

Projet de jeu : Queens

Phase 1 – Structure de base et initialisation

Tâche 1.1 – Structures de données

- Définir une structure *Grille* avec taille variable état des cases (vide, reine, marquage spécial)
- Définir une structure *Joueur* avec nom et meilleur score
- Créer les énumérations pour les états des cases (VIDE, REINE, MARQUE)

Tâche 1.2 – Gestion des fichiers

- Fonction de lecture du fichier des scores (*read_scores()*)
- Fonction de sauvegarde des scores (*save_scores()*)
- Fonction de mise à jour du meilleur score d'un joueur

Tâche 1.3 – Gestion du temps (facultatif)

- Implémenter les fonctions de chronométrage (début/fin de partie)
- Fonction de calcul du temps écoulé en secondes

Phase 2 – Interface utilisateur dans la console

Tâche 2.1 – Menu principal

- Initialiser ncurses et configurer l'écran
- Créer le menu pour le choix : 'Nouvelle partie' ou 'Quitter'
- Gestion de la saisie du nom du joueur
- Gestion de la saisie de la taille de grille (5-10)

Tâche 2.2 – Affichage de la grille

- Fonction d’affichage de la grille avec ses coordonnées
- Affichage des reines ('Q'), des marquages spéciaux ('X'), cases vides ('.')
- Affichage du temps écoulé en temps réel
- Affichage des messages d’état/erreur

Tâche 2.3 – Gestion des entrées

- Fonction de saisie des coordonnées (ligne, colonne)
- Menu contextuel : « Placer reine », « Enlever reine », « Marquer case », « Enlever marquage »
- Validation des coordonnées saisies

Phase 3 – Logique de validation

Tâche 3.1 – Validation des règles de base

- Fonction *check_ligne()* : une seule reine par ligne
- Fonction *check_col()* : une seule reine par colonne
- Fonction *check_diag()* : une seule reine par diagonale

Tâche 3.2 – Validation adjacence

- Fonction *check_adjacence()* : pas de reines sur les cases adjacentes (8 directions)

Tâche 3.3 – validation globale

- Fonction *valid_place()* : combine toutes les vérifications en mm temps
- Fonction *detect_place_error()* : retourne les règles violées
- Fonction d’affichage des erreurs spécifiques

Phase 4 – Logique de jeu

Tâche 4.1 – Actions du joueur

- Fonction *place_queen(x, y)* avec validation
- Fonction *erase_queen(x, y)*
- Fonction *mark_box(x, y)* pour les cases impossibles
- Fonction *erase_mark(x, y)*

Tâche 4.2 – Détection de victoire

- Fonction *count_queens()*
- Fonction *game_won()* : N reines placées correctement (N=taille grille)
- Gestion de la fin de partie et sauvegarde du score

Tâche 4.3 – Validation en temps réel

- Vérification après chaque action du joueur
- Affichage des violations en cours de partie
- Messages d'aide contextuelle

Phase 5 – Boucle de jeu principal

Tâche 5.1 – Etat de jeu

- Enumération des états (MENU, EN_COURS, GAGNE, QUITTER)
- Machine à états pour gérer les transitions

Tâche 5.2 – Boucle principale

- Gestion des évènements clavier
- Mise à jour de l'affichage
- Gestion du temps en continu

Phase 6 : Finalisation et tests

Tâche 6.1 - Gestion des erreurs

- Validation de toutes les entrées utilisateur
- Gestion des cas limites (grille pleine, fichier scores corrompu)
- Messages d'erreurs clairs

Tâche 6.2 – Optimisation interface

- Rafraîchissement intelligent de l'écran
- Gestion propre de la fermeture ncurses
- Interface responsive selon taille du terminal

Tâche 6.3 – Tests

- Tests unitaires des fonctions de validation
- Tests d'intégration du flux de jeu complet
- Tests avec différentes tailles de grille

Schéma de flux du programme

