Exercice récapitulatif du cours

Description

Cet exercice va élaborer le jeu Candy Crush dans une version très simplifiée et qui se focalisera sur l'algorithmique. La partie visuelle ne sera pas fort élaborée.

Comme indiqué, on va travailler sur une version simplifiée mais qui devra pouvoir être complexifiée sans problèmes. Les hypothèses simplificatrices suivantes seront d'application :

- Bien que le vrai jeu comporte plusieurs milliers de niveaux, on se limitera à trois niveaux.
- L'objectif de chaque grille consistera à éliminer la gélatine. La gélatine sera affichée à l'écran sous la forme d'une seconde grille. Une case comportant de la gélatine sera affichée par la lettre « G ».
- Pour chaque niveau, la gélatine sera positionnée de manière aléatoire.
- Chaque niveau se jouera sur une grille de 20 x 20.
- Les pions seront représentés par la première lettre de leur couleur. Nous aurons donc les pions suivants :
    - J : pion Jaune
    - V : pion Vert
    - B : pion Bleu
    - R : pion Rouge
    - M : pion Mauve
- Pour inverser deux pions, le joueur doit introduire les coordonnées (X, Y) de chacun des deux pions à inverser.
- On va se focaliser sur les séries de trois pions identiques qui se suivent soit en Vertical, soit en Horizontal.

La Matrice

La structure

La Matrice de jeu va prendre la forme d'un tableau à deux dimensions dont chaque élément est une case.

Chaque case du jeu sera implémentée par une structure comprenant les données suivantes :

- Type Pion : indique le type de Pion occupant la case du jeu
- Gélatine : Oui/Non

Pour chaque niveau, on doit aussi pouvoir gérer un ensemble de données :

- Le nombre de coups maximum qui peuvent être joués

Les fonctions

Il nous faut les fonctions suivantes :

- L'initialisation de la Matrice qui doit recevoir la Matrice en paramètre
- Suppression-V : trois pions identiques ont été décelés en position Verticale. Il s'agit de les supprimer du jeu et faire glisser les pions du dessus. Si une case renseignée contenait une gélatine alors elle sera supprimée. On fait descendre les pions et on fait entrer aléatoirement de nouveaux pions. La fonction doit recevoir la Matrice et l'action à exécuter en paramètre.

- Suppression-H : trois pions identiques ont été décelés en position Horizontale. Il s'agit de les supprimer du jeu et faire glisser les pions du dessus. Si une case renseignée contenait une gélatine alors elle sera supprimée. On fait glisser les pions des colonnes d'une position vers le bas et on fait entrer un pion aléatoire dans chaque colonne. La fonction doit recevoir la Matrice et l'action à exécuter en paramètre.
- Verification : calcule si oui ou non l'utilisateur a gagné. Le joueur a gagné s'il ne reste plus aucune gélatine.
- Calcul : action générée lorsque l'utilisateur à intervertit deux cases. Il s'agit de calculer si trois pions se suivent en Vertical ou en Horizontal. Si trois pions se suivent en vertical, la fonction devra ajouter une action « Suppression V » sur la Queue. Si trois pions se suivent en horizontal, alors il faut ajouter une action « Suppression H » sur la Queue. Si la Queue est pleine, il faut afficher un message d'erreur et arrêter le programme.
- Déplacement : Cette fonction reçoit en coordonnée les deux pions qui doivent être intervertis. La fonction va effectivement les intervertir.

L'Action

Une Action doit être réalisée sur le jeu.

Une Action comprendra les données suivantes :

- Le nom de l'action : tableau de caractères de taille 20
- Les coordonnées X et Y d'un premier pion en format integer
- Les coordonnées X et Y d'un autre pion en format integer

L'Affichage

Les Fonctions

Il nous faut les fonctions suivantes :

- Fonction d'affichage qui reçoit la Matrice en paramètre et va afficher les deux grilles de jeu (en version simplifiée) : une avec les pions et l'autre avec la gélatine. En version plus complexe, un affichage couleur est possible avec la gélatine en arrière-fond.
- Fonction Lecture : fonction qui retourne les coordonnées X et Y des deux pions qui doivent être déplacés.

La Queue

Pour faire fonctionner le jeu, nous allons utiliser une file (Queue) qui contiendra les actions qui doivent être réalisées. Nous l'appellerons Action Queue. Dans cette Queue, nous y placerons les actions à réaliser.

Cette Queue prendra la forme d'un tableau qui pourra avoir une taille dynamique. Chaque case du tableau étant une action.

Implémentation de la Queue

Une file ou une queue est une structure de données qui satisfait à l'ordre FIFO : First In, First Out. La première action introduite sera la première à être traitée.

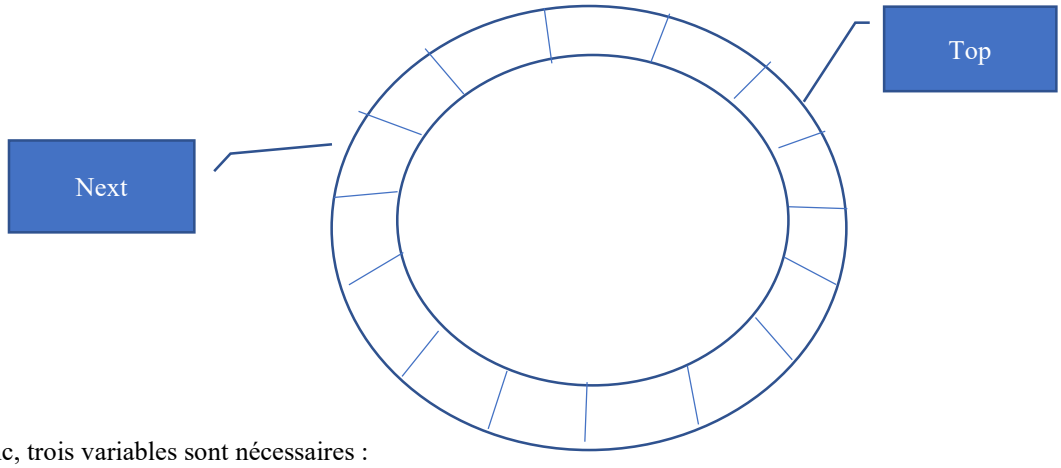
Une Queue peut être implémentée de différentes manières. Dans le cadre de cet exercice, on va utiliser un tableau. Dans une version simplifiée de l'exercice, on peut travailler avec un tableau de taille fixe. Dans une version plus complète, on peut travailler avec un tableau de taille dynamique.

Action 4	Action 5						

Dès qu'une action a été traitée, il faut la retirer de la Queue. Cela voudrait dire qu'il faut faire reculer toutes les autres actions vers le sommet de la Queue :

Action 9	Action 10		Action 4	Action 5	Action 6	Action 7	Action 8

Mais, on peut éviter ces mouvements si on voit ce tableau comme un anneau :



Donc, trois variables sont nécessaires :

- **Top** : Index du premier élément dans la Queue
- **Next** : Index du prochain élément libre dans la Queue
- **Size** : Index Maximal de la Queue

Dans la Queue, il faut évidemment aussi y placer le tableau avec les Actions.

Trois fonctions sont à écrire :

1) L'initialisation qui initialise la Queue
2) La fonction Add qui permet d'ajouter une action que l'on passe en paramètre. La fonction Add doit aussi indiquer si l'insertion a pu se faire
3) La fonction Get qui retourne une action. Le Get doit aussi indiquer si une action a pu être retournée

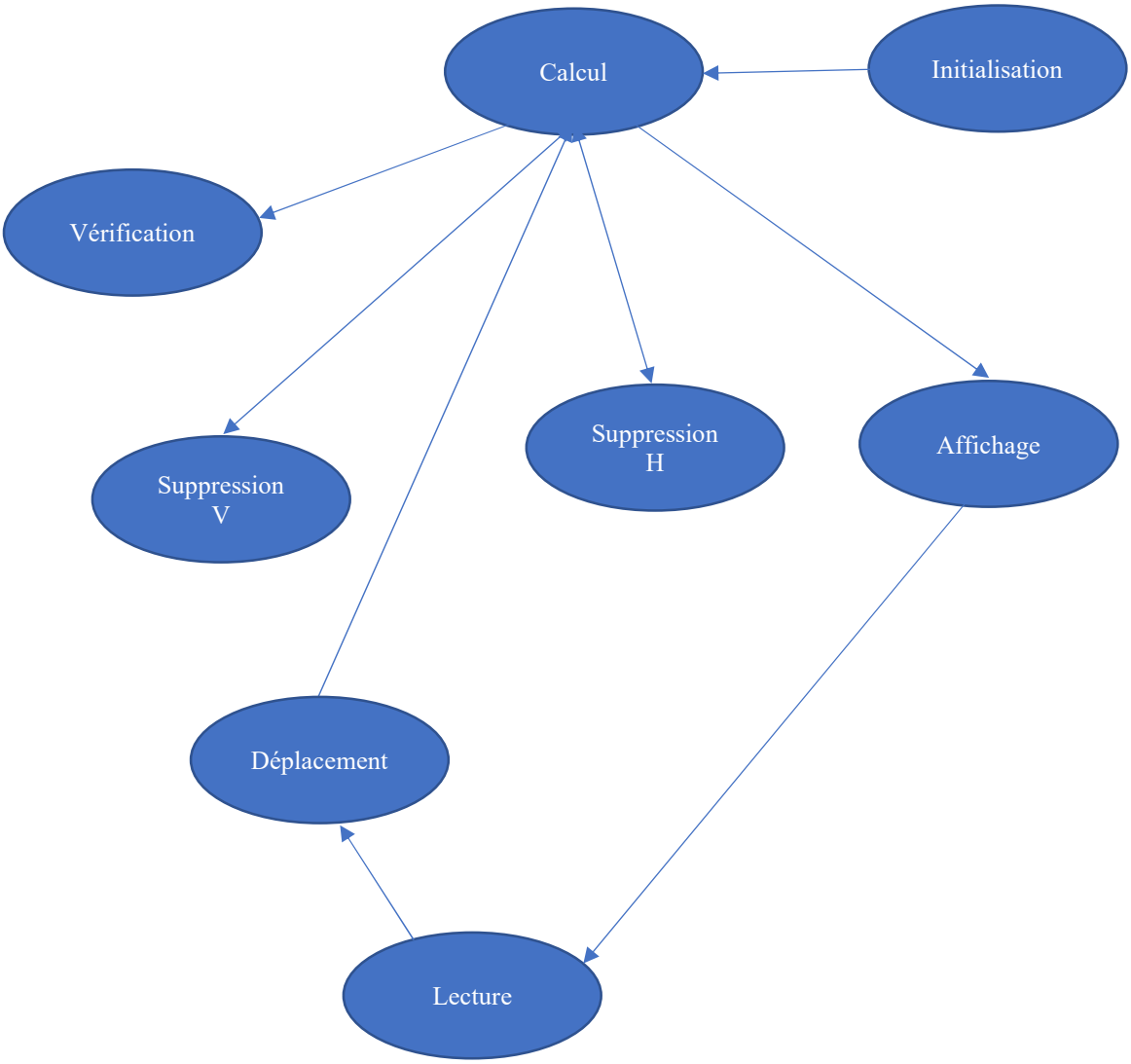
Génération des Actions

Voyons maintenant comment les actions vont être générées. Quand une action est terminée alors l'action sera retirée de la Queue.

Action	Description
CALCUL	Cette action doit être générée dès que l'action LECTURE est terminée. Si on détecte que trois pions sont alignés en vertical, alors il faut placer une nouvelle action dans la Queue. - SUPPRESSION V avec X et Y du pion supérieur et le X et le Y du pion inférieur. Si on détecte que trois pions sont alignés en horizontal, alors il faut générer une nouvelle action dans la Queue : - SUPPRESSION H avec X et Y du premier pion et X et Y du dernier pion. Une seule Action peut être placée dans la Queue.

	Si on ne détecte aucun alignement de pions alors on place les actions VERIFICATION et LECTURE dans la queue
AFFICHAGE	Cette action a pour effet d'effacer l'écran et de redessiner les deux matrices : - La matrice avec l'emplacement des pions. Chaque case contient la première lettre du pion qui occupe la case. - La matrice avec l'emplacement de la gélatine. On ajoute l'action LECTURE dans la Queue.
SUPPRESSION V	- Suppression des trois pions identifiés en position verticale - Suppression des éventuelles gélatines - Faire glisser les pions du dessus pour occuper les trois pions supprimés - Pour les trois nouveaux pions à faire entrer, il faut une génération aléatoire de pions. - Placer l'action CALCUL dans la queue
SUPPRESSION H	- Suppression des trois pions identifiés en position horizontale - Suppression des éventuelles gélatines - Faire glisser les pions des trois colonnes du dessus pour occuper les trois pions supprimés - Pour les trois nouveaux pions à faire entrer, il faut une génération aléatoire de pions. - Placer l'action CALCUL dans la queue
VERIFICATION	- Il faut examiner la grille avec la gélatine en vue de vérifier s'il y a encore de la gélatine - S'il y en a encore, on ne fait rien - S'il n'y en a plus alors il faut ajouter l'action FIN NIVEAU dans la queue
LECTURE	- Lecture des coordonnées X et Y des deux cases qui doivent être interverties On ajoute l'action CALCUL dans la Queue.
INITIALISATION	Cette fonction va initialiser le début d'un nouveau niveau On ajoute l'action CALCUL dans la Queue afin de traiter les éventuelles séquences de pions qui seraient déjà existantes dans la Matrice.
DEPLACEMENT	Cette action va déclencher l'inversion des deux pions dont les coordonnées ont été introduites. La fonction va générer l'action CALCUL.

Séquence des Actions



Organisation des Fonctions

Dans cet exercice, il y a aussi un grand nombre de fonctions à écrire. On va essayer de les organiser dans une optique qui s'approche de la programmation orientée objets.

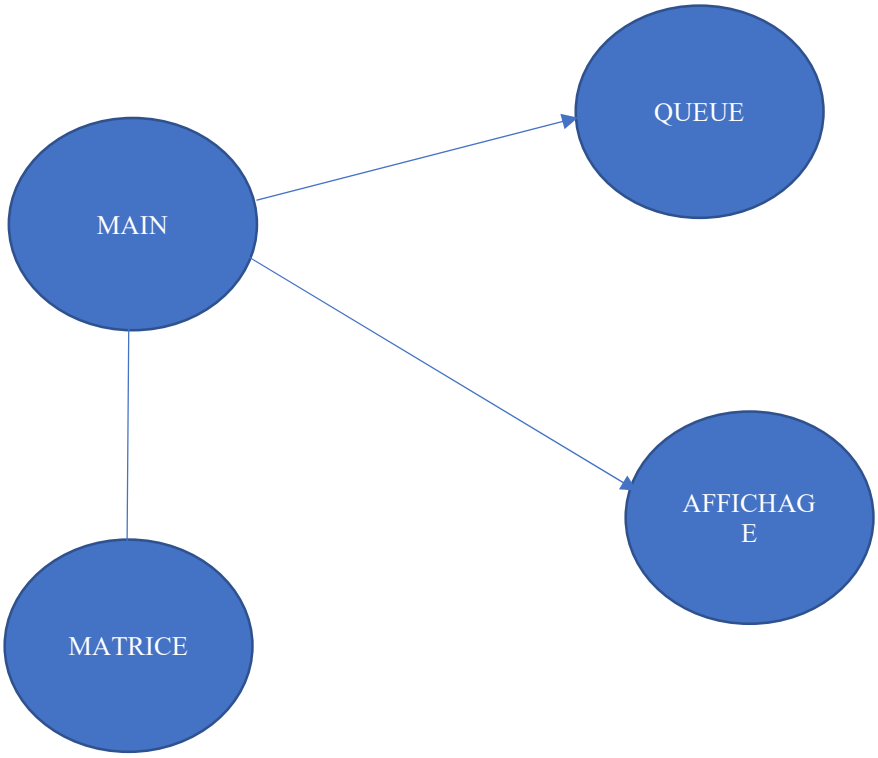
Le schéma ci-dessous montre comment organiser les fonctions. Chaque cercle aura son fichier.cpp :

- Main.cpp
- Queue.cpp + Queue.h. Le fichier .h va contenir la définition de la structure nécessaire au fonctionnement de la Queue. Le fichier Queue.cpp va contenir la liste de toutes les fonctions permettant de faire fonctionner la Queue.
- Matrice.cpp + Matrice.h. Le fichier .h va contenir la définition de la structure nécessaire à la mémorisation des deux matrices. Le fichier Matrice.cpp va contenir toutes les fonctions nécessaires pour la manipulation des Matrices.
- Affichage.cpp + Affichage.h. Le fichier Affichage.cpp va contenir la liste des fonctions nécessaires à l'affichage.

Dans la fonction Main(), on va y trouver la déclaration :

- D'une variable de type Queue
- D'une variable de type Matrice

Ces deux variables seront souvent passées en paramètres de fonctions.



On va essayer que :

- Seules les fonctions se trouvant dans le fichier Queue savent comment la Queue fonctionne
- Seules les fonctions se trouvant dans le fichier Affichage savent comment l'affichage se réalise
- Seules les fonctions se trouvant dans le fichier Matrice savent comment les matrices fonctionnent

Pseudo-Code général

Voici un pseudo code qui donne la structure générale de la solution.

PGM

Niveau = 1

Tant Que Niveau <= 3

AddQueue (INITIALISATION)

Tant Que Action = GetQueue() > 0

Si Action = AFFICHAGE
Alors Appel de la fonction Affichage-AfficherMatrices avec comme paramètres
- Matrice de jeu
- Queue

Si Action = LECTURE
Alors Appel de la fonction Affichage-LirePionAChanger avec comme paramètres
- X et Y du premier pion
- X et Y du deuxième pion
- Queue

Si Action = CALCUL
Alors Appel de la fonction Matrice-Calcul avec comme paramètres
- X et Y du premier pion
- X et Y du deuxième pion
- Matrice de jeu
- Queue

Si Action = SUPPRESSION-V
Alors Appel de la fonction Matrice-SuppressionV avec comme paramètres :
- X et Y du premier pion
- X et Y du deuxième pion
- Matrice de jeu
- Queue

Si Action = SUPPRESSION-H
Alors Appel de la fonction Matrice-SuppressionH avec comme paramètres :
- X et Y du premier pion
- X et Y du deuxième pion
- Matrice de jeu
- Queue

Si Action = VERIFICATION
Alors Appel de la fonction Matrice-Verification avec comme paramètres :
- Matrice de jeu

Si Action = INITIALISATION
Alors Appel de la fonction Matrice-Initialisation avec comme paramètres :
- Matrice de jeu

Fin Tant Que

Fin Tant Que

Ordre des fonctions à écrire

1) Le fichier avec les fonctions de la Queue
2) Création dans le main de la Matrice → création du fichier Matrice et écriture de la fonction Initialisation.
3) Création des fichiers Affichage.h et Affichage.cpp avec la fonction Affichage
4) Écriture des fonctions : Calcul, SuppressionH et SuppressionV
5) Écriture de la fonction Main et des quelques fonctions restantes comme la fonction de lecture des pions à intervertir