Rapport du projet d'Algorithmique

Développement du jeu **AWALE**
(Version classique et version multi-joueurs)

Réalisé par :

Mme Hammar Meriem : IATIC3
M. Mourao Dany : IATIC3
M. Teruel Raphael : IATIC3
M. Roumec Bryan : IATIC3
M. Roussel Quentin : IATIC3

Projet IATIC3 effectué du 24/11/2018 au 31/01/2019

Présentation du Projet Algorithmique

Développement du Jeu AWELE

Etudiants en IATIC3
Dhekra Abouda – Encadrant

Soutenu à l'ISTY le 26/11/208

PLAN

I- Présentation du Jeu

Dans le cadre de notre projet demandé, nous nous intéressons à coder un programme en langage C permettant de modéliser le jeu suivant : Awélé. Ce jeu comporte plusieurs versions et règles de jeu, variant donc d'un site à un autre. Il comporte un plateau de jeu composé de 12 trous, dont 6 sont pour vous (en bas) et les 6 autres pour l'adversaire (en haut). De plus, chaque trou est composé de 3 graines au début, ce qui fait au total 36 graines à jouer en début de partie.

Partie fonctionnelle du jeu :

Le joueur débutant la partie est tiré aléatoirement puis les joueurs jouent à tour de rôle. Pour pouvoir jouer, le joueur doit prendre l'ensemble des graines contenues dans l'un de ses six trous disponibles et les semer à sa droite. Il peut également éventuellement, au cours de la partie, faire un tour complet du plateau (12 graines ou plus), mais devra alors le trou où il a sélectionné les graines.

Pour gagner des points (graines), il faut que le joueur récolte ce qu'il a semé. Lorsque le joueur sème ses graines, on doit regarder le dernier trou où l'on pose la graine. Ainsi, si la dernière case correspond à un trou de l'adversaire et qu'il y a 2 ou 3 graines, on peut les récolter, ce qui comptabilisera notre nombre de points. On les enlève alors du plateau pour ajouter ces graines à notre score. Autre règle particulière : si l'adversaire du joueur a toutes ses cases vides, le joueur doit (s'il le peut) jouer un coup qui dépose une graine dans le camp adverse.

Fin de partie :

Il y a une fin de partie lorsque le nombre de graines récoltées par un joueur est au moins égal à 19, ou lorsque toutes les cases d'un joueur sont vides. On doit alors compter le nombre de graines détenues et faire le bilan des graines récoltées. Le joueur qui en récolte le plus gagne la partie. De plus, lorsque le jeu tourne en boucle (depuis plus de 20 tours), c'est-à-dire qu'aucun des joueurs n'arrive à récolter de graines, le jeu est arrêté et on fait le bilan encore une fois.

II- Cahier des charges

1- Réalisation :

Jeu Awalé à réaliser par groupe de 5 étudiants en IATIC3 à l'Institut Scientifique et Techniques des Yvelines (Vélizy-Villacoublay).

Pour la réalisation de ce jeu, nous avons donc décidé de se voir en groupe à l'ISTY durant nos heures libres et de partager nos fichiers sur GitHub pour avoir accès aux différents fichiers. On communique également via l'intermédiaire de réseaux sociaux tels que WhatsApp ou Messenger.

2- Outils :

Bibliothèque SDL pour la partie graphique et le jeu est donc écrit en C.

3- Missions :

Répartition des fichiers :
Faire un fichier ne contenant que la partie graphique.
Faire un autre fichier ne contenant que la partie du jeu.
Faire un autre fichier contenant ce qui relie les deux (contrôleur).

Ainsi, Bryan, Dany et Meriem devront se consacrer à la partie graphique, tandis que Raphaël et Quentin se consacreront à la partie du jeu. Bryan devra coder la partie contrôleur (celle qui rassembles les deux parties). Enfin, Dany et Quentin réaliseront un rapport tout au long de notre avancée. Cette répartition des tâches nous permet donc d'avoir des parties équitables et devra également nous permettre d'avancer le plus rapidement possible dans ce projet.

Partie Graphique :

Utiliser la SDL (image – tile maping ou non)

Faire un menu avec des images pour demander à l'utilisateur s'il veut jouer contre l'ordinateur ou l'utilisateur.

Lorsque l'utilisateur a choisi, il doit pouvoir enregistrer son nom avant de jouer.

Faire une fonction pour mettre en place une grille de jeu avec 2 lignes et 6 colonnes ou l'on pourra observer les 3 graines dans chaque case.

Réalisation d'un menu pour que l'utilisateur ait le choix de jouer contre l'ordinateur ou contre un autre joueur (1vs1).

Partie du jeu :

Partie globale du jeu :

Il faut utiliser un tableau 2D de type int pour pouvoir stocker 3 graines dans chaque case du tableau.

⇨ int tableau 2d [2][6] // 2 lignes et 6 colonnes

Pouvoir choisir la case que l'on veut jouer et déplacer les graines une par une dans chaque case (dans le sens anti horaire).

- Faire une fonction où l'on alterne le tour des deux joueurs.

Fin de partie :

Il y a fin de partie lorsque :

1. L'un des joueurs n'a plus de pions à sa disposition.
2. L'un des joueurs a mangé 19 pions.

} GAGNER/PERDRE

3. Le nombre de tours sans récolte est de 20. => MATCH NUL

Pour réaliser cette fonction, on peut retourner une valeur (RIEN=0, J1 GAGNE = 1, J2 GAGNE = 2, MATCH NUL = 3) et donc continuer le jeu ou l'arrêter au tour d'après (on continue lorsque la valeur est égale à 0).

Ainsi, on enregistre le score et les noms des joueurs dans un fichier.

Jouer contre l'ordinateur :

Faire une IA plus ou moins développé (voire différents niveaux à proposer), pour cela il faudrait :

Teste si une case peut être joueur
Test si des graines peuvent être manger sur notre partie du terrain
Teste si on peut manger une ou plusieurs graines et choisir et la case qui rapporte le plus
Sinon jouer au Hasard

Partie Contrôleur :

La partie contrôleur sert à faire le lien entre la partie graphique et partie jeu. Il y a aussi la sauvegarde des pseudos et des scores dans un fichier à réaliser et toutes les fonctions un peu annexes qui ne servent pas forcément au jeu comme par exemple un menu pause.

Gestion de fichier :

Lorsque l'on choisit son mode de jeu (IA ou 1v1), on doit rentrer son nom dans un menu qui va apparaitre à l'écran. Ce nom apparaitra donc alors en bas de l'écran accompagné de votre score. De plus, à la fin de la partie, le nom et le score seront enregistrés dans un fichier que l'on pourra ouvrir pour voir les 5 derniers scores.

Ainsi, il faut pouvoir :

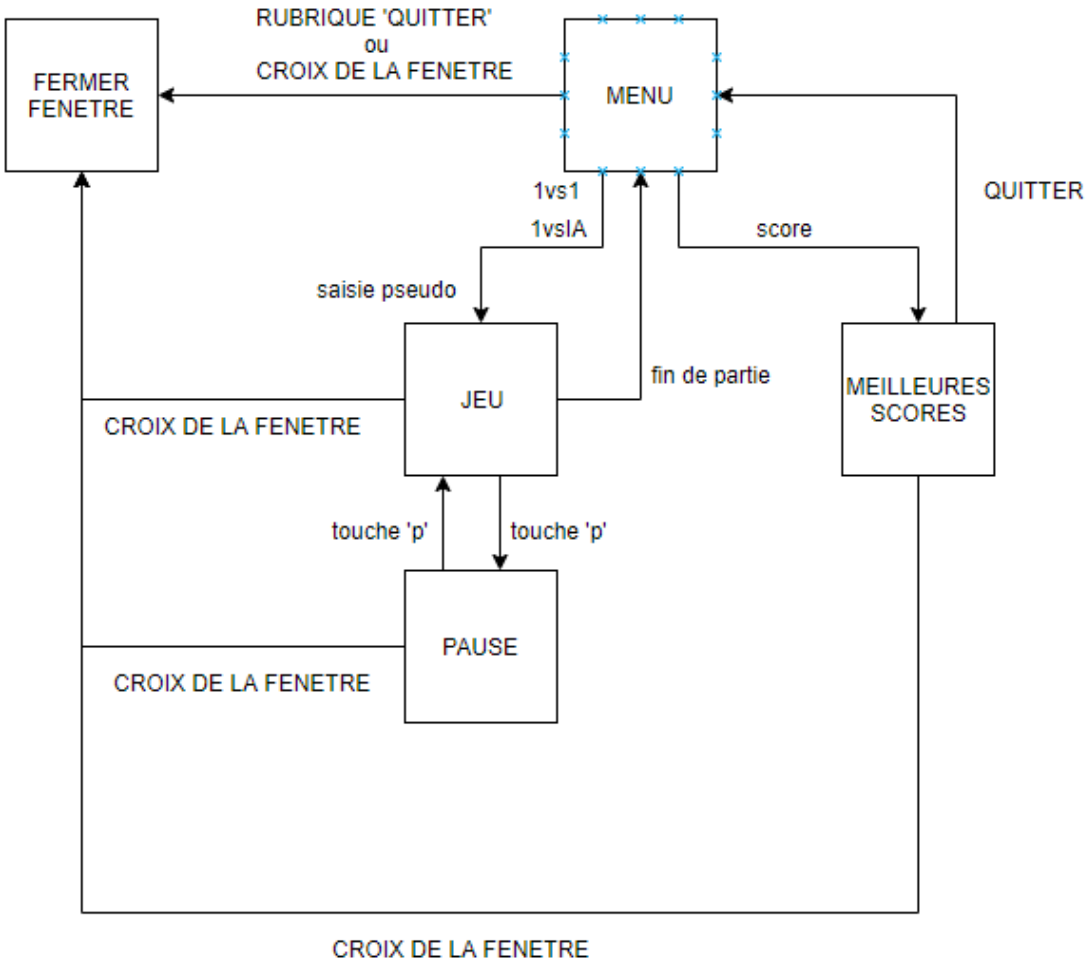
Lire fichier existant
Trier les scores
Ecrire le score dans un fichier

III. Modélisation

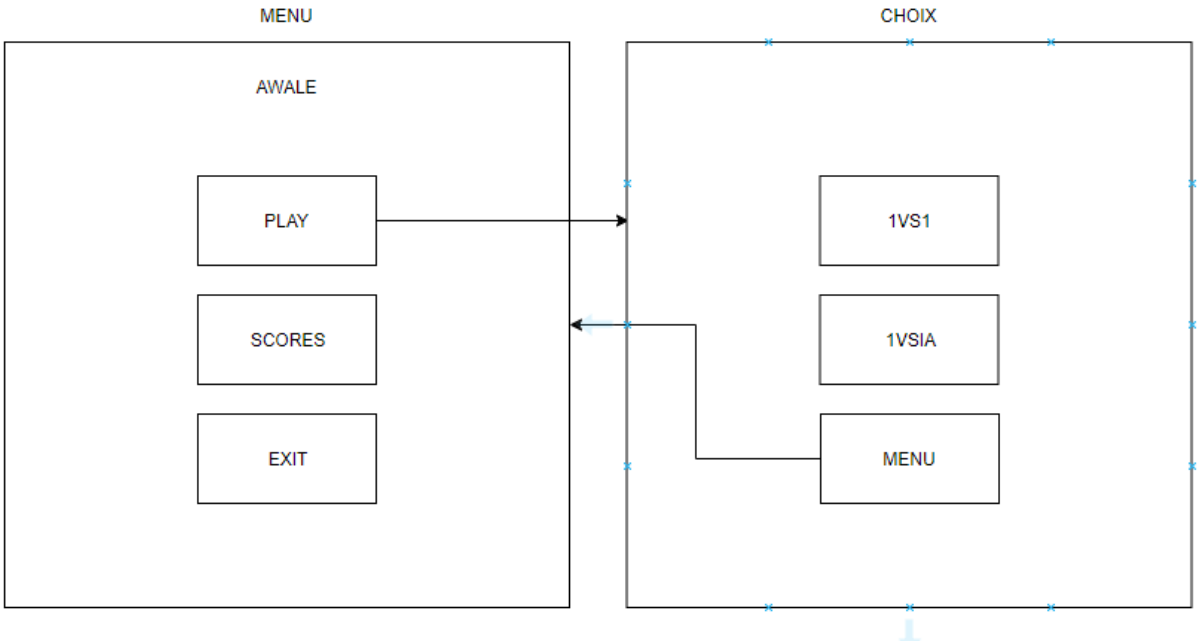
Partie modélisation du projet :

On initialise le plateau de jeu avec 2 lignes, 6 colonnes et 3 graines dans chaque case et le score des deux joueurs à 0.

Modélisation des fenêtres de notre jeu :

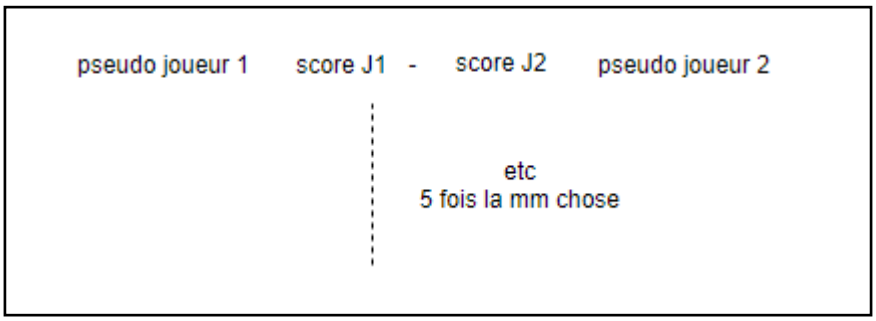


Modélisation du MENU :



Modélisation des SCORES :

On affichera qu'une fenêtre et les 5 derniers scores enregistrées. En effet, il y aura donc le score des deux joueurs et leurs pseudonymes.



Pour la modélisation du jeu en 1vs1 (à peu près la même chose donc pour l'IA) :

On alterne les tours des joueurs pour que chacun joue l'un après l'autre.

i = nb_tour&0b1

Cette syntaxe permet d'écrire nb_tour en binaire et de savoir directement si le nombre est pair ou impaire et nous permet de gagner du temps puisqu'il n'y a pas de comparaison (if (condition) else).
Si le nb_tour de type int, est pair, alors nb_tour&0b1 renvoie 0 et donc i = 0, ce qui correspond au joueur 2 de jouer.
Si le nb_tour de type int est impair, alors nb_tour&0b1 renvoie 1 et donc i = 1, ce qui correspond au joueur 1 de jouer.

Pour chaque joueur, on a la possibilité de choisir l'une de nos six cases si elle contient une ou plusieurs graines. Si une case vide est sélectionnée, aucune action ne se produit, il faut donc choisir une case dite « remplie ».

Une fois une case correcte choisit, on passe à la distribution des graines une par une dans une case après l'autre en partant vers la droite (sens antihoraire). Les cases suivantes se remplissent donc avec une graine supplémentaire.

On passe ensuite aux conditions de ramassage des graines. S'il on dépose notre dernière graine dans une case de notre adversaire, on regarde si cette case contient deux ou trois graines (après dépôt de notre graine). Si ce n'est pas le cas, rien ne se passe, on finit simplement notre distribution et on passe au tour de l'adversaire.
Sinon, on a la possibilité de ramasser toutes les graines de la case et de les retirer du jeu, cela fait augmenter notre score du nombre de graine ramassées. De plus, une fois ce ramassage validé, on regarde si la case précédente contient également deux ou trois graines, si ce n'est pas le cas, on ne fait rien de plus, sinon on retire également ces graines du jeu et on les additionne au score du joueur. Puis les tours s'enchainent à tour de rôle.

Les conditions d'arrêt sont les suivantes :

- si le score d'un joueur est supérieur ou égal à 19, donc que le joueur a ramassé 19 graines ou plus, alors il gagne la partie.
- si aucune récolte (ramasser des graines dans la partie de l'adversaire) n'est effectuée pendant 20 tours alors la partie s'arrête et on regarde le score des deux joueurs, celui avec le score le plus élevé gagne, si les scores sont égaux, il y a match nul.
- si toutes les cases d'un des deux joueurs sont vides, la partie s'arrête et on compare également les scores.

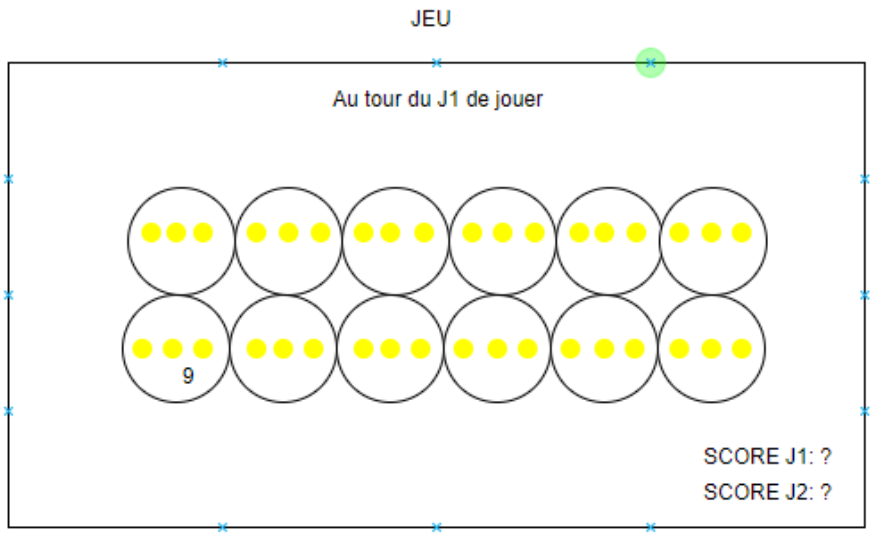
Pour la modélisation du jeu en Joueur vs IA :

IA niveau1 : l'IA joue aléatoirement n'importe quelle case jouable

(À faire) IA niveau2 : on crée une IA capable de savoir s'il peut récolter des graines chez l'adversaire ou si l'adversaire peu récolter, on sauvegarde le résultat dans un tableau de structure. L'IA joue avec en priorité les case qui peuvent être possiblement manger, si il y en a pas on joue les case qui peuvent rapporter des point, si il y pas de graine a sauver ni à manger, l'ordinateur joue aléatoirement

Pour la partie graphique :

On a choisi d'utiliser la bibliothèque SDL.
Pour la partie du jeu (que ce soit en 1vs1 ou contre l'ordinateur), on affiche deux images d'un rectangle contenant 6 ronds. L'une au-dessus de l'autre au milieu du terminal. Les graines sont des ronds jaunes placés dans les cases (ronds définit précédemment). On affiche également le tour de chaque joueur et les score des deux joueurs en bas à droite de la fenêtre. De plus, si le nombre de graines est strictement supérieur 6 dans l'une des cases, on affiche un numéro en bas à droite ce la case affichant le nombre de graines disponibles. On peut voir ici dans le schéma ci-dessous écrit 9 mais il faudrait qu'il y ait 6 graines dans la case pour afficher 9.

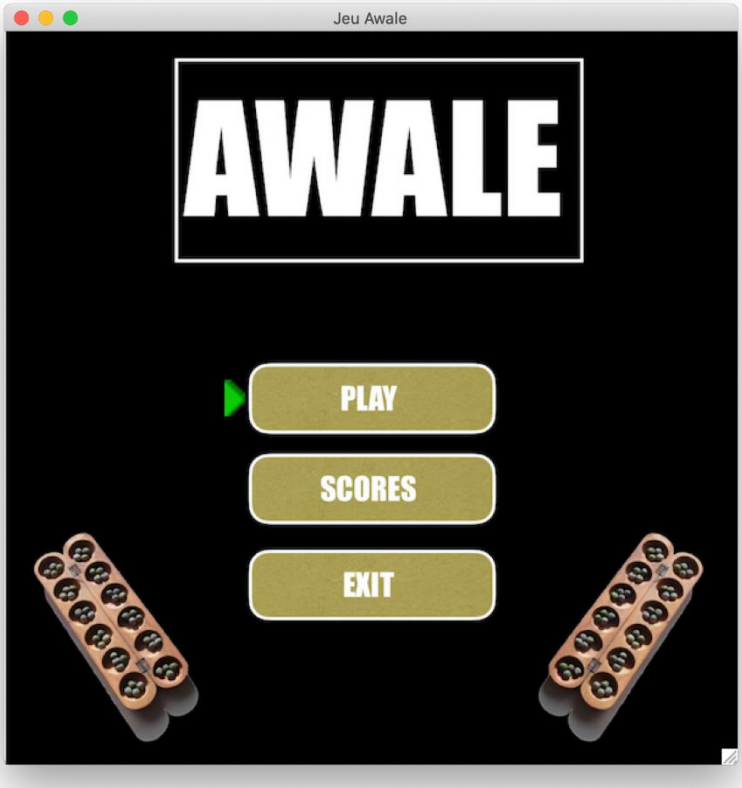


IV. Avancement

Cette partie permet de voir ce qu'on a donc réalisé à partir de notre modélisation faite partie III.

Partie Graphique :

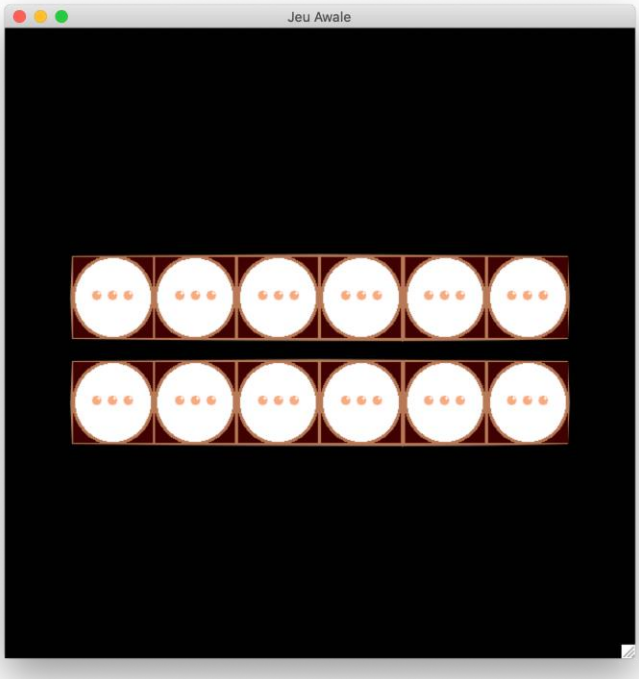
Le Menu est fonctionnel et nous permet de choisir grâce à un curseur vert (avec les touches directionnelles) les différentes options qui se présentent à nous. En effet, on peut choisir de jouer, ce qui nous demandera si l'on veut jouer contre l'ordinateur ou en 1v1. On peut fermer la fenêtre avec la croix ou la rubrique 'EXIT'. Cependant, la partie 'SCORES' n'a pas encore été réalisé pour le moment.



Taper son pseudo :
On peut taper jusqu'à 12 caractères et effacer si on a fait une erreur lors de la saisie. Pour valider son pseudonyme, il faut taper la touche 'entrée'. Ensuite, une autre fenêtre identique à celle-ci demande encore le pseudo pour le deuxième joueur.

Partie Jeu :

Concernant, la partie du jeu graphiquement, on n'a toujours pas inséré le message en haut de la fenêtre pour avertir le joueur 1 ou 2 de jouer. De plus, les scores et pseudonymes en bas à droite ne sont également pas insérés. Et enfin, on essaye toujours de lier notre partie jeu (semer les graines, récolte des graines et points etc…) à notre partie graphique.



Fonctionnement :

Le jeu joueur contre joueur fonctionne très bien, la partie graphique est créé et va être améliorée par la suite.

Liste des choses à faire et améliorer :

De nombreuses choses restent à faire, d'une part nous devons améliorer la partie graphique :

- visualiser le score
- améliorer l'avancement des graines (graphiquement)
- le ramassage des graines (graphiquement)
- la rentrée des pseudonymes
- ajout d'un menu pause
- ajout de musique/effets sonores
- sauvegarde de score/pseudo
- répartir le code (jeu) en plusieurs fonctions

Pour le fonctionnement du jeu, il reste à approfondir la partie joueur contre IA, l'implémenter, la tester et si possible la rendre plus performante. De plus, il faut ajouter la fonction de sauvegarde des pseudonymes et des scores. Également, implémenter le menu principal.
On désire rendre notre programme en général moins lourd, le plus optimisé possible.

V. Difficultés rencontrées

On rencontre des difficultés avec la bibliothèque SDL puisque les couleurs ne sont pas respectées. En effet, en voulant insérer une couleur blanche en fond, on se retrouve avec une couleur jaune. Puis, dès que l'on réduit la fenêtre du jeu et qu'on la réaffiche, elle se met bien en blanc. Pour pallier ce problème, on a décidé de mettre un fond noir. On a également des difficultés pour bien faire la transition entre la partie jeu et graphique, cela vient du fait que la réunion des fonctions codées de chacun doit être réadaptés pour qu'elles fonctionnent entre elles.

VI. Bilan

La partie du jeu est quasiment terminé (à part IA, pas encore « intelligente »), il nous reste à compléter la partie graphique puis améliorer notre jeu avec des sons, voire gagner de l'espace mémoire. Ce projet d'algorithmique sur le jeu Awalé par groupe de 5 personnes, nous a donc permis de travailler en groupe et de bien se répartir les tâches entre nous.