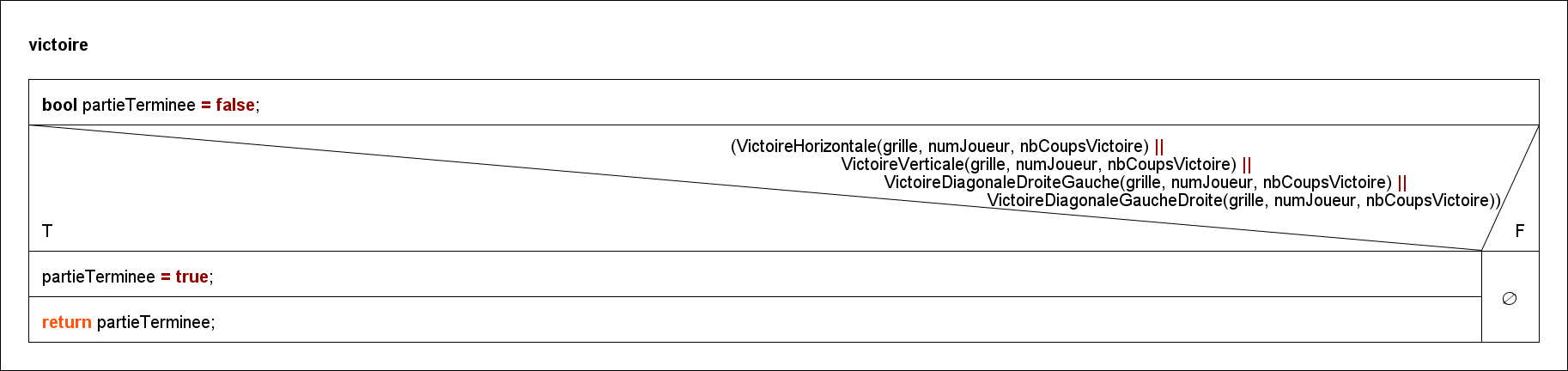
spécification

| victoire | Paramètres | | | |
| --- | --- | --- | --- | --- |
| Nom | Type | Description | In/Out |
| grille | int[,] | grille complete | in |
| numJoueur | int | numero du joueur | in |
| nbCoupsVictoire | int | nombre Coups Victoire | in |
|  |  |  |  |
| Hypothèses (conditions sur paramètres IN) | on suppose que l’utilisateur entrera une valeur >= au nombre | | | |
| But | qu’il puisse demander combien de pion il veut aligner | | | |

Variables locales

bool partieTerminee = false;

GNS



Modèle d’analyse

Enoncé

faire une analyse de la egalité

spécification

| egalite | Paramètres | | | |
| --- | --- | --- | --- | --- |
| Nom | Type | Description | In/Out |
| grille | int[,] | grille complète | in |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
| Hypothèses (conditions sur paramètres IN) | on suppose que l’utilisateur entrera une valeur >= au nombre | | | |
| But | qu’il puisse demander combien de pion il veut aligner | | | |

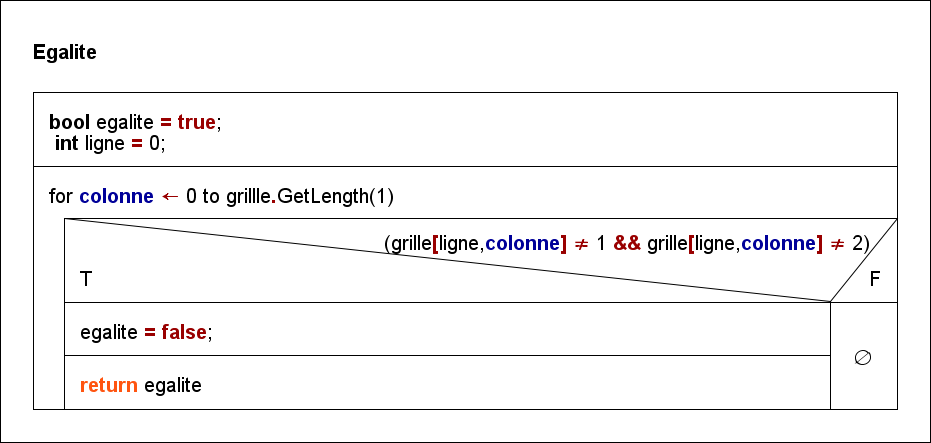
Variables locales

bool égalité= true

int ligne = 0;

ligne && colonne int

GNS



Test position Valide

Enoncé

spécification

| testPositionValide  bool | Paramètres | | | |
| --- | --- | --- | --- | --- |
| Nom | Type | Description | In/Out |
| grille | int[,] | grille | in |
| colonne | int | la colonne de la position du pion | in |
| noJoueur | int | quel joueur | out |
|  |  |  |  |
| Hypothèses (conditions sur paramètres IN) | on suppose qu’il ne se trompe pas de colonne | | | |
| But | de corriger l’erreur (en dehors de la matrice) | | | |

Variables locales

bool test = false;

GNS

Position Valide

Enoncé

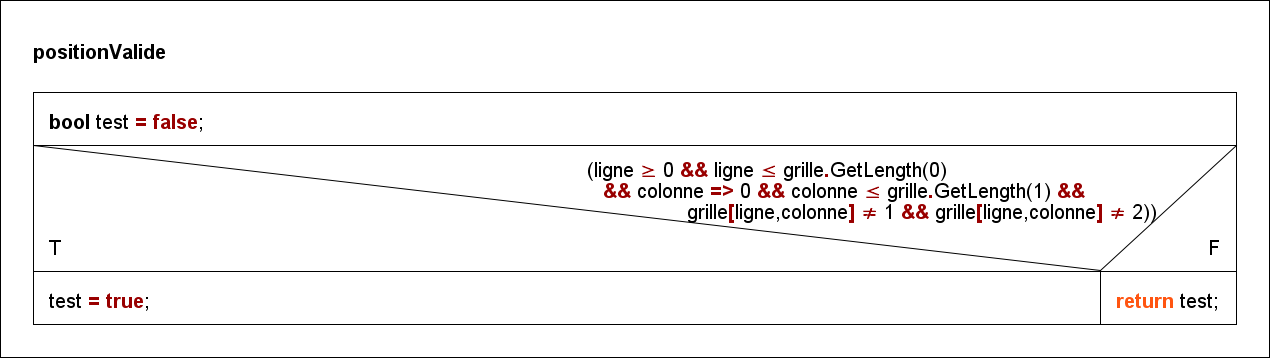
spécification

| PositionValide  bool | Paramètres | | | |
| --- | --- | --- | --- | --- |
| Nom | Type | Description | In/Out |
| grille | int[,] | grille | in |
| colonne | int | la colonne de la position du pion | in |
| noJoueur | int | quel joueur | out |
|  |  |  |  |
| Hypothèses (conditions sur paramètres IN) | on suppose qu’il ne se trompe pas de colonne | | | |
| But | de corriger l’erreur (en dehors de la matrice) | | | |

Variables locales

bool test = false;

GNS



Modifier La Grille

Enoncé

spécification

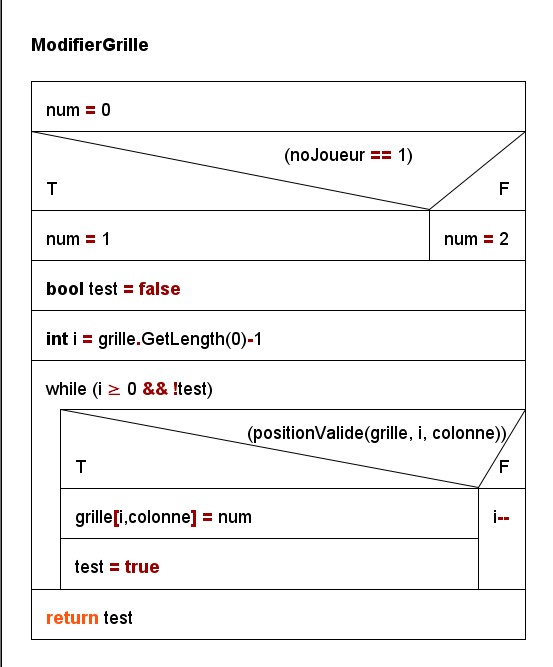
| ModifierGrille | Paramètres | | | |
| --- | --- | --- | --- | --- |
| Nom | Type | Description | In/Out |
| grille | int[,] | grille | in |
| colonne | int | la colonne de la position du pion | in |
| noJoueur | int | quel joueur | out |
|  |  |  |  |
| Hypothèses (conditions sur paramètres IN) | qu’il n’y ai pas d’autre joueur a part le 1 et le 2 | | | |
| But | modifier la grille pour que la machine puisse determiner quelquechose (gagnant, perdant, egalite, continue de la partie…) | | | |

Variables locales

bool test = false;

int num

GNS



Colonne Pleine

Enoncé

spécification

| colonnePleine | Paramètres | | | |
| --- | --- | --- | --- | --- |
| Nom | Type | Description | In/Out |
| grille | int[,] | grille | out |
| colonne | int | la colonne de la position du pion | in |
|  |  |  |  |
|  |  |  |  |
| Hypothèses (conditions sur paramètres IN) | la colonne ne doit pas être rempli sinon la colonne est pleine | | | |
| But | devoir lui renvoyer que la colonne est pleine | | | |

Variables locales

bool colonnePleine= false;

GNS

