HELHa
Haute École Louvain en Hainaut

Rapport de Projet (Session Août)

Bachelier en informatique de gestion
Bloc 1

Projet - Labyrinthe

(Mr Alary, Mr Kaouass, Mr Legrand, Mr Steux)

Zarzycki Alexis

Verly Noah

Année académique 2022 – 2023

ÉCONOMIQUE

Table des matières

Introduction

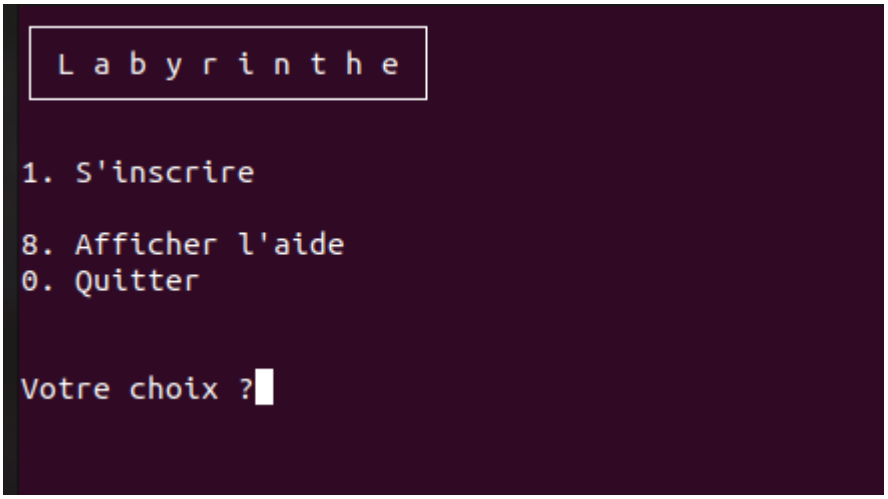
En cette année académique 2022-2023, nous avons dû pour le cours de Projet, créer un jeu. Ce jeu est un labyrinthe, pour pouvoir utiliser celui-ci, l'utilisateur devra s'inscrire. Il pourra ensuite, afficher les scores, démarrer une partie classique, ou bien démarrer un mini-jeu « attraper le monstre ».

Les objectifs du projet

- Développer un projet sur base d'un squelette
- Rédiger des tests fonctionnels
- Développer des tests unitaires
- Concevoir un projet en équipe grâce à Git
- Utiliser la matière apprise en classe
- Présenter et défendre un projet

Mode d'emploi de l'application

Lancement de l'application



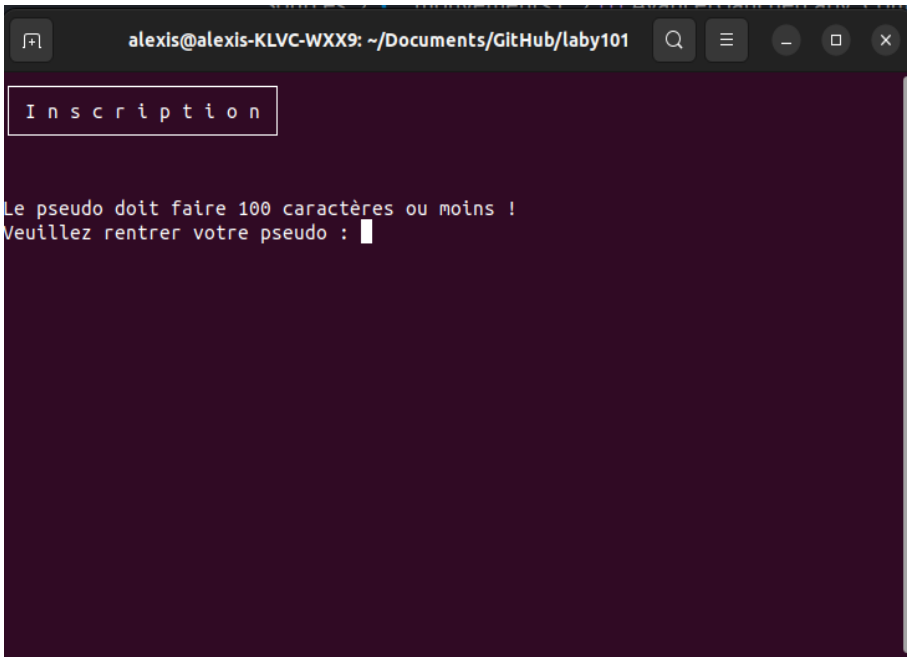
Dès le lancement de l'application, nous avons un menu qui s'affiche.

Celui-ci nous propose de s'inscrire, afficher l'aide ou quitter l'application.

Si vous entrez 8, voici ce qui s'affiche :

Si vous entrez 0, voici ce qui s'affiche :

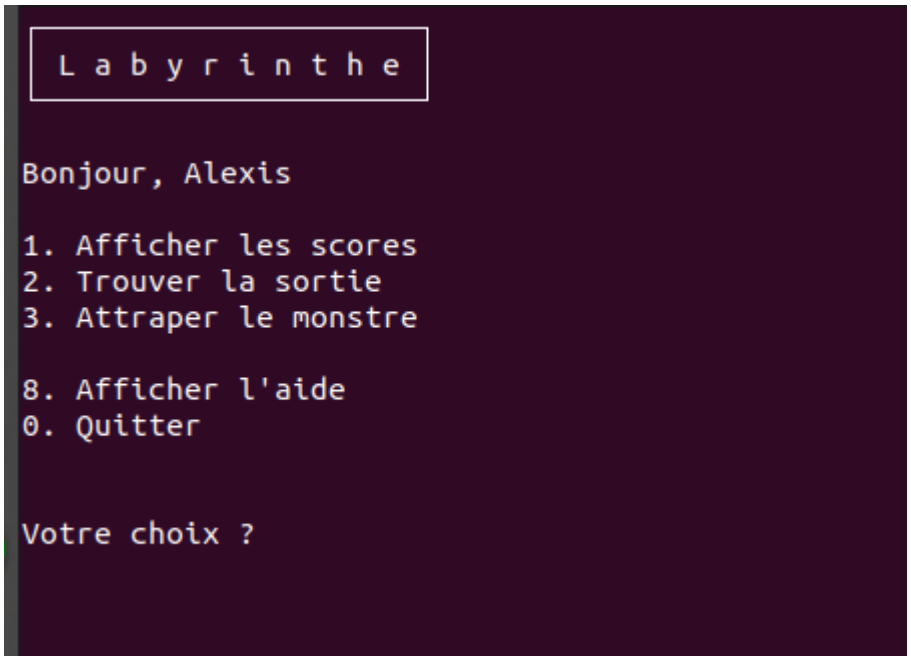
Si vous entrez 1, voici ce qui s'affiche :



Ensuite, le programme vous demande d'entrer un pseudo.

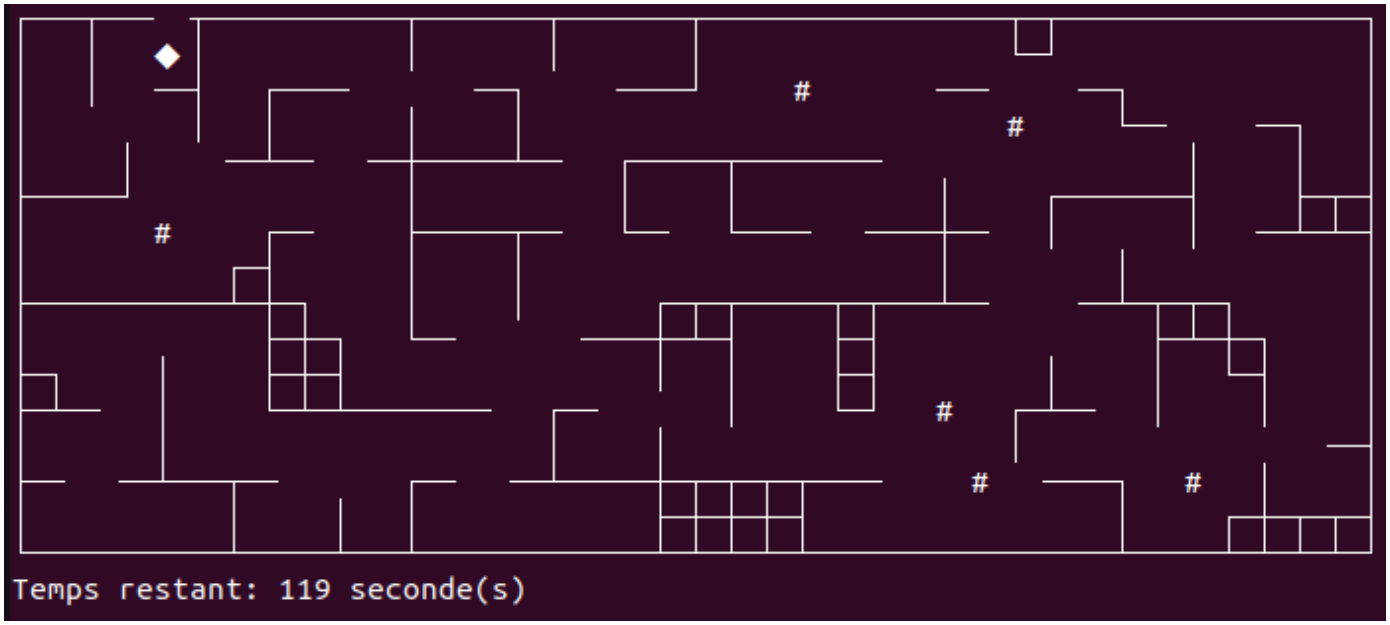
Si le pseudo fait plus de 100 caractères, l'utilisateur devra le saisir à nouveau.

Une fois que le pseudo est correct, un nouveau menu s'affiche :



Si vous entrez 1, le tableau des scores s'affiche :

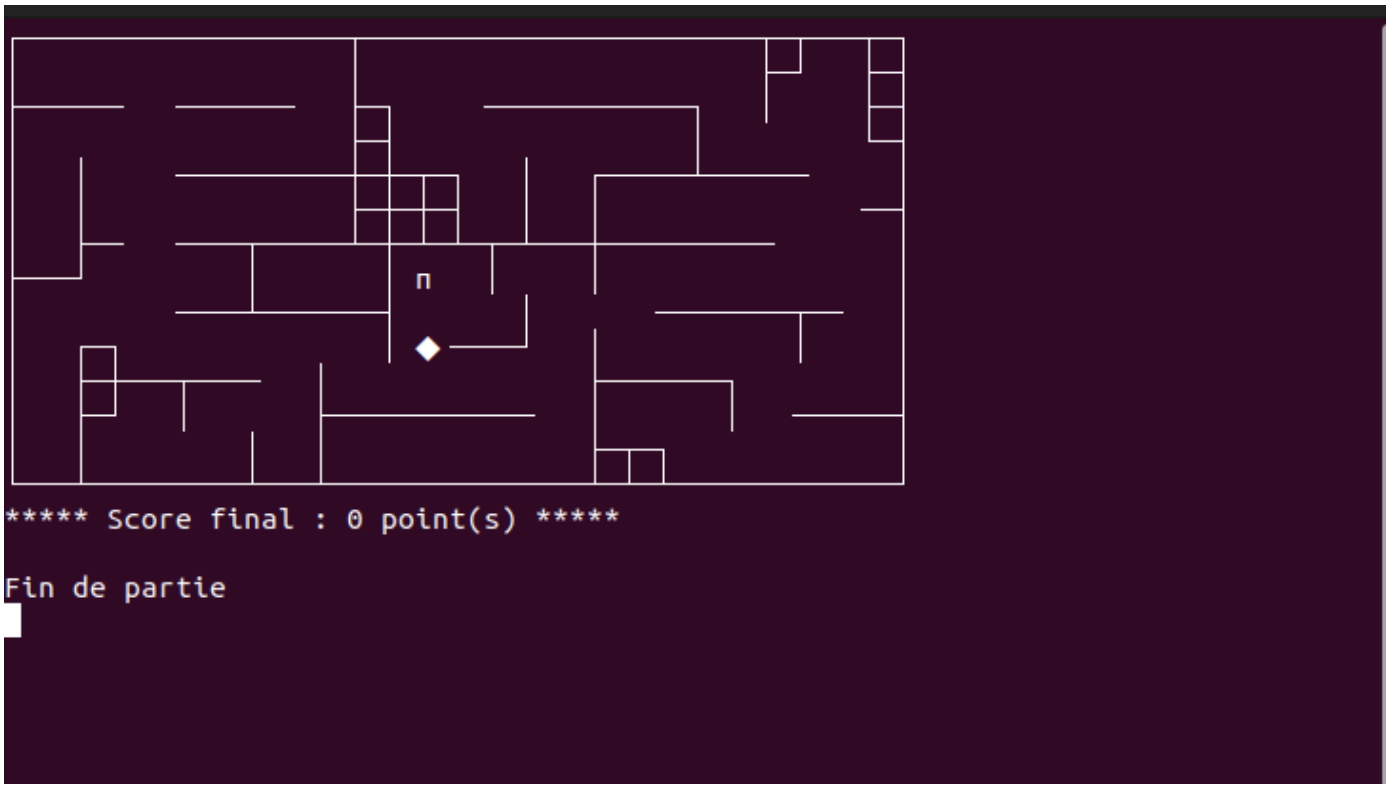
Si vous entrez 2, le jeu « Trouver la sortie » se lance :



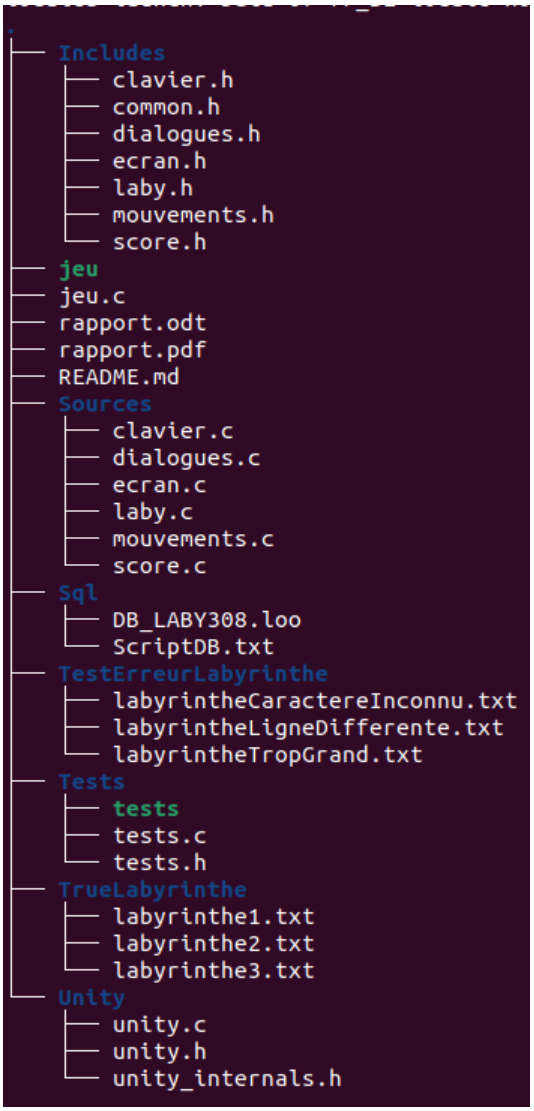
Si vous entrez 3, le jeu « Attraper le monstre » se lance :

Lorsque l'un des deux mini-jeux se termine, le menu de fin s'affiche.

Une fois que la partie est terminée, le score est sauvegardé et l'utilisateur retourne à l'écran d'accueil.



Structure du projet



Liste des différentes sources utilisées

clavier.c :

Dans le fichier clavier.c, nous nous occupons de la gestion des fonctions contenant l'utilisation du clavier.

dialogues.c :

Dans le fichier dialogues.c, nous nous occupons de la gestion des fonctions contenant l'affichage des menus

ecran.c

Dans le fichier ecran.c, nous nous occupons de la gestion des fonctions permettant l'affichage à l'écran.

laby.c

Dans le fichier laby.c, nous nous occupons de la gestion des fonctions permettant la lecture, affichage, conversion du Labyrinthe.

mouvements.c

Dans le fichier mouvements.c, nous nous occupons de la gestion du déplacement du joueur ainsi que celui du monstre.

score.c

Dans le fichier score.c, nous nous occupons de la gestion de la base de données (consultation, ajout, suppression), le timer ainsi que le score.

tests.c

Dans le fichier tests.c, nous nous occupons de la gestion des tests unitaires. C'est dans ce fichier que ceux-ci sont développés et tester.

Jeu.c

Dans le fichier jeu.c, nous nous occupons de la gestion globale de l'application. Le jeu.c correspond à notre main, il nous permet de lancer l'application.

Tableau des fonctions

Nom et signature de la fonction	Nom des paramètres	Type des paramètres	Utilité	Type de retour	Nom du fichier sources
*LireTexte()	Aucun	Aucun	Lit un texte a l'écran qui retourne une chaîne créer en mémoire avec le texte ou un NULL en cas d'erreur	char	Clavier.c
LireToucheEtAfficher()	aucun	aucun	Lit une touche au clavier en bloquant en l'affichant Et renvoie le code de caractère après l'avoir affiché	int	Clavier.c
LireToucheBloquantSansAfficher()	aucun	aucun	Lit une touche au clavier en bloquant et sans rien afficher // Renvoie le code de caractère	int	Clavier.c
SauverJoueurCourant	*nomDeJoueur	char	Sauve le joueur courant	void	Clavier.c
LireJoueurInscrit	aucun	aucun	Lit le joueur inscrit	bool	Clavier.c
*LireJoueurCourant	aucun	aucun	Lit le joueur courant	char	Clavier.c
AfficherOptions	aucun	aucun	Dialogue d'affichage des options du jeu	void	Dialogues.c
ChoisirOption	aucun	aucun	Choix d'une option lue au clavier et appel de la fonction correspondante // La lecture se fait en lisant une touche enfoncée sans l'afficher et sans ENTER. Retourne un true ou un false	bool	Dialogues.c

InitialiserEcran	aucun	aucun	Initialise l'écran pour le mode texte positionnable	Void	Ecran.c
RestaurerEcran	aucun	aucun	Restaure l'écran pour le mode normal	void	Ecran.c
RetourALaLigne	aucun	aucun	Place la position courante au début de la ligne suivante	void	Ecran.c
AfficherTexteIndenteAvecRetour	*texteAAfficher	char	Affiche un texte avec indentation à partir de la position courante puis revient au début de la ligne suivante	void	Ecran.c
AfficherTexteIndenteSansRetour	*texteAAfficher	char	Affiche un texte à partir de la position courante puis revient au début de la ligne suivante	void	Ecran.c
AfficherTexteSansRetour	*texteAAfficher	char	Affiche un texte à partir de la position courante sans retour au début de la ligne suivante	void	Ecran.c
AfficherCharSansRetour	caractereAAfficher	Unsigned char	Affiche un caractère à partir de la position courante sans retour au début de la ligne suivante	void	Ecran.c
AfficherCharSpecialSansRetour	caractereAAfficher	Unsigned char	Affiche un caractère spécial à partir de la position courante sans retour au début de la ligne suivante	void	Ecran.c
AfficherTexteDansCadre	*texteAAfficher	char	Affiche un texte dans un cadre	void	Ecran.c
AfficherTempsRestant	tempsRestant	long	Affiche le temps restant à la position courante en revenant ensuite au début de la ligne	void	Ecran.c

EffacerEcran	aucun	aucun	Efface l'écran	void	Ecran.c
AttendreConfirmation	*messageAAfficher	char	Affiche un message, puis demande d'enfoncer une touche et attend que ce soit fait	void	Ecran.c
AttendreSecondes	dureeEnSecondes	int	Attends une durée donnée (en seconde)	void	Ecran.c
AfficherScore	score	int	Affiche un score à la position courante puis revient au début de la ligne suivante	void	Ecran.c

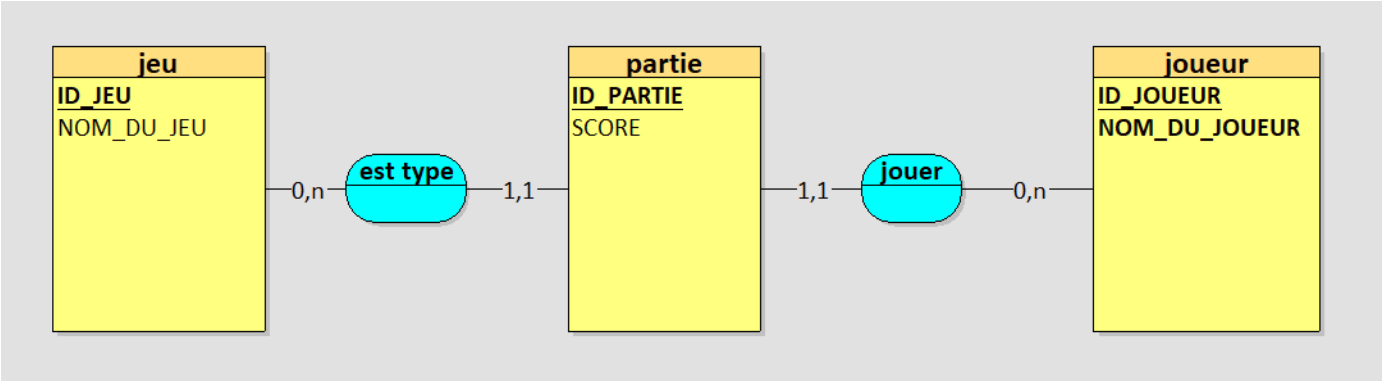
PositionnerCurseur	positionX positionH	Int int	Positionne le curseur à une position donnée de la console sans effacer l'écran	void	Ecran.c
loadingAnimation	aucun	aucun	Lance une animation de chargement	void	Ecran.c
afficherToucheDuJeu	aucun	aucun	Affiche les touches du jeu	void	Ecran.c
ConvertirEnTexteAffichable	cellule *celluleAAfficher murEnHaut murADroite murEnBas murAGauche	Char struct Laby_Cell bool bool bool bool	Convertis une cellule 'mur' du labyrinthe en deux caractères à afficher en fonction de la présence ou non de murs en haut, en bas, à gauche et à droite	void	Laby.c
ConvertirUneCelluleEnTexteAffichable	*labyrinthe positionV positionH	struct Laby_Complet int Int	Convertir une cellule du labyrinthe en texte affichable sous la forme de deux caractères contigus (gauches et droits)	void	Laby.c
*ChargerLabyrintheAuHasard	aucun	aucun	Charge le labyrinthe à partir d'un fichier et le convertir en une structure Laby_Complet contenant le labyrinthe sous tous les formats et ses paramètres	struct Laby_Complet	Laby.c
CelluleAuHasard	*labyrinthe *resultat auBord	struct Laby_Complet struct Laby_Position bool	Tire une cellule au hasard et renvoie ses coordonnées	void	Laby.c
PlacerAuHasard	*labyrinthe elementAPlacer elementARemplacer auBord	struct Laby_Complet char char bool	Place au hasard un élément dans le labyrinthe	void	Laby.c
AfficherLabyrinthe	*labyrinthe	struct Laby_Complet	Affiche tout le labyrinthe: murs, joueur, monstre, cookies...	void	Laby.c
RafraichirAffichageLabyrinthe	*labyrinthe	struct Laby_Complet	Rafraichis le labyrinthe: murs, joueur, monstre, cookies...	void	Laby.c
LireFichierLabyrinthe	*nomFichierLabyrinthe *labyrinthe *messageAFormater	Char struct Laby_Complet struct Laby_Message	Lit un fichier texte représentant un labyrinthe en vérifiant qu'il est correct	bool	Laby.c

LibererLabyrinthe	*labyrinthe	struct Laby_Complet	Libère la mémoire d'un labyrinthe	void	Laby.c
ConvertirLabyrinthe	*labyrinthe *messageAFormater	struct Laby_Complet struct Laby_Message	Convertit tout le labyrinthe en texte affichable à l'écran chaque cellule est convertie en deux caractères affichables	bool	Laby.c
*OuvrirFichier	*nomFichierLabyrinthe *messageAFormater	Char struct Laby_Message	Ouvre un fichier et renvoie une erreur si pas OK	FILE	Laby.c
FermerFichier	*fichierLabyrinthe	FILE	Ferme un fichier	void	Laby.c
AtteindreCible	*labyrinthe source positionV positionH cibleV cibleH afficher	struct Laby_Complet char int int int int bool	Vérifie que la destination atteignable et y déplacent le joueur	char	Mouvements.c
AvancerHaut	*labyrinthe *position source afficher	struct Laby_Complet struct Laby_Position char bool	Avance le joueur ou le monstre vers le haut, si possible	char	Mouvements.c
AvancerBas	*labyrinthe *position source afficher	struct Laby_Complet struct Laby_Position char bool	Avance le joueur ou le monstre vers le bas, si possible	char	Mouvements.c
AvancerGauche	*labyrinthe *position source afficher	struct Laby_Complet struct Laby_Position char bool	Avance le joueur ou le monstre vers la gauche, si possible	char	Mouvements.c

AvancerDroite	*labyrinthe *position source afficher	struct Laby_Complet struct Laby_Position char bool	Avance le joueur ou le monstre vers la droite, si possible	char	Mouvements.c
SortieTrouvee	*labyrinthe	struct Laby_Complet	Vérifie si une sortie a été trouvée, c'est-à-dire que le joueur est sur un bord	bool	Mouvements.c
SeDeplacer	*labyrinthe	struct Laby_Complet	déplace un joueur dans le labyrinthe et le redessine	char	Mouvements.c
DeplacerMonstre	*labyrinthe	struct Laby_Complet	Déplace le monstre	void	Mouvements.c
JouerUnePartie	*labyrinthe AttraperMonstre	struct Laby_Complet bool	Se déplacer jusqu'à trouver une sortie, attraper un monstre ou abandonner	long	Mouvements.c
JouerPartieSortie	aucun	aucun	Jeu qui consiste à trouver la sortie avec un temps limite	void	Mouvements.c
JouerPartieMonstre	aucun	aucun	Jeu qui consiste à attraper un monstre avec un temps limite	void	Mouvements.c
EffacerLaDBEnMemoire	*tousLesScores	struct Laby_Score_Complet	Efface de la mémoire la structure qui a stocke tout le contenu de la DB	void	Score.c
ExecuterInstructionSQL	DBProduction *sqlConnection *instructionSQL	Bool MYSQL char	Exécute une instruction SQL	void	Score.c
CreerLaDB	DBProduction *sqlConnection	Bool MYSQL	Crée la base de données	void	Score.c
*ConnecterLaDB	DBProduction creerLaDBSiNecessaire terminerSiErreur	Bool Bool bool	Connexion a la base de données, en la créant si nécessaire	MYSQL	Score.c
FermerLaDB	*sqlConnection	MYSQL	Ferme la connexion vers la DB	void	Score.c
LireJoueursDansDB	*tousLesScores *sqlConnection	struct Laby_Score_Complet MYSQL	Extrait tous les joueurs de la DB	void	Score.c
LireJeuxDansDB	*tousLesScores	struct	Extrait tous les jeux de la DB	void	Score.c

	*sqlConnection	Laby_Score_Complet MYSQL			
LirePartiesDansDB	*tousLesScores *sqlConnection	struct Laby_Score_Complet MYSQL	Extrait toutes les parties jouées de la DB	void	Score.c
ChargerLaDBEnMemoire	DBProduction *tousLesScores	Bool struct Laby_Score_Complet	Charger toutes les jeux, joueurs et scores en mémoire, dans l'ordre décroissant des points	void	Score.c
AjouterJoueurDansDB	DBProduction joueur	Bool struct Laby_Score_Joueur	Ajouter un joueur dans la DB	bool	Score.c
AjouterJeuDansDB	DBProduction jeu	Bool struct Laby_Score_Jeu	Ajouter un jeu dans la DB	bool	Score.c
AjouterPartieDansDB	DBProduction partie	Bool struct Laby_Score_Partie	Ajouter une partie dans la DB	bool	Score.c
SInscrire	aucun	aucun	Demander au joueur de s'inscrire en donnant son pseudo	void	Score.c
AjouterScore	*typeDePartie score	Char long	Ajoute un score à la DB	void	Score.c
AfficherMeilleursScores	nombreDeScoresParJeu	int	Affiche les meilleurs scores de chaque jeu	void	Score.c
trierParOrdreDecroissant	*tousLesScores	struct Laby_Score_Complet	Fonction qui permet de trier les scores par ordre décroissant	void	Score.c

Schéma de la base de données



Pour la base de données, nous avons créé un modèle E/A. Dans ce modèle, nous avons 3 tables « jeu », « partie », « joueur ». Chaque table possède une clé primaire « ID_JEU », « ID_PARTIE », « ID_JOUEUR ».

La table « jeu » a pour but de stocker le type de jeu. La table « partie » a pour but de stocker les scores des parties, la table « joueur » permet de stocker les pseudo des joueurs.

Ces relations permettent d'avoir une clé étrangère « jeu », ainsi qu'une autre clé étrangère « joueur » dans « partie ».

Répartition des tâches

Nom de la fonction	Auteur
*LireTexte()	Louis
LireToucheEtAfficher()	Louis
LireToucheBloquantSansAfficher()	Louis
SauverJoueurCourant	Louis
LireJoueurInscrit	Noah
*LireJoueurCourant	Noah
AfficherOptions	Noah
ChoisirOption	Noah
InitialiserEcran	/
RestaurerEcran	/
RetourALaLigne	Alexis
AfficherTexteIndenteAvecRetour	Alexis
AfficherTexteIndenteSansRetour	Alexis
AfficherTexteSansRetour	Alexis
AfficherCharSansRetour	Alexis
AfficherCharSpecialSansRetour	/
AfficherTexteDansCadre	Noah
AfficherTempsRestant	Noah
EffacerEcran	/
AttendreConfirmation	Louis
AttendreSecondes	/
AfficherScore	Noah
PositionnerCurseur	/
loadingAnimation	/

afficherToucheDuJeu	Noah
ConvertirEnTexteAffichable	Noah
ConvertirUneCelluleEnTexteAffichable	Noah
*ChargerLabyrintheAuHasard	Noah
CelluleAuHasard	Noah
PlacerAuHasard	Noah
AfficherLabyrinthe	Louis
RafraichirAffichageLabyrinthe	Louis
LireFichierLabyrinthe	Noah
LibererLabyrinthe	Noah
ConvertirLabyrinthe	Alexis
*OuvrirFichier	Noah
FermerFichier	Noah
AtteindreCible	Noah
AvancerHaut	Noah
AvancerBas	Noah
AvancerGauche	Noah
AvancerDroite	Noah
SortieTrouvee	Alexis
SeDeplacer	Alexis
DeplacerMonstre	Alexis
JouerUnePartie	Noah
JouerPartieSortie	Noah
JouerPartieMonstre	Noah
EffacerLaDBEnMemoire	Noah
ExecuterInstructionSQL	Alexis
CreerLaDB	Alexis
*ConnecterLaDB	Louis
FermerLaDB	Alexis
LireJoueursDansDB	Alexis
LireJeuxDansDB	Alexis

LirePartiesDansDB	Alexis
ChargerLaDBEnMemoire	Noah
AjouterJoueurDansDB	Noah
AjouterJeuDansDB	Noah
AjouterPartieDansDB	Noah
Sinscrire	Noah
AjouterScore	Noah
AfficherMeilleursScores	Noah
trierParOrdreDecroissant	Noah

Jeu de tests fonctionnels

Id	Use Cases	Nom test	Description	Résultat attendu	Résultat obtenu
1	inscription	Effectuer votre inscription	- Avoir lancé le jeu - Appuyer sur votre touche « 1 » - Entrez votre pseudo comprenant moins de 20 caractères. - Appuyer sur votre touche « enter »	Retourne à l'accueil en affichant bonjour « pseudo »	
2	inscription	Effectuer votre inscription (sans caractères)	- Avoir lancé le jeu - Appuyer sur votre touche « 1 » - Entrez votre pseudo comprenant 0 caractère. - Appuyer sur votre touche « enter »	Vous resterez sur la page d'inscription qui vous demande d'entrée votre pseudo	
3	inscription	Effectuer votre inscription(avec + de 100 caractères)	- Avoir lancé le jeu - Appuyer sur votre touche « 1 » - Entrez votre pseudo comprenant + de 100 caractères. - Appuyer sur votre touche « enter »	Vous resterez sur la page d'inscription qui vous demande d'entrée votre pseudo	
4	Jouer partie sortie	Jouer au labyrinthe pour	- Avoir lancé le jeu - s'être inscrit	Le labyrinthe avec une sortie	

		trouver la sortie	- Appuyer sur votre touche « 2 »	s'affiche avec un timer	
5	Jouer partie monstre	Jouer au labyrinthe pour attraper le monstre	- Avoir lancé - s'être inscrit - Appuyer sur votre touche « 3 »	Le labyrinthe avec un monstre s'affiche avec un timer	
6	Déplacer joueur	Déplacer joueur vers la gauche	- Avoir lancé le jeu - s'être inscrit - Avoir lancé jouer partie sortie ou jouer partie monstre - Appuyer sur votre touche « 4 »	Votre personnage se déplacera vers la gauche(sauf si mur)	
7	Déplacer joueur	Déplacer joueur vers la droite	- Avoir lancé le jeu - s'être inscrit - Avoir lancé jouer partie sortie ou jouer partie monstre - Appuyer sur votre touche « 6 »	Votre personnage se déplacera vers la droite(sauf si mur)	
8	Déplacer joueur	Déplacer joueur vers le haut	- Avoir lancé le jeu - s'être inscrit - Avoir lancé jouer partie sortie ou jouer partie monstre - Appuyer sur votre touche « 8 »	Votre personnage se déplacera vers le haut(sauf si mur)	
9	Déplacer joueur	Déplacer joueur vers le bas	- Avoir lancé le jeu - s'être inscrit - Avoir lancé jouer partie sortie ou jouer partie monstre - Appuyer sur votre touche « 2 »	Votre personnage se déplacera vers le bas(sauf si mur)	
10	Abandonner jeu	Abandonner le jeu en pleine partie	- Avoir lancé le jeu - s'être inscrit - Avoir lancé jouer partie sortie ou jouer partie monstre - Appuyer sur votre touche « q »	Votre partie s'arrêtera et vous reviendriez à l'accueil	
11	scores	Affiches les scores	- Avoir lancé le jeu - s'être inscrit - rappuyer sur votre touche « 1 »	Vous verrez les scores que vous avez faits pour chaque partie	
12	Aide	Affiches les commandes	- Avoir lancé le jeu - Appuyer sur votre touche « 8 »	Vous verrez les différentes touches qui vous permettront de déplacer votre	

				joueur ou d'abandonner	
13	Arrêter	Arrêter le jeu	- Avoir lancé le jeu - Appuyer sur votre touche « 0 »	Un « Au revoir ! » s'affichera à l'écran	

Éventuelles améliorations

Pour ce projet, nous avons quelques points à améliorer. Nous aurions pu avoir une meilleure utilisation de Git. En effet, nous avons eu quelques fois des conflits de fusion des fichiers. Nous avons aussi rencontré quelques problèmes de fuite de mémoire.

Liste des matières utilisée

Pour ce projet, nous avons dû utiliser la matière suivante de nos 4 AA présente dans l'UE.

1. Méthodes et principes de programmation 2 :
   - ➢ Développer des algorithmes plus complexes
   - ➢ Analyser et concevoir un modèle E/A optimiser en fonction des exigences demandées
2. Langage de programmation procédural 2 :
   - ➢ Gestion de la mémoire d'un programme
   - ➢ Utilisation de structure
   - ➢ Utilisation de Ncurses pour l'affichage sous Linux
3. Projet 1 :
   - ➢ Découpe et organisation du projet
   - ➢ Utilisation de Git
   - ➢ Utilisation d'un framework de test unitaire (Unity)
   - ➢ Organisation de code
4. Fichiers et bases de données 2 :
   - ➢ Ouverture, fermeture, lecture et écriture d'un fichier
   - ➢ Connexion, déconnexion, lecture et écriture dans une base de données à l'aide de la librairie « mysql »

Conclusion

En conclusion, pour finaliser ce projet, nous avons pu relire notre code de juin, et améliorer ce qui n'allait pas (Génération du labyrinthe, affichage du labyrinthe, problème d'allocation mémoire, faute d'orthographe, manque de cohérence).