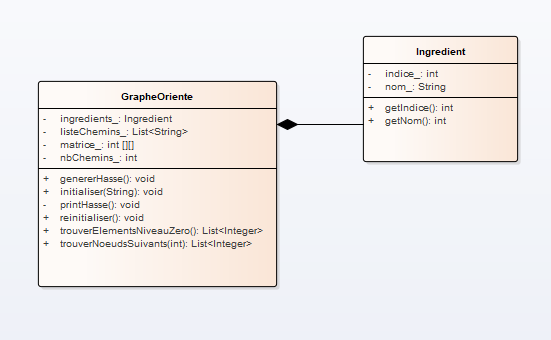
1. Drones
2. Déjeuner et dessert

* Diagramme de Classes



* Explication de la Solution

La première étape est de lire et d’enregistrer les données obtenues à partir de la lecture du fichier fourni pour initialiser les attributs. Afin de bien représenter les différents éléments à manipuler, il était nécessaire de créer des classes pour faciliter la gestion et l’encapsulation des données. A cet effet, nous avons créé deux classes : Ingredient et GrapheOriente.

La Classe Ingredient représente un Ingredient qui est caractérisé par deux informations : le nom de l’ingrédient et l’indice qui lui est associé. Quant à elle la classe GrapheOriente représente le Graphe en soit, contenant ainsi la matrice des arcs, les ingrédients utilisés, le nombre de chemins possibles sur le Diagramme de Hasse et la liste de tous les chemins possibles du diagramme de Hasse.

En effet, la matrice des arcs est en deux dimensions permet d’initialiser toutes les données. On enregistre donc tous les arcs entre les Nœuds. Si un arc existe entre deux nœuds (de i vers j), alors on insère à la position [i][j] la valeur 1 et -1 à la valeur [j][i] afin de respecter l’orientation des arcs. Cette matrice est la base pour réaliser toutes les tâches suivantes à savoir l’affichage du graphe Orienté et la génération du diagramme de Hasse.

Pour l’affichage du Graphe orienté, l’idée est assez simple. Il suffit de :

* Parcourir le tableau des ingrédients
* Récupérer chaque ingrédient
* Aller vérifier les ingrédients avec lesquels il est en relation et
* Afficher ces ingrédients dans le format demandé

La méthode CreerGrapheOriente de la classe GrapheOriente se base sur cet algorithme pour afficher le graphe Oriente des Ingredients.

En ce qui concerne l’affichage du diagramme du Hasse (méthode genererHasse( ) ) l’algorithme utilisé est le suivant :

* Identifier tous les minimaux du Diagramme du Hasse
* Parcourir chaque minimal et chercher les nœuds qui sont juste après lui. Cette partie est faite de façon récursive afin de réduire la longueur du code. Ainsi, sur chacun des nœuds suivants identifiés, on reprend la procédure jusqu’au maximaux tout en tenant compte des différentes règles du diagramme de Hasse
* Enregistrer au fur et à mesure qu’on parcourt les différents nœuds la liste des chemins
* Afficher les différents chemins obtenus au format demandé
* Difficultés rencontrées

Pour la réalisation du diagramme de Hasse :

* Intégration des règles de conception du diagramme de Hasse dans le code
* Difficulté de débogage avec les fonctions récursives
* Solutions potentielles
* Créer de nouvelles méthodes pour limiter le travail effectué par la fonction récursive
* Intégration progressive des règles du diagramme de Hasse au code (une fonction traite juste la réflexivité et une autre la transitivité)