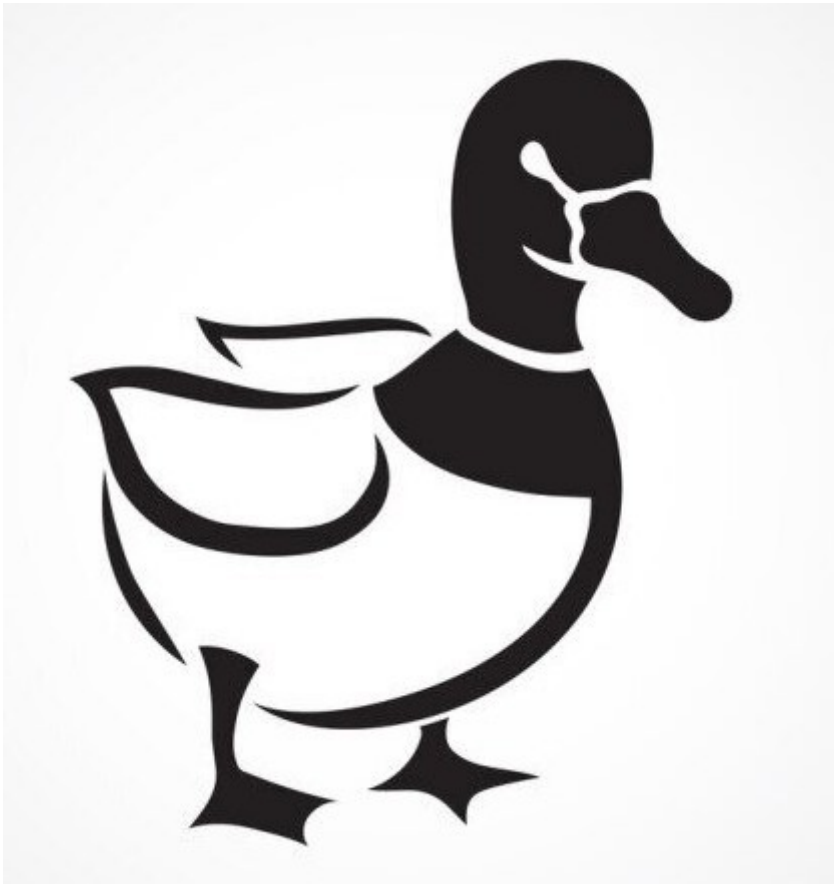
Equipe Maxime, Nicolas, Quentin

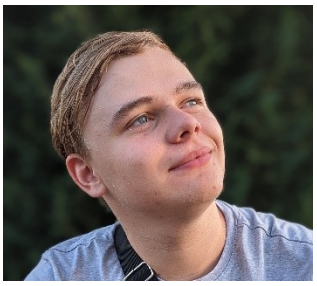
Devoir d'algorithmie

JEU DE L'OIE



INTRODUCTION

L’EQUIPE :



Maxime GRANDIDIER,
Je suis Bachelier d'un bac général et technologique, spécialité Mathématique et Science de l'ingénieur Je suis passionné d'informatique c'est pourquoi je cherche à devenir développeur informatique.

Nicolas LEIRITZ, 22 ans
J'ai fais un BTS STL SPCL puis un BTS CIRA afin de programmer des automates en usines. J'ai fais une formation CDA à l'AFPA de Pompey. J'ai ensuite intégré PCN afin de pouvoir rencontrer des professionnels du monde du numérique et j'ai enfin intégré le CESI.

Quentin ALVERNHE, 32 ans.

Je suis au CESI en reconversion professionnelle. J’ai d’abords travaillé dans la production audiovisuelle en tant que chargé de production et assistant mise en scène. Dans le cadre de cette reconversion j’ai notamment fais la Coding Academy by Epitech. À terme je souhaiterais m’orienter vers du développement web en front-end.

PARTIE 1 : LE JEU DE L'OIE

I/ METHODE DE TRAVAIL

1 /ORGANISATION

Pour ce TP nous devons réaliser un Algorithme régissant un jeu de l'oie avec certaines contraintes.
Nous devons faire en sorte que le joueur puisse lancer un dé, et en fonction de la case sur laquelle il tombe, certains évènement se produisent.
Les évènements susceptibles de se produire sont les suivants :
- Lorsque le joueur tombe sur les cases 10, 30 ou 50 il recule d'une case.
- Lorsque le joueur tombe sur les cases 20, 40 ou 60 il recule de deux cases.
- Le joueur doit tomber pile sur la case 63 pour finir, si il va plus loin, il recule du nombre de case supplémentaire.

Nous devons aussi enregistrer un certain nombre de données qui seront affichés à la fin de la partie. La partie est terminée soit parce que le joueur a gagné, soit parce qu'il a décidé d'arrêter de jouer.

Dans un premier temps nous avons formalisé les différentes étapes sous forme d'un logigramme générale(1) afin d'avoir une vision globale et simplifiée du travail à faire.
En réalisant ce logigramme, nous avons vite identifié deux étapes importantes et assez complexes. Nous avons donc décidé de créer deux logigrammes supplémentaires pour formaliser ces deux étapes qui sont :
- Le calcul de la position et du déplacement du joueur (2)
- La mise à jour et l'enregistrement des données du jeu (3)

Une fois ce travail fait, nous avons fusionné le logigramme détaillant la position et le déplacement du joueur (2) avec le logigramme détaillant la mise à jour et l'enregistrement des données (3) dans un quatrième logigramme de synthèse (4).
De cette façon, nous avons pu avoir une vision d'ensemble cohérente de l'architecture que devrait avoir l'algorithme une fois terminé.

Dans un second temps, une fois tous les logigrammes réalisés, nous avons pu commencer la rédaction de l'algorithme, étape par étape. À chaque étape, nous écrivions une version de l'algorithme chacun de notre côté avant de mettre notre travail en commun pour avoir la version qui nous paraissait la plus cohérente.
C'est en procédant de cette manière que nous avons pu isoler les différents problèmes et trouver les solutions qui nous paraissent les plus logiques.

Pour finir nous avons pris du temps pour expliquer notre méthode et notre organisation de travail afin de faciliter la compréhension pour un tiers. Nous nous sommes répartis les tâches pour réécrire chaque élément au propre et pour mettre tout en commun dans un dossier cohérent.

2/ REFLEXION SUR L'ALGORITHME

Pour ce travail nous avons opté pour le choix d'une boucle TANT QUE.
En effet, nous avons assez vite formalisé le lancer du dé, et le fait qu'à chaque lancé, la position (variable *pos*) est mise à jour en ajoutant le nouveau score du dé.
Nous avons donc décidé d'incorporer cette mécanique dans une boucle pour répéter l'action tant que le joueur décide de continuer à jouer (variable *cj*) et que sa position le permet. Enfin, nous avons ajouté dans cette boucle différentes conditions pour que des effets s'appliquent en fonction de la position.
Ainsi l'algorithme calcule chaque lancé de dé en les additionnant pour déterminer la position du joueur.
À chaque tour, l'algorithme interroge la position du joueur et applique l'effet désiré s'il est sur une case spéciale.
De la même manière, si la position du joueur dépasse la case 63, l'Algorithme calcule la différence et fait reculer le joueur du même nombre de case, tout en comptabilisant le nombre de fois ou cela se produit.
Enfin nous avons ajouté des compteurs qui s'activent à chaque fois que le joueur tombe sur une case spéciale ou dépasse la case 63 afin de pouvoir savoir combien de fois chaque évènement s'est produit.

3/ LES VARIABLES :

Pour pouvoir écrire cet algorithme, nous avons dû créer plusieurs variables qui sont décrites ci-dessous.

- dé : Valeur du jet de dé du joueur

- pos : Position du joueur

- nbl : Nombre de lancer de dé du joueur

- totaldé : Total de tous les jets de dé additionnés

- nbd : Nombre de recul à cause des dizaines

- nbdd : Nombre de recul à cause des double dizaines

- nbrd : Nombre de recul à cause du dépassement de la case 63

- moyenne : La moyenne des lancés de dé

- cj : Booléen pour déterminer si le joueur désir continuer à joueur

4/ LES ANNEXES (LOGIGRAMMES) :

Vous trouverez plus bas dans ce dossier les annexes correspondantes aux différents logigramme que nous avons évoqué plus haut.

(1) Logigramme générale
(2) Logigramme du calcul de la position et du déplacement du joueur
(3) Logigramme de la mise à jour et de l'enregistrement des données
(4) Logigramme de synthèse

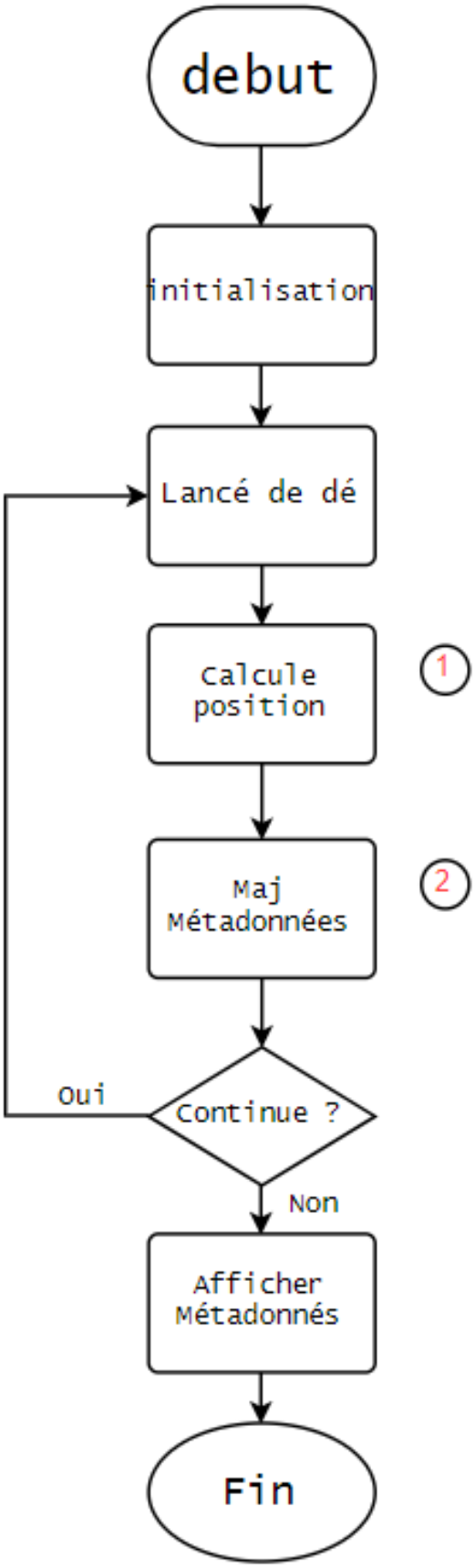
II/ ALGORITHME DU JEU DE L'OIE

```
ALGO JEU_DE_L_OIE

Données

   Entier : dé, pos, nbl, totaldé, nbd, nbdd, nbrd

   Réel : moyenne

   Bool : cj

Fin_Données

DEBUT

   dé <- 0
   pos <- 0
   nbl <- 0
   totaldé <- 0
   nbd <- 0
   nbdd <- 0
   nbrd <- 0
   moyenne <- 0

   SAISIR "Voulez-vous jouer ?", cj
   TANT QUE cj = VRAI ET pos < 63

         dé <- RANDOM(1 ;6)
         totaldé <- totaldé + dé
         pos <- pos + dé
         nbl <- nbl + 1
         moyenne <- totaldé / nbl


         SI pos > 63
               ALORS nbrd <- nbrd + 1
                     pos <- 63 - (pos - 63)
         FIN_SI

         SI pos = 10 OU pos = 30 OU pos = 50
               ALORS nbd <- nbd + 1
                     pos <- pos – 1
               SINON SI pos = 20 OU pos = 40 OU pos = 60
                        ALORS nbdd <- nbdd + 1
                              pos <- pos – 2
                        SINON SI pos = 63
                                 ALORS AFFICHER "Bravo, vous avez gagné !"
                                       cj <- FAUX
         FIN_SI


         SI cj = VRAI
               ALORS AFFICHER "Le joueur est à la case ", pos, "."
               SAISIR  "Voulez-vous continuer ? VRAI/FAUX", cj
         FIN_SI


   FIN_TANT_QUE

 AFFICHER "Partie terminée !"
 AFFICHER "----------Données du joueur ----------"
 AFFICHER "Le joueur a lancé le dé ", nbl, " fois"
 AFFICHER "Le joueur a reculé ", nbd, " fois à cause des dizaine"
 AFFICHER "Le joueur a reculé ", nbdd, " fois à cause des double-dizaine"
 AFFICHER "Le joueur a reculé ", nbrd, " fois à cause du dépassement de la case 63"
 AFFICHER "La moyenne des lancés de dé du joueur est de ", moyenne, "."

FIN
```

ANNEXES

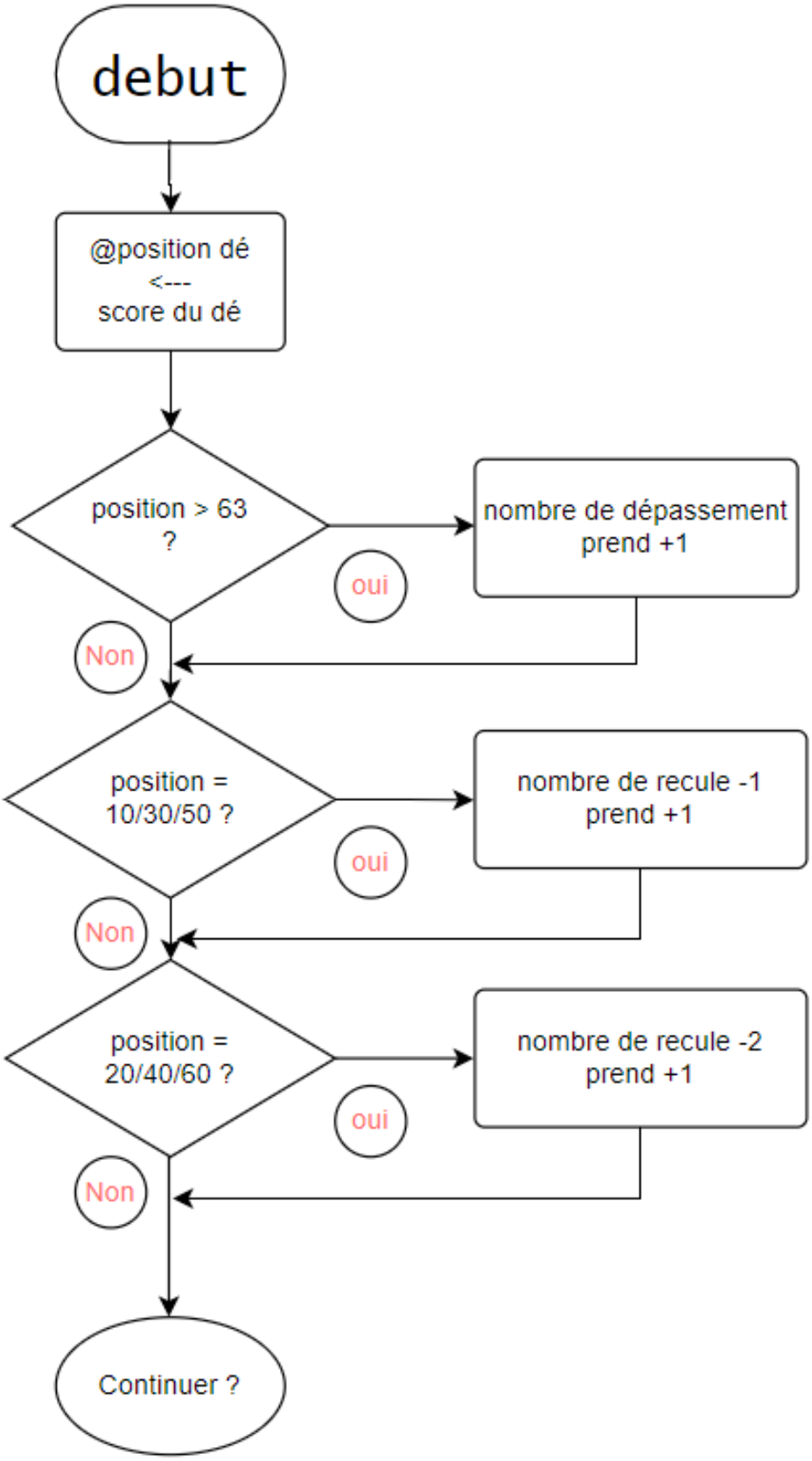
1) Logigramme 1

Logigramme général de l'algorithme



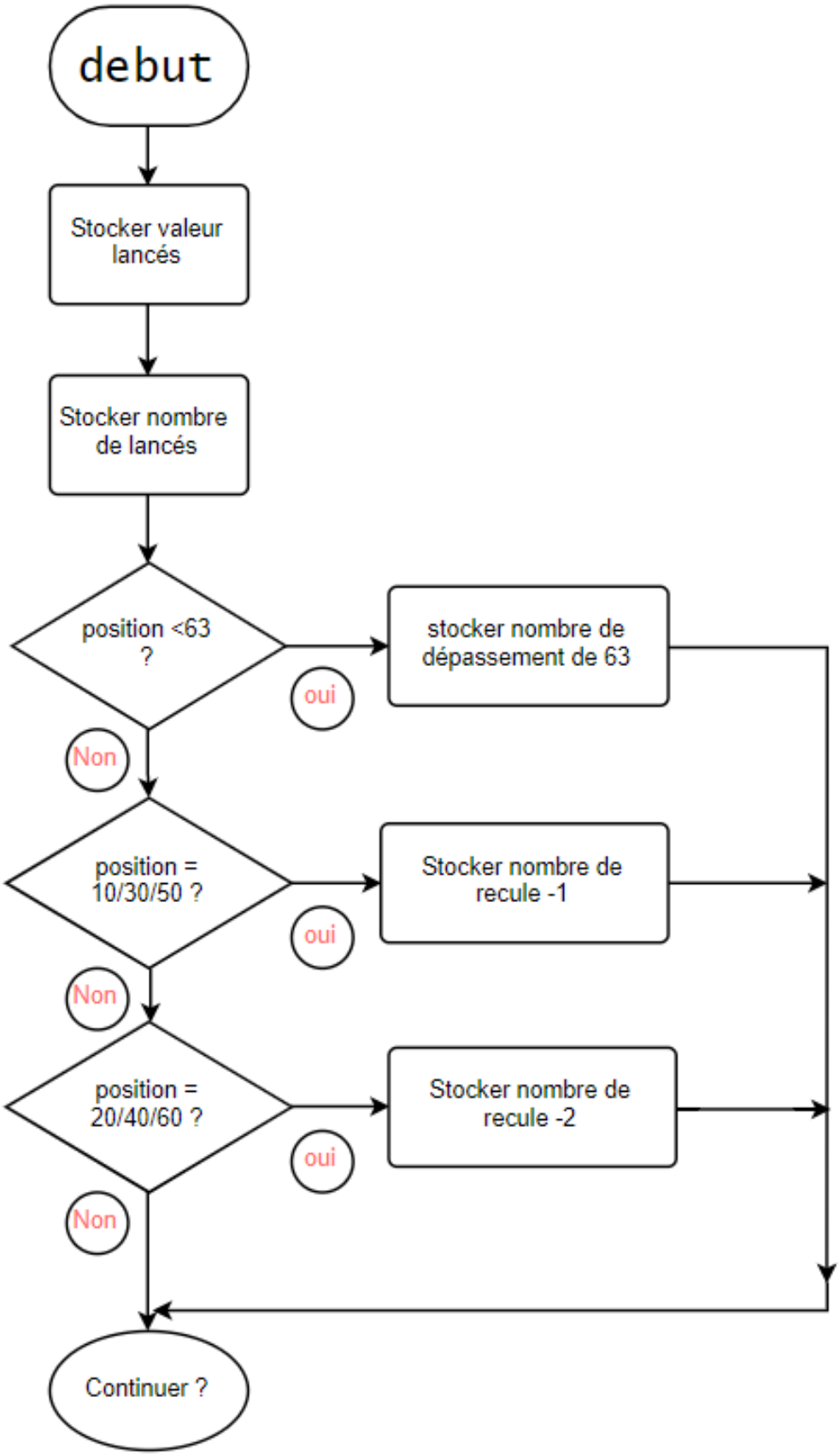
2) Logigramme 2

Logigramme du déplacement du joueur



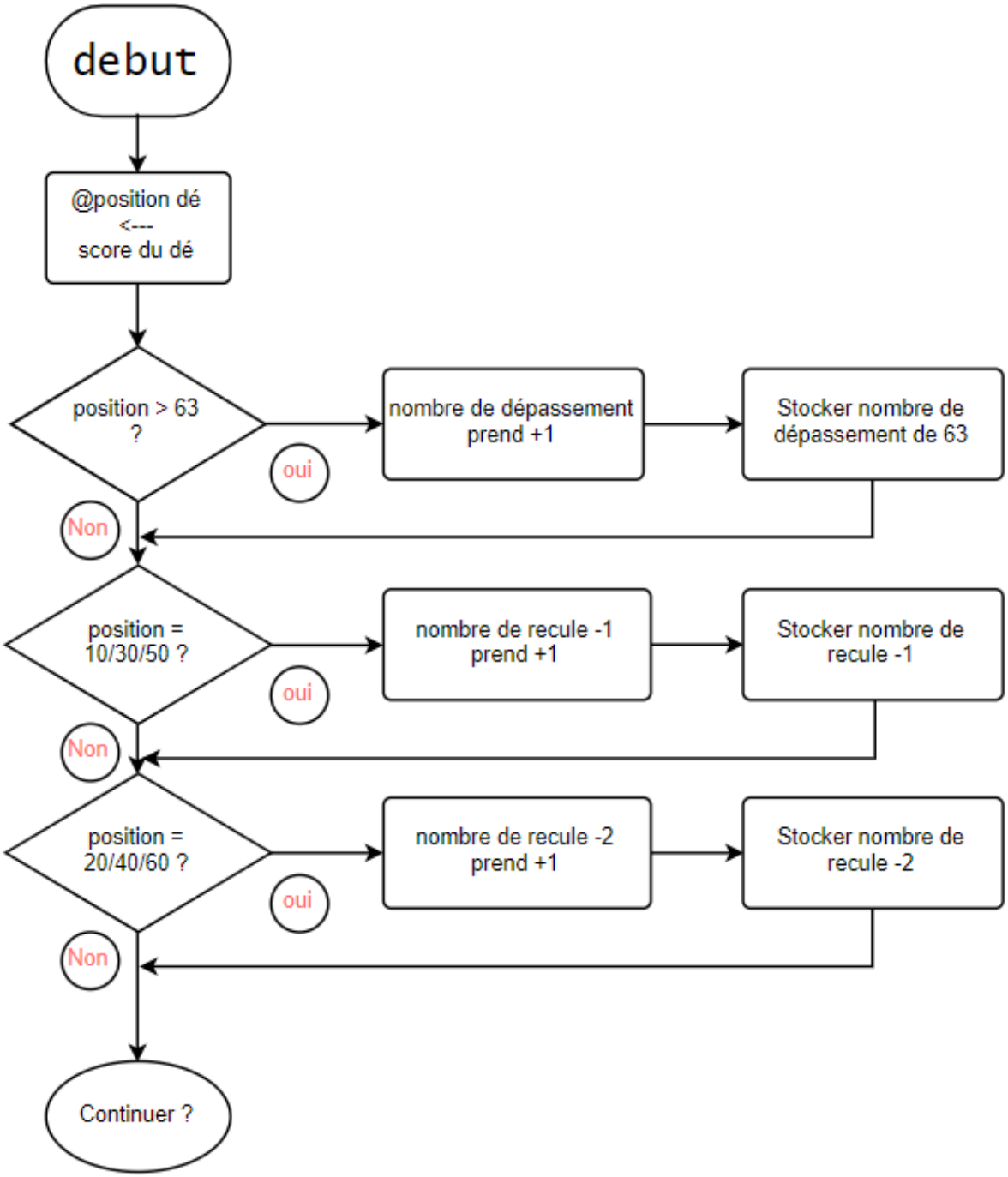
3) Logigramme 3

Logigramme pour le stockage des données



4) Logigramme 4

Logigramme de synthèse comprenant le déplacement et le stockage des données



PARTIE 2 : NOUVELLES REGLES

I/ METHODE DE TRAVAIL

1 /ORGANISATION

Dans la mesure ou le jeu de l'Oie fonctionne bien, il nous était demandé de réfléchir à des règles supplémentaires à ajouter au jeu. Nous avons donc réfléchis à de nouvelles règles, chacun de notre côtés puis avons mis nos propositions en commun pour choisir quelles règles nous souhaitions ajouter.
Finalement notre choix s'est porté sur 5 fonctionnalités supplémentaires pour ce jeu :

- Le jeu est jouable à deux. Le joueur 1 part de la case 0 pour aller en case 63, le joueur 2 part de la case 63 pour aller en case 0.
- Les cases 31 et 41 sont des prisons, si on tombe dessus on a 5 tours pour faire un 4+ pour pouvoir sortir, sinon on est éliminé.
- Au début de la partie, il est possible de choisir de jouer avec un D6 ou un D8.
- La case 42 téléporte le joueur qui tombe dessus sur une case au hasard.
- Si un joueur dépasse 5 fois la case 63 ou la case 0, il est éliminé.

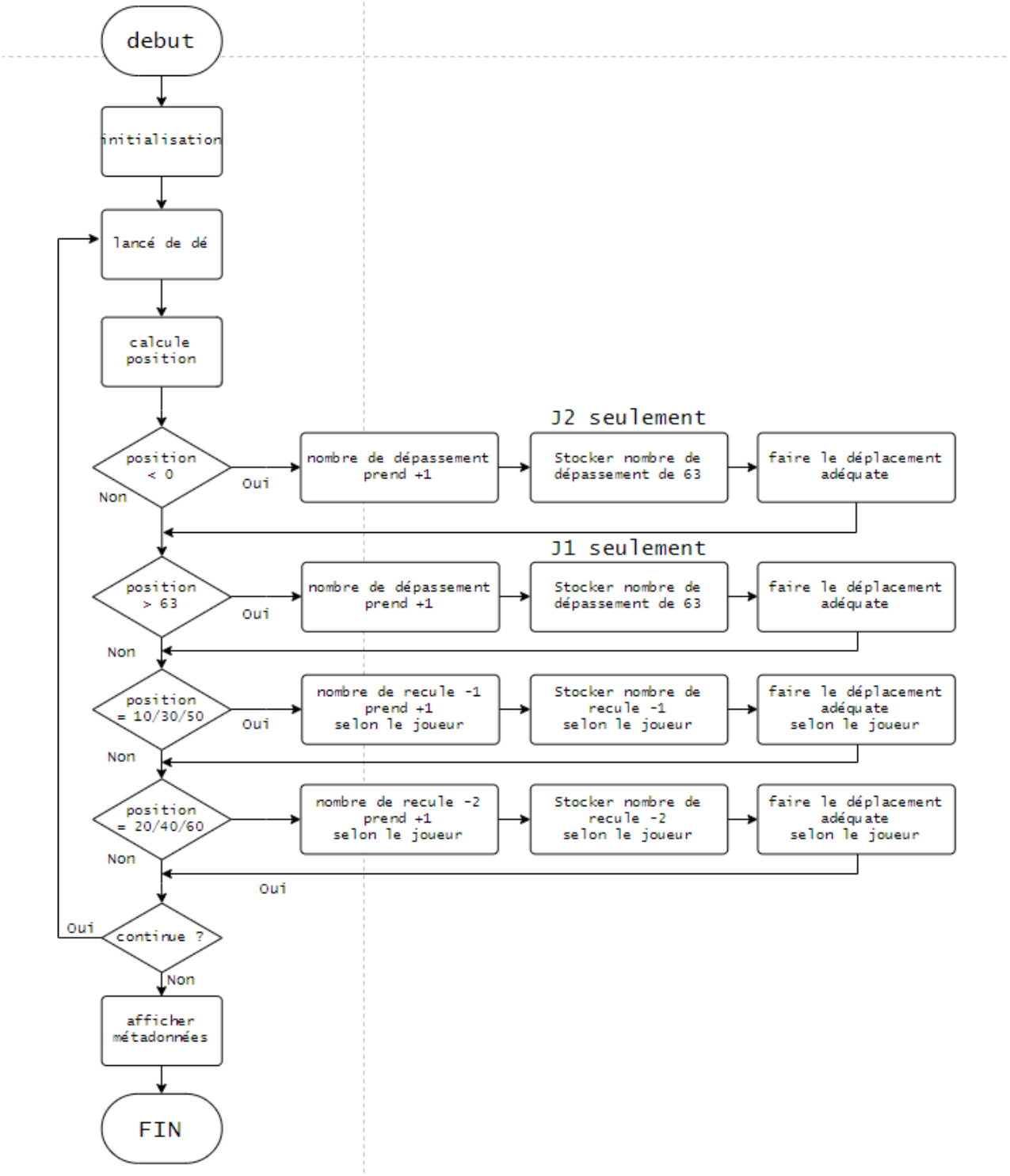
Pour ajouter ces nouvelles règles, nous sommes partis de notre algorithme de départ et avons implémenté les nouvelles fonctions chacun de notre côté, pour pouvoir avoir différent point de vue.
Nous nous sommes ensuite réunis pour comparer nos travaux et choisir quelle solution nous paraissait la meilleur et la plus simple à mettre en œuvre. Une fois que nous avons choisis quelle solution nous apparaissait comme la meilleure pour chaque règle nous avons compilé toute les règles dans un seul et même algorithme intégrant toute les fonctionnalités.

2/ REFLEXION SUR LES NOUVELLES REGLES

Deuxième joueur :

Pour cette règle, nous avons doublé les variables que nous avions définis afin d'avoir la même mécanique s'appliquant au joueur 1 et au joueur 2 et afin de pouvoir récupérer les données des deux joueurs également.
Nous avons pris le partis de faire partir le joueur deux depuis la case 63, son but étant d'atteindre la case 0.
Pour cela il nous a simplement suffit de changer la position de départ du joueur deux et, plutôt que d'additionner les jets de dé pour calculer la position, le résultat du dé est soustrait à la position du joueur.
De plus, nous avons du penser aux cases spéciales, ainsi au lieu de reculer de 1 ou 2 cases, le joueur 2 avance de une ou deux cases si il tombe sur une case spéciale.
Ainsi le joueur 1 progresse de la case 0 vers la case 63 et le joueur 2 régresse de la case 63 vers la case 0.

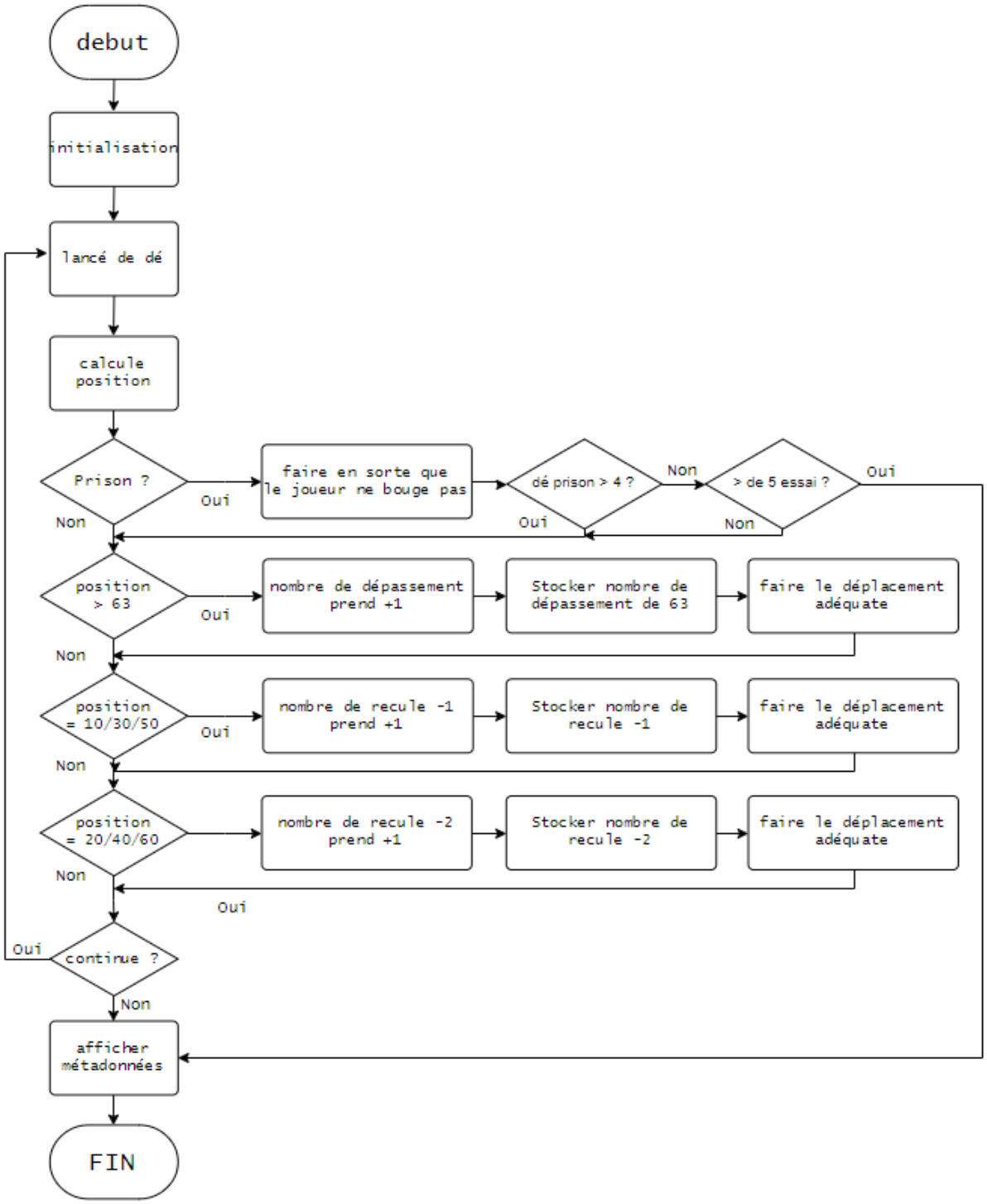
logigramme représentant le jeu de l'oie avec un deuxième joueur en page suivante



La Prison :

Pour la prison, nous avons choisis arbitrairement deux cases car nous avions déjà prévu un deuxième joueur.
Nous avons donc créé une variable supplémentaire pour déterminer si un joueur est en prison ou non. De plus nous avons mis en place un mécanisme pour que le joueur puisse sortir de prison sur un jet de dé de 4 ou plus (s'il réussit il avance d'une case). Nous avons donc créé un nouveau dé (variable *déprison*) pour que les jets de dés pour sortir de prison soient bien comptabilisés. De plus nous avons également ajouté un compteur afin de compter le nombre d'essai du joueur(variable *essaiescape*), au cinquième échec, il est éliminé.
Nous avons utilisé une boucle SELON pour cet algorithme pour signaler au joueur, en fonction de son nombre d'essai, combien de chance il lui reste. Ainsi en fonction du résultat et du nombre d'essai, le joueur est soit autorisé à sortir soit il doit patienter, soit il est éliminé.

logigramme représentant le jeu de l'oie avec la présence d'une case prison

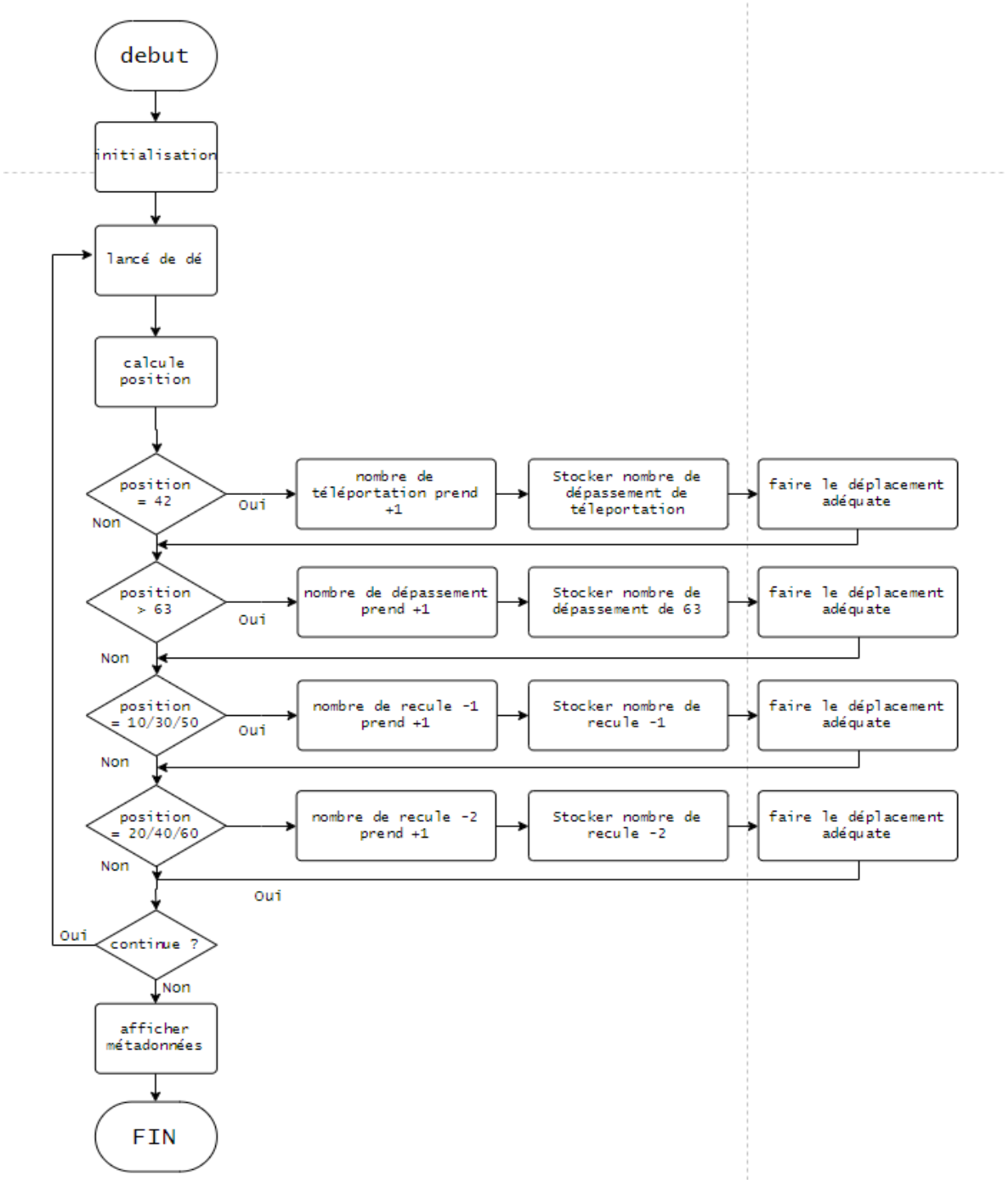


Le téléporteur :

Plus simple, cette règle nous a demandé moins de réflexion. Nous avons simplement eu à ajouter une fonction RANDOM qui va déterminer la nouvelle position du joueur au hasard.
Cet évènement se produit lorsque le joueur tombe sur la case 42.

De façon concrète cette nouvelle règle ne nous a pas posé trop de problème. Nous avons surtout veillé à ce qu'elle intervienne assez tôt dans l'Algorithme afin que ses effets soient pris en compte avant les autres évènements. Ainsi si la case 42 téléporte le joueur sur une case spéciale, l'évènement devant se produire peut bien avoir lieu.

logigramme représentant le jeu de l'oie comportant une case de téléportation



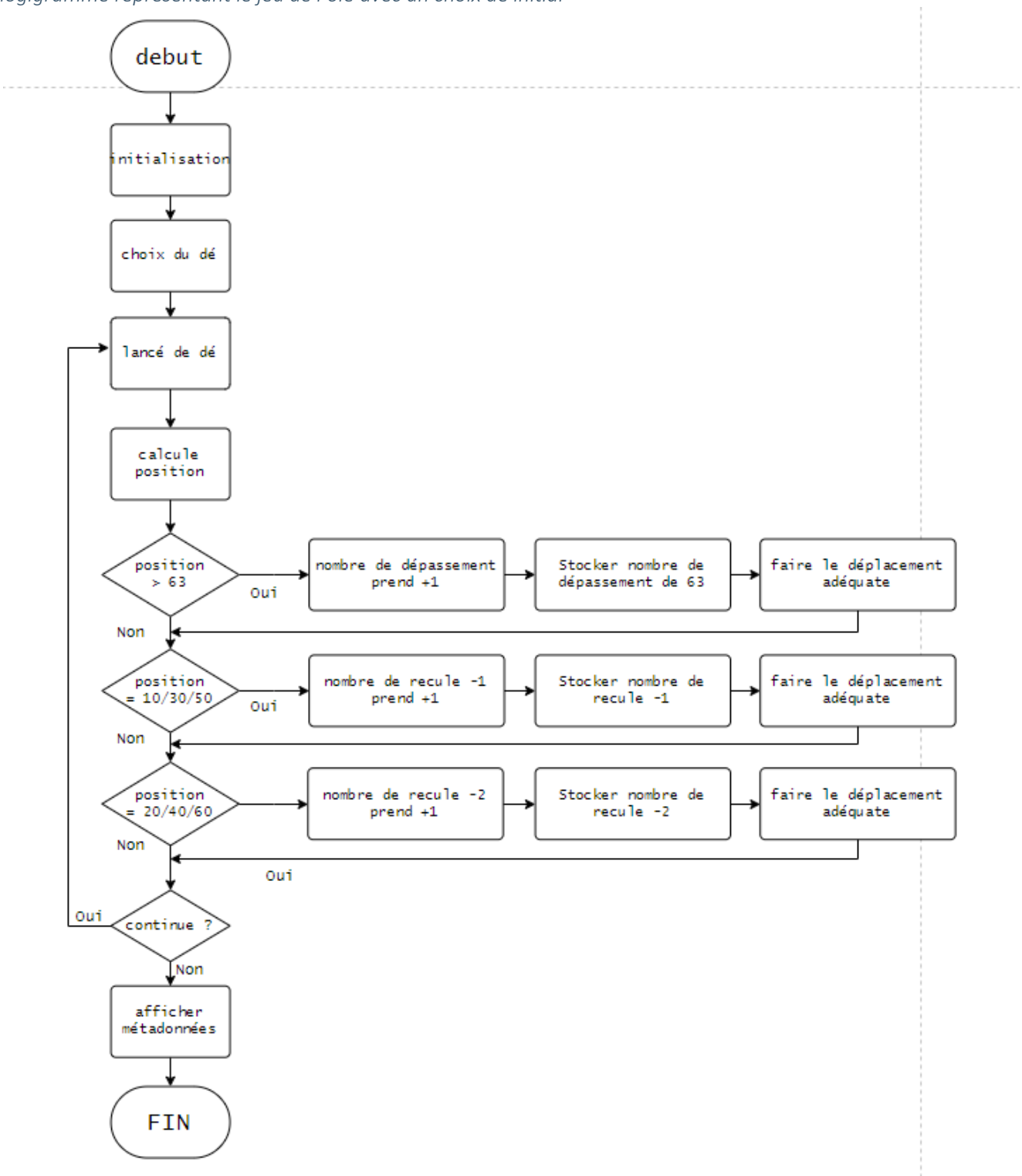
La mort subite :

Si un joueur dépasse 5 fois la case 63 (ou la case 0 dans le cas du joueur 2), il est éliminé.
Pour créer cette règle, nous nous sommes appuyé sur l'Algorithme du jeu de base.
En effet dans le jeu de base nous avions déjà fait un compteur pour savoir le nombre de fois ou un joueur n'a pas réussi à tomber sur la case d'arrivée et l'a dépassé. Nous avons ajouté une condition qui fait que lorsque ce chiffre arrive à 5, le joueur est éliminé.

Mode de jeu D6 ou D8 :

Notre dernière règle consiste à créer un deuxième mode de jeu afin de varier le déroulé des parties et le rythme du jeu.
Au départ il est demandé au joueur de rentré une commande pour déterminer le mode de jeu.
Il lui est explicitement demandé de choisir le mode 1 ou le mode 2, le mode 1 se joue avec un dé à 6 faces, le mode 2 avec un dé à 8 faces.
Nous avons donc dû faire évoluer la fonction RANDOM afin que l'intervalle corresponde au mode de jeu choisi. Ainsi l'intervalle correspond désormais aux valeurs (1 ; *Maxdé*). Maxdé étant une variable qui prends la valeur 6 si le joueur choisis le mode 1 et la valeur 8 si le joueur choisis le mode 2.
De cette manière, le premier joueur choisis le mode de jeu qui sera valide pendant toute la durée de la partie.

logigramme représentant le jeu de l'oie avec un choix dé initial



3/ LES VARIABLES :

Pour la création de ces règles nous avons dû créer de nouvelles variables que vous trouverez ci-dessous.

- essaiescape : Compteur pour déterminer à combien d'essai est le prisonnier qui essaye de sortir.

- maxdé : Détermine la valeur max des jets de dés en fonction du mode de jeu voulu.

- modejeu : Pour déterminer si le jeu est en mode D6 ou en mode D8.

- déprison : Valeur du dé servant à sortir de la prison, si 4+ le joueur est libéré.

Par ailleurs, nous avons doublés toutes les variables (y compris celles de l'algorithme principale) afin qu'elles soient applicables au joueur 1 et au joueur 2 (Exemple : pos1 et pos2, dé1 et dé2, moyenne1 et moyenne2…).

II/ ALGORITHME DES NOUVELLES REGLES

1/DEUXIEME JOUEUR

Données

```
Entier : dé2, pos2, nbl2, totaldé2, nbd2, nbdd2, nbrd2
Réel : moyenne2
Bool : cj
```

Fin_Données

DEBUT

```
dé2 <- 0
pos2 <- 63
nbl2 <- 0
totaldé2 <- 0
nbd2 <- 0
nbdd2 <- 0
nbrd2 <- 0
moyenne2 <- 0

SAISIR "Voulez-vous jouer ?", cj

TANT QUE cj = VRAI ET pos2 > 0

        totaldé2 <- totaldé2 + dé2
        pos2 <- pos2 + dé2
        nbl2 <- nbl2 + 1
        moyenne2 <- totaldé2 / nbl2

        SI pos2 < 0
                ALORS nbrd2 <- nbrd2 + 1
                        pos2 <- 0 - (pos - 0)
        FIN_SI
        SI pos2 = 10 OU pos2 = 30 OU pos2 = 50
                ALORS nbd2 <- nbd + 1
                        pos <- pos + 1
                SINON SI pos2 = 20 OU pos2 = 40 OU pos2 = 60
                        ALORS nbdd2 <- nbdd2 + 1
                                pos2 <- pos2 + 2
                        SINON SI pos2 = 0
                                ALORS AFFICHER "Bravo, vous avez gagné !"
                                        cj <- FAUX
        FIN_SI
        SI cj = VRAI
                ALORS AFFICHER "Le joueur 2 est à la case ", pos2, "."
                SAISIR  "Voulez-vous continuer ? VRAI/FAUX", cj
        FIN_SI

FIN_TANT_QUE

AFFICHER "Partie terminée !"
AFFICHER "----------Données du joueur 2----------"
AFFICHER "Le joueur a lancé le dé ", nbl2, " fois"
AFFICHER "Le joueur a reculé ", nbd2, " fois à cause des dizaine"
AFFICHER "Le joueur a reculé ", nbdd2, " fois à cause des double-dizaine"
AFFICHER "Le joueur a reculé ", nbrd2, " fois à cause du dépassement de la case 63"
AFFICHER "La moyenne des lancés de dé du joueur 2 est de ", moyenne2, "."
```

FIN

2/LA PRISON

```
déprison <- 0
essaiescape <- 0

SI pos = 31 ou pos = 41
        TANT QUE déprison < 4 FAIRE
                SELON essaiescape FAIRE

                        0 : AFFICHER "Vous avez encore 5 essais pour sortir de prison"
                                        essaiescape <- essaiescape + 1
                        1 : AFFICHER "Vous avez encore 4 essais pour sortir de prison"
                                        essaiescape <- essaiescape + 1
                        2 : AFFICHER "Vous avez encore 3 essais pour sortir de prison"
                                        essaiescape <- essaiescape + 1
                        3 : AFFICHER "Vous avez encore 2 essais pour sortir de prison"
                                        essaiescape <- essaiescape + 1
                        4 : AFFICHER "Vous avez encore 1 essai pour sortir de prison"
                                        essaiescape <- essaiescape + 1
                SINON
                        AFFICHER "Vous avez épuisé tous vos essais !"
                                cj <- FAUX
            déprison <- RANDOM(1;6)

            SI déprison >= 4

              ALORS AFFICHER "Félicitations ! Vous sortez de prison et avancer de 1"
              pos <- pos + 1

            FIN_SI

        Fin_TANT QUE
FIN_SI
```

3/LA TELEPORTATION

```
SI pos = 42
        ALORS pos <- RANDOM(0;63)
FIN_SI
```

4/LA MORT SUBITE

```
SI nbrd = 3
        ALORS AFFICHER "Vous avez perdu !"
        cj <- FAUX
FIN_SI
```

5/CHOIX DU MODE DE JEU

```
maxdé <- 0
modejeu <- 0

SAISIR "Si vous voulez jouer en mode D6 tapez 1, si vous voulez jouer en mode D8 tapez 2", modeje
TANT QUE modejeu < 1 ou modejeu > 2 FAIRE
            SELON modejeu FAIRE
                    1 : AFFICHER "Vous jouez avec un dé 6"
                        maxdé <- 6
                    2 : AFFICHER "Vous jouez avec un dé 8"
                        maxdé <- 8
             SINON AFFICHER "Entrez une valeur logique !"
Fin_TANT_QUE

Dé <- RANDOM(1 ;maxdé)
```

PARTIE 3 : ALGORITHME GENERAL AVEC TOUTES LES REGLES

```
Algo Jeu_de_loie

Données

    Entier : dé, dé2, déprison, déprison2, pos, pos2, nbl, nbl2, totaldé, totaldé2,
             nbd, nbd2, nbdd, nbdd2, nbrd, nbrd2, valeurdé, valeurdé2, essaiescape, essaiescape2, modejeu, maxdé
    Réel : moyenne, moyenne2
    Bool : cj

Fin_Données


Début

    pos <- 0
    pos2 <- 63
    nbl <- 0
    nbl2 <- 0
    totaldé <- 0
    totaldé2 <- 0
    nbd <- 0
    nbd2 <- 0
    nbdd <- 0
    nbdd2 <- 0
    nbrd <- 0
    nbrd2 <- 0
    moyenne <- 0
    moyenne2 <- 0
    essaiescape <- 0
    essaiescape2 <- 0
    modejeu <- 0
    maxdé <- 0

    SAISIR "Voulez-vous jouer ?", cj
    SAISIR "Si vous voulez jouer en mode D6 tapez 1, si vous voulez jouer en mode D8 tapez 2", modejeu

  TANT QUE modejeu < 1 ou modejeu > 2 FAIRE

  SELON modejeu FAIRE
        1 : AFFICHER "Vous jouez avec un dé 6"
            maxdé <- 6

        2 : AFFICHER "Vous jouez avec un dé 8"
            maxdé <- 8

        SINON
            AFFICHER "Entrez une valeur logique !"
  Fin_TANT

TANT QUE cj = VRAI ET pos < 63 ET pos2 > 0 FAIRE

    dé <- RANDOM(1;modejeu)
    dé2 <- RANDOM(1;modejeu)
    totaldé <- totaldé + dé
    totaldé2 <- totaldé2 + dé2
    pos <- pos + dé
    pos2 <- pos2 - dé2
    nbl <- nbl + 1
    nbl2 <- nbl2 + 1
    moyenne <- totaldé / nbl
    moyenne2 <- totaldé2 / nbl2
```

```
SI pos = 31 ou pos = 51 ET essaiescape < 5
        ALORS AFFICHER "Le joueur 1 est en prison à la case ", pos, " ! Faites un lancé entre 4 et 6 pour sortir"

          SI déprison < 4 FAIRE

        SELON essaiescape FAIRE

                0 : AFFICHER "Vous avez encore 5 essais pour sortir de prison"
                        essaiescape <- essaiescape + 1
                1 : AFFICHER "Vous avez encore 4 essais pour sortir de prison"
                        essaiescape <- essaiescape + 1
                2 : AFFICHER "Vous avez encore 3 essais pour sortir de prison"
                        essaiescape <- essaiescape + 1
                3 : AFFICHER "Vous avez encore 2 essais pour sortir de prison"
                        essaiescape <- essaiescape + 1
                4 : AFFICHER "Vous avez encore 1 essai pour sortir de prison"
                        essaiescape <- essaiescape + 1
                SINON
                    AFFICHER "Vous avez épuisé tout vos essais !"
                    cj <- FAUX

                déprison <- RANDOM(1;6)

                SI déprison >= 4

                    ALORS AFFICHER "Félicitations ! Le joueur 1 sort de prison et avancer de 1"
                    pos <- pos + 1

                FIN_SI
          FIN_SI

  SI pos2 = 31 ou pos2 = 51 ET essaiescape2 < 5
      ALORS AFFICHER "Le joueur 2 est en prison à la case ", pos, " ! Faites un lancé entre 4 et 6 pour sortir"

  SI déprison2 < 4 FAIRE

SELON essaiescape2 FAIRE

        0 : AFFICHER "Vous avez encore 5 essais pour sortir de prison"
                essaiescape2 <- essaiescape2 + 1
        1 : AFFICHER "Vous avez encore 4 essais pour sortir de prison"
                essaiescape2 <- essaiescape2 + 1
        2 : AFFICHER "Vous avez encore 3 essais pour sortir de prison"
                essaiescape2 <- essaiescape2 + 1
        3 : AFFICHER "Vous avez encore 2 essais pour sortir de prison"
                essaiescape2 <- essaiescape2 + 1
        4 : AFFICHER "Vous avez encore 1 essai pour sortir de prison"
                essaiescape2 <- essaiescape2 + 1
        SINON
            AFFICHER "Vous avez épuisé tout vos essais !"
            cj <- FAUX

        déprison2 <- RANDOM(1;6)

        SI déprison2 >= 4

            ALORS AFFICHER "Félicitations ! Le joueur 2 sors de prison et avancer de 1"
            pos <- pos + 1

        FIN_SI
  FIN_SI
```

```
        SI pos = 10 OU pos = 30 OU pos = 50 OU pos2 = 10 OU pos2 = 30 OU pos2 = 50
                ALORS AFFICHER "Vous reculez de 1 case"
                    nbd <- nbd + 1
                    pos <- pos - 1
                    nbd2 <- nbd2 + 1
                    pos2 <- pos2 + 1

                SINON SI pos = 20 OU pos = 40 OU pos = 60 OU pos2 = 20 OU pos2 = 40 OU pos2 = 60
                        ALORS AFFICHER "Vous reculez de 2 cases"
                            nbdd <- nbdd + 1
                            pos <- pos - 2
                            nbdd2 <- nbdd2 + 1
                            pos2 <- pos2 + 2
        FIN_SI


    AFFICHER "Le joueur 1 est la case ", pos, "  et le joueur 2 est à la case ", pos2, "."

    SI cj = VRAI
        ALORS SAISIR "Voulez-vous continuer ? Vrai/Faux", cj
    FIN_SI



    Fin_TANT QUE

    AFFICHER "Partie terminée !"

    AFFICHER "----------------------Données du joueur 1----------------------"

    AFFICHER "Le joueur 1 a lancé le dé ", nbl, " fois"
    AFFICHER "Le joueur 1 a reculé ", nbd, " fois à cause des dizaine"
    AFFICHER "Le joueur 1 a reculé ", nbdd, " fois à cause des double-dizaine"
    AFFICHER "Le joueur 1 a reculé ", nbrd, " fois à cause du dépassement de la case 63"
    AFFICHER "La moyenne des lancés de dé du joueur 1 est de ", moyenne, "."

    AFFICHER "----------------------Données du joueur 2----------------------"

    AFFICHER "Le joueur 2 a lancé le dé ", nbl2, " fois"
    AFFICHER "Le joueur 2 a reculé ", nbd2, " fois à cause des dizaine"
    AFFICHER "Le joueur 2 a reculé ", nbdd2, " fois à cause des double-dizaine"
    AFFICHER "Le joueur 2 a reculé ", nbrd2, " fois à cause du dépassement de la case 0"
    AFFICHER "La moyenne des lancés de dé du joueur 2 est de ", moyenne2, "."
FIN
```

CONCLUSION

En conclusion nous souhaitons vous dire que nous avons beaucoup aimé travailler sur ce projet. S'il nous paraissait ambitieux dans un premier temps, nous avons pu ensemble identifier les problèmes à résoudre. Nous avons appris à travailler de concert et nous sommes satisfait du résultat.

Nous avons apprécié ce travail d'équipe, avoir des points de vue différents est un réel atout pour ce genre de projet. Nous avons aussi aimé le fait de pouvoir vous proposez nos idées afin de pouvoir améliorer le jeu.

C'est un défi que nous avons trouvé aussi bien intéressant qu'amusant. Amusant car nous pouvions proposer ce que nous voulions que se soit afin d'aider ou handicaper le joueur. Et intéressant car nous avons pu construire nos algorithmes dans un cadre original et concret ce qui nous a permis de tester leurs limites.

Nous tenions à compiler l'entièreté de notre travail dans une version globale. Nous avons placé des captures d'écran du logiciel *Visual Studio Code,* que nous avons utilisé pour mettre en forme notre algorithme final, afin que la lecture soit plus lisible.

C'est donc avec plaisir que nous vous proposons notre version final du je l'oie incorporant toutes les règles que nous avons imaginées.

Nous espérons que le projet vous a plu et nous avons hâte de retravailler sur ce type de projet dans le futur.