TP C++ n°3 : Gestion des entrées / sorties

Sauvegarde et restitution d’un catalogue de trajets

# Objectifs

L’objectif de ce TP est de reprendre le catalogue de trajets créé au TP précédent et de lui ajouter deux fonctionnalités : la sauvegarde de trajets dans un fichier et l’importation de trajets depuis un fichier. Cette importation et cette sauvegarde sont accompagnées d’options que l’utilisateur peut choisir.

# Description du fichier de sauvegarde

|  |  |
| --- | --- |
| 1  2  3  4 | s,Lyon,Bordeaux,Train;  c,Lyon,Paris,{s,Lyon,Marseille,Bateau;s,Marseille,Paris,Avion;}  s,Lyon,Paris,Auto; |

Figure : Contenu du fichier demo

Dans les fichiers de sauvegarde, il y a un trajet par ligne et chaque trajet est décomposé en 4 sections séparées par des virgule suivi d’un caractère pour indiquer la fin du trajet. Nous avons fait ce choix pour la praticité et la lisibilité du fichier. En effet un caractère de séparation entre les informations des trajets est indispensable car on ne connait pas à l’avance la taille des noms de ville. De plus, avoir un caractère séparateur entre les trajets permets de lire un trajet entièrement en une commande. Enfin, le retour à la ligne sert uniquement à lisibilité, créé une séparation claire et visuelle entre les différents trajets et permet dans une certaine mesure de garder un maximum d’information sur l’écran.

La première section est composée d’un caractère indiquant si le trajet est un trajet simple ou un trajet composé, ‘s’ pour un trajet simple et ‘c’ pour un trajet composé. Avoir cette information en début de ligne, un endroit qui est fixe, permet de déterminer très rapidement le type de trajet sans avoir à faire d’analyse plus poussée

Les deuxièmes et troisièmes sections sont respectivement le nom de la ville de départ et le nom de la ville d’arrivée.

La quatrième section varie en fonction du type de trajet. Pour un trajet simple, elle est le mode de transport suivi d’un point-virgule pour indiquer la fin du trajet simple. Pour un trajet composé, elle est la liste des trajets composant le trajet composé. Les trajets de cette liste sont sous le même format que les autres trajets du fichier de sauvegarde mais il n’y a pas de retour à la ligne entre chaque trajet. La liste de ces trajets est mise entre accolade. L’accolade fermante permet de déterminer la fin de la liste de trajet et donc du trajet composé. L’accolade ouvrante quant à elle n’est pas nécessaire mais accroit grandement la lisibilité. On peut remarquer que grâce à cette accolade fermante il est possible d’importer des trajets composés, composés eux même de trajets composés. En effet, le programme va importer ce trajet comme un trajet composé normal puis revenir au trajet composé de départ sans soucis (nous avons néanmoins dû créer une variable qui indique si le trajet est dans un trajet composé afin de régler certains soucis de lecture qui se présentaient).

Pour les trajets composés, la deuxième et troisième section créé une redondance d’information. En effet, on pourrait déterminer la ville de départ et d’arrivée en analysant la liste de trajets qui compose le trajet principal. Mais cela serait couteux et peu efficace lors d’un chargement de trajets ayant un départ et/ou une arrivée spécifiée dans les options que peut choisir l’utilisateur. Ainsi nous avons décidé de les mettre dans des sections a part, cela permet également d’avoir un format très similaire au trajet simple et ainsi de traiter les trajets indépendamment de leur type pour le chargement avec départ et/ou arrivée spécifiés.

# Description de la répartition des fonctionnalités

Classe Catalogue

La classe catalogue est responsable de l’ouverture des fichiers et la création des flux d’entrée et de sortie. C’est elle qui s’assure que le fichier peut être ouvert lors d’une importation et qui indique un message d’erreur si ce n’est pas le cas. De plus, c’est le catalogue qui s’assure que les conditions soient vérifiées pour les options :

* Choix du type de trajet à importer/exporter
* Sélection de trajets à importer/exporter sur un intervalle

Classe Trajet et ses descendants

Les classes descendantes de la classe Trajet implémentent les méthodes d’exportation vers un flux de sortie « ExporterTrajets » et d’importation depuis un flux d’entrée « ImporterTrajets ». Ce sont elles qui vérifient si les conditions de l’option import/export selon la ville de départ et/ou arrivée sont vérifiées. Elles font remonter cette information au catalogue grâce à un entier. Cela permet d’éviter de faire un saut de ligne si l’exportation du trajet ne remplis pas les conditions par exemple.

# Conclusion

## Problèmes marquants

Pour le fichier de sauvegarde, nous avions initialement prévu de mettre des points-virgules uniquement dans la liste de trajets des trajets composés puis nous avons remarqués que mettre des points virgule a la fin de tout les trajets simples permettait de réutiliser facilement la fonction d'importation des trajets simples. Ainsi, nous avons pu n’utiliser qu’une seule fonction que ce soit pour importer mais aussi exporter un trajet simple « normal » ou un trajet simple faisant partie d’un trajet composé.

Un autre problème marquant a été à l’origine de multiples modifications du format de notre fichier de sauvegarde afin de trouver la meilleure manière de représenter les trajets. En effet, nous avons rapidement compris que nous devions faire un compromis entre la lisibilité du ficher pour l’application et la lisibilité du fichier par un humain, pour pouvoir facilement tester ces fonctionnalités.

## Axes d’évolution et d’amélioration

Comme axe d’amélioration, nous avons pensé qu’il serait intéressant de créer un nouveau format pour enregistrer nos trajets qui serait plus optimisé et demanderait moins d’espace mémoire. Ce serait utile si nous devions pouvoir stocker les données d’une application qui aurait un très grand nombre de trajets ou si la machine qui exécute notre application est très limitée en espace mémoire.