**EL-HAMDEOUI Marouan, PEYRONNET Léo,**

**PHILIBERT Matéo, VALDANT Tom**

# Rapport du mini-projet Candy Crush

## I. Quel niveau de difficulté avez-vous implémenté ?

Dans notre projet Candy Crush, nous avons implémenté le niveau de difficulté 3, c'est-à-dire la capacité à détecter des combinaisons non seulement entre des bonbons adjacents directs, mais également entre les voisins des voisins. Cela permet de gérer des suppressions de bonbons plus complexes et d'offrir une meilleure dynamique de jeu que les niveaux basiques.

## II. Décrivez les règles de votre jeu en quelques lignes

### *Règles*

Notre jeu **Candy Crush** est jouable entièrement dans le **terminal**. Le but est d'aligner des bonbons de la même couleur pour les faire disparaître et marquer des points. Voici les règles principales :

- Le plateau de jeu est une **grille de bonbons** générée aléatoirement.
- Le joueur sélectionne deux cases à échanger.
- Si l'échange forme une **ligne ou colonne d'au moins 3 bonbons identiques**, ces derniers sont supprimés.
- Les bonbons au-dessus tombent pour remplir les vides, et de nouveaux bonbons apparaissent en haut.
- Le jeu permet des combinaisons avancées : il peut détecter et supprimer **les combinaisons indirectes** (ex. en chaîne, ou via des voisins éloignés).
- Le jeu continue jusqu'à ce que le joueur décide d'arrêter.

### *Comment jouer*

Lancer le fichier `main.py` dans le terminal. Le jeu affichera le plateau de jeu et vous demandera de choisir deux bonbons à échanger en indiquant leurs coordonnées respectives. Vous pouvez continuer à jouer jusqu'à ce que vous décidiez de quitter (Ctrl+C).

Un fichier LISEZ-MOI.md est inclut dans les fichiers avec de plus amples informations.

**EL-HAMDEOUI Marouan, PEYRONNET Léo,**

**PHILIBERT Matéo, VALDANT Tom**

# III. Écrivez l'algorithme principal de votre jeu en français

Initialiser la grille :

Appeler la fonction init_grille(x, y) pour créer une grille de bonbons sans combinaison immédiate.

Afficher la grille :

Utiliser la fonction affichage_grille(grille) pour montrer l'état actuel du jeu à l'utilisateur.

Boucle principale du jeu :

Tant que l'utilisateur ne décide pas d'arrêter et que des combinaisons sont possibles :

a. Demander un échange :

Appeler demander_utilisateur_bonbons(grille) pour obtenir deux positions de bonbons à échanger.

Vérifier que ces bonbons sont bien adjacents (déjà fait dans la fonction).

b. Échanger les bonbons :

Appeler echanger_bonbon(grille, b1, b2).

c. Détecter les combinaisons :

Utiliser detecte_coordonnees_combinaison pour voir s'il y a une combinaison après l'échange.

Si aucune combinaison n'est formée, annuler l'échange.

Sinon :

Étendre la combinaison (avec etendre_combinaison).

Supprimer les bonbons de la combinaison (mettre les cases à -1).

Faire descendre les bonbons (descendre_bonbons) et insérer de nouveaux (inserer_bonbons).

Répéter tant qu'il existe encore des combinaisons (trouver_combinaisons_grille puis suppression).

d. Afficher la grille mise à jour.