ECE HEROES

Projet Match-3 en C

RAPPORT FINAL

ING1 - Groupe 6
Décembre 2025

Membres de l'équipe :
Adam Fatah
Wali Faddel
Emmanuel Brami Poirier d'Ange d'Orsay
Yanis AIT EL HAJ

SOMMAIRE

1. Planning réel et répartition des taches

1.1 Planning prévisionnel vs réel

Le planning prévu dans le rapport mi-parcours était théorique. Voici la comparaison avec ce qui s'est réellement passe :

Semaine	Prevu	Realise
Sem 1-2	Analyse & Affichage	Analyse du CDC, conception
Sem 3	Détection simple	Préparation rapport mi-parcours
Sem 4	Gravite & Boucle	Début codage structures
Sem 5	Extensions	Tests premiers modules
Sem 6	Tests & Rendu	Développement intensif complet + Tests + Intégration

1.2 Répartition réelle des taches

Étant donne le développement concentre sur la dernière semaine, nous avons travaille de manière collaborative sur l'ensemble des fichiers, principalement en sessions de travail communes sur Discord.

Membre	Contributions principales
Adam	Figures spéciales (croix 9, carre 4x4, 6 alignes couleur, bombe 3x3), implémentation des 3 niveaux, écran règles du jeu, corrections finales jeu.h
Wali	Logique match-3, module affichage (menu, grille, HUD), gestion clavier, sauvegarde, affichage emojis fruits, intégration affichage_console.h, debug temps/vies
Emmanuel	Boucle principale main.c, ajout de commentaires dans le code, tests fonctionnels, validation gameplay
Yanis	Tests et validation, rapport mi-parcours, documentation, ajout de commentaires

Note : Le travail collaboratif a implique que chaque membre a contribue a la plupart des fichiers. La repartition ci-dessus indique les domaines ou chacun a apporte le plus de contributions.

2. Modifications par rapport au rapport mi-parcours

Le rapport mi-parcours presentait une conception theorique. Plusieurs modifications importantes ont ete apportees lors de l'implementation :

2.1 Structure de donnees

Prevu : Une structure simple 'Jeu' avec grille 25x45.
Realise : Structure 'Niveau' enrichie avec structures imbriquees 'Case' et 'Contrat'. Grille reduite a 10x15 pour une meilleure jouabilite en console.

2.2 Architecture modulaire

Prevu : 3 modules (main.c, jeu.c, affichage.c)
Realise : 6 modules pour une meilleure separation :

- structure.h - Definitions des types et constantes
- jeu.c/h - Logique metier du match-3
- affichage.c/h - Interface utilisateur
- affichage_console.c/h - Bibliotheque console (fournie)
- sauvegarde.c/h - Persistance des donnees
- main.c - Orchestration generale

2.3 Fonctionnalités ajoutées

- Système de 3 vies conservées entre niveaux
- Score cumulatif sur tous les niveaux
- Écran de règles du jeu
- Affichage avec émojis (fruits)
- Reset automatique des objectifs après combos initiaux

2.4 Détail des 3 niveaux (Contrats)

Les contrats ont été définis avec une difficulté croissante :

Niveau	Objectifs	Coups	Temps
1	10 Item 1 (pastèque)	30	2 min
2	15 Item 1 + 15 Item 2 (myrtille)	25	1 min 30
3	20 Item 1 + 20 Item 2 + 10 Item 3 (kiwi)	20	1 min

2.5 Intégration bibliothèque du cours

En fin de projet, nous avons intègré la bibliothèque affichage_console.h fournie en cours, remplaçant nos fonctions custom. Changements majeurs :

- gotoligcol(y,x) remplace par gotoxy(x,y) - attention a l'inversion des parametres
- color() remplacé par set_color()
- system("cls") remplace par clrscr()
- Constantes couleurs (BLANC, NOIR) remplacées par enum COLORS (WHITE, BLACK)

3. Contenu des fichiers header

3.1 structure.h

```
#ifndef STRUCTURES_H
#define STRUCTURES_H

#define LIGNES 10
#define COLONNES 15
#define NB_TYPES 5
#define MAX_PSEUDO 50

typedef enum { VIDE=0, ITEM_1, ITEM_2, ITEM_3, ITEM_4, ITEM_5, MUR, BONUS,
MALUS } TypeItem;

typedef struct { TypeItem type; int estSelectionne; } Case;
typedef struct { int quantiteCible[NB_TYPES+1]; int
quantiteActuelle[NB_TYPES+1]; } Contrat;
typedef struct { Case grille[LIGNES][COLONNES]; char pseudo[MAX_PSEUDO];
    int vies, score, coupsRestants, numeroNiveau, tempsTotalSec;
    time_t startTime; Contrat contrat; } Niveau;
#endif
```

3.2 jeu.h

```
#ifndef JEU_H
#define JEU_H
#include "structure.h"

void initialiserNiveau(Niveau *niveau, int numNiveau);
void echangerCases(Case *case1, Case *case2);
void effetBombe(Niveau *niveau, int x, int y);
void effetLigne(Niveau *niveau, int ligne);
void effetColonne(Niveau *niveau, int colonne);
void effetCouleur(Niveau *niveau, int type);
void effetCarre4x4(Niveau *niveau, int x, int y);
int detecterEtSupprimerAlignements(Niveau *niveau);
void faireTomberEtRemplir(Niveau *niveau);
void gererCombos(Niveau *niveau);
int verifierContrat(Niveau *niveau);
#endif
```

3.3 affichage.h

```
#ifndef AFFICHAGE_H
#define AFFICHAGE_H
#include "structure.h"

void afficherCadre(void);
void afficherGrille(Niveau *niveau, int curseurX, int curseurY);
void afficherHUD(Niveau *niveau);
int afficherMenuPrincipal(void);
#endif
```

3.4 sauvegarde.h

```
#ifndef SAUVEGARDE_H
#define SAUVEGARDE_H
#include "structure.h"
```

```
int sauvegarderPartie(Niveau *niveau);
int chargerPartie(Niveau *niveau);
#endif
```

4. Graphe d'appels

Voici le graphe d'appels simplifie montrant les relations entre les fonctions principales :

5. Tests réalisés

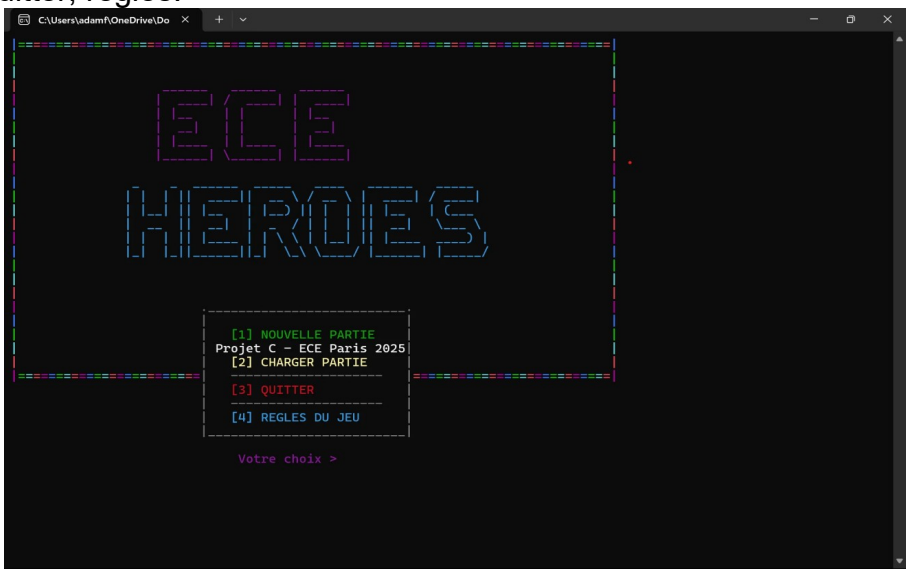
5.1 Tests fonctionnels

Test	Description	Resultat
Menu principal	Navigation entre les 4 options	OK
Déplacement curseur	Flèches directionnelles	OK
Selection/Echange	Espace pour sélectionner, échange adjacent	OK
Détection 3+ alignes	Horizontal et vertical	OK
Croix de 9	Supprime ligne + colonne	OK
Carre 4x4	Supprime 16 cases	OK
6 alignes (couleur)	Supprime tous les items du type	OK
Bombe 3x3 (ext.)	5 alignes niveau 2+, zone 3x3	OK
Gravite	Chute + remplissage aléatoire	OK
Combos automatiques	Enchaînement jusqu'à stabilisation	OK
Sauvegarde/Chargement	Persistance complète état	OK
Progression 3 niveaux	Difficulté croissante	OK
Gestion vies	3 vies, conservation entre niveaux	OK

5.2 Captures d'écran

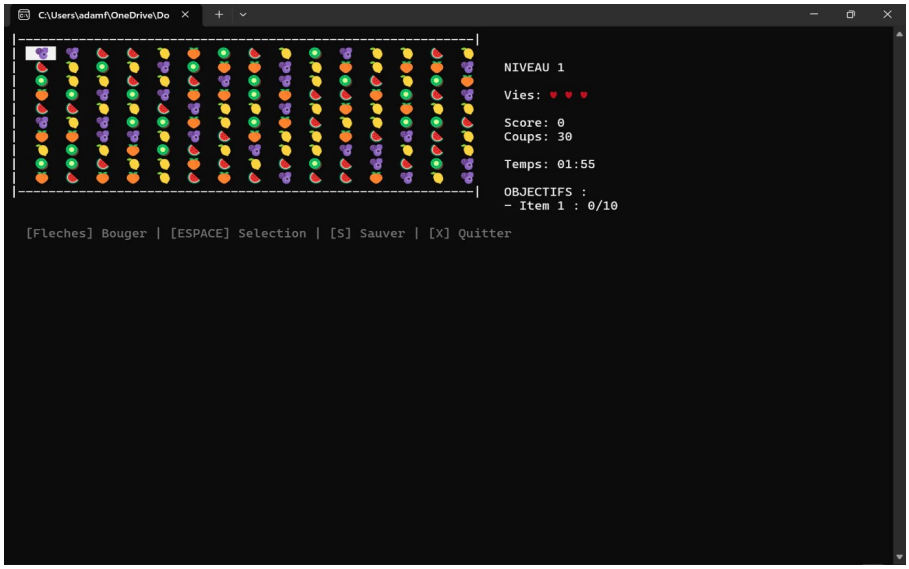
Menu principal

Le menu affiche le titre en ASCII art colore avec les 4 options : nouvelle partie, charger, quitter, regles.



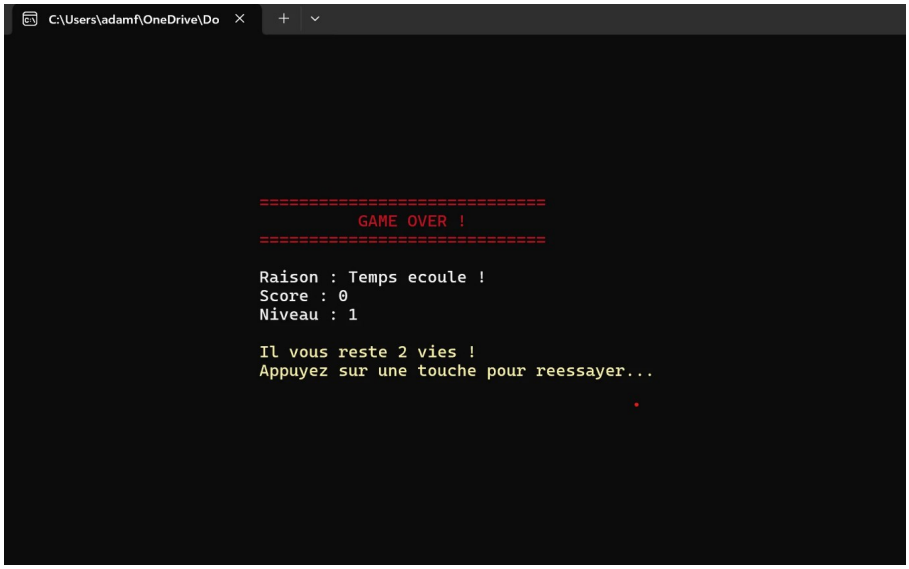
Ecran de jeu (Niveau 1)

La grille affiche les fruits en emojis. Le HUD a droite montre le niveau, les vies, le score, les coups, le temps et les objectifs.



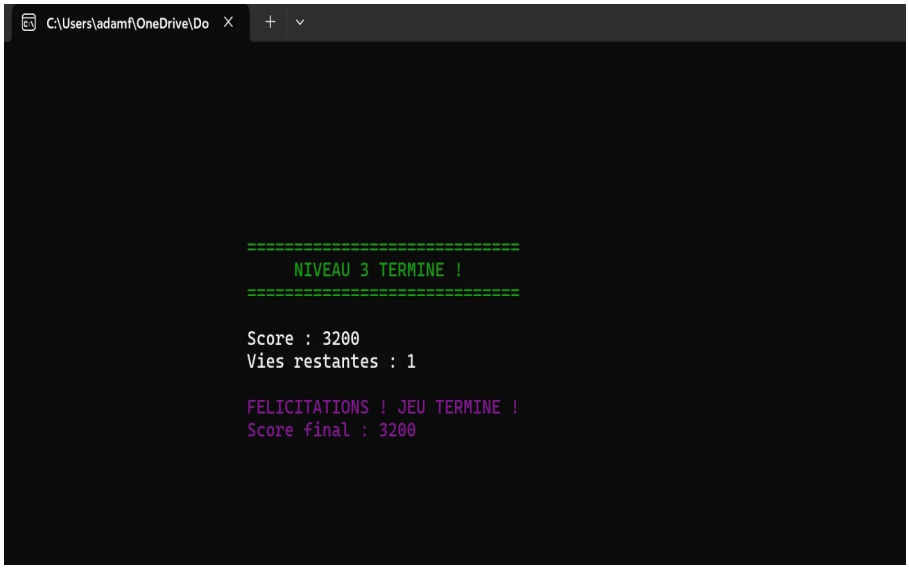
Écran de défaite

Quand le temps est ecoule ou les coups épuises, l'ecran Game Over s'affiche avec la raison et les vies restantes.



Écran de victoire finale

Après avoir compléter les 3 niveaux, l’écran de félicitations affiche le score final.



5.3 Bugs corriges

1. **Inversion x/y dans gotoxy() :** Corrige lors de l'intégration de affichage_console.h
2. **Score/Objectifs reset :** Les combos initiaux ne comptaient pas - corrige avec reset apres gererCombos() initial
3. **Prototypes incohérents :** Noms de fonctions dans jeu.h corriges (itemBombe -> effetBombe)

6. Bilans

6.1 Bilan collectif

Ce projet nous a permis de mettre en pratique les notions de programmation en C apprises en cours. Malgré un démarrage tardif du développement, nous avons réussi a livrer un jeu fonctionnel grace a un travail intensif durant la dernière semaine.

Points positifs :

- Bonne coordination en situation de rush
- Communication efficace via Discord
- Respect du délai final
- Jeu fonctionnel et jouable

Difficultés rencontrées :

- Démarrage tardif du codage
- Gestion du stress en fin de projet
- Intégration tardive de affichage_console.h
- Debugging intensif sur une courte periode

6.2 Bilans individuels

Adam

En tant que chef de projet, j'ai appris a coordonner une équipe sous pression. Le rush de dernière semaine a été intense mais formateur. J'ai principalement travaille sur le module d'affichage et l'intégration finale. Cette expérience m'a appris l'importance de commencer plus tôt a l'avenir.

Wali

Le développement des algorithmes de détection a été un défi passionnant malgré le temps limite. J'ai beaucoup appris sur les parcours de tableaux 2D et les figures spéciales. Travailler sous pression m'a permis de développer ma capacité a coder efficacement.

Emmanuel

J'ai travaille sur le fichier main.c et l'ajout de commentaires dans le code pour améliorer la lisibilité. Cette expérience m'a permis de mieux comprendre l'architecture globale d'un projet en C.

Yanis

J'ai principalement contribue aux phases de tests et de validation du jeu. Le rapport mi-parcours m'a permis de comprendre l'importance de la documentation dans un projet. J'aurais aime m'impliquer davantage dans le code.

7. Sources et aide utilisée

7.1 Documentation

- Cours d'Algorithmique et Programmation structurée (ECE Paris)
- Documentation C standard (cppreference.com)
- Module affichage_console.h fourni par l'équipe enseignante

7.2 Aide humaine

- Questions posées aux enseignants en TD
- Entraide entre membres de l'équipe

7.3 Aide artificielle (IA)

Nous avons utilise Claude (Anthropic) comme assistant pour :

- Compréhension de concepts algorithmiques
- Debugging de certaines fonctions
- Suggestions d'améliorations du code

Important : L'IA a servi d'outil d'apprentissage et d'assistance, mais tout le code a été compris, adapte et teste par les membres de l'équipe. Nous assumons l'entière responsabilité du travail rendu.

7.4 Outils utilises

- IDE : Visual Studio Code
- Compilateur : GCC (MinGW)
- Gestion de versions : Git / GitHub
- Communication : Discord

--- Fin du rapport ---