INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES LYON

DAME DIENG – ANH KHOA BUI

COMPTE RENDU DU TP N°2 DE C++-B3113

Ce document est destiné à expliquer le travail effectué durant les deux séances de TP et en dehors de celles-là. L'objectif est de détailler les codes sources l'accompagnant en vue de mettre en exergue les travaux effectués.

I. DESCRIPTION SIMPLE DES CLASSES

Notre application est composée de cinq classes : la classe Trajet, la classe Trajet_simple, la classe Trajet_compose, la classe Liste_chainee et la classe Catalogue.

1) LA CLASSE TRAJET

Cette classe est la classe mère des classes Trajet_simple et Trajet_compose. Il est constitué d'un constructeur (classique c'est-à-dire avec des paramètres) qui construit un trajet en prenant sa ville de départ et sa ville d'arrivée, des attributs protégés en l'occurrence deux pointeurs qui pointent deux chaines de caractères représentant les villes de départ et d'arrivée, d'un constructeur par défaut, d'un destructeur et de méthodes publiques membres de la classe. Ces méthodes sont principalement la méthode Afficher et les getters (pour accéder aux attributs protégés).

- La méthode Afficher() : elle affiche les villes de départ et d'arrivée d'un trajet
- La méthode getDepart() : elle permet d'accéder, en dehors de la classe, au pointeur pointant la ville de départ d'un trajet
- La méthode getArrivee() : elle permet également d'accéder au pointeur pointant la ville de départ

2) LA CLASSE TRAJET_SIMPLE

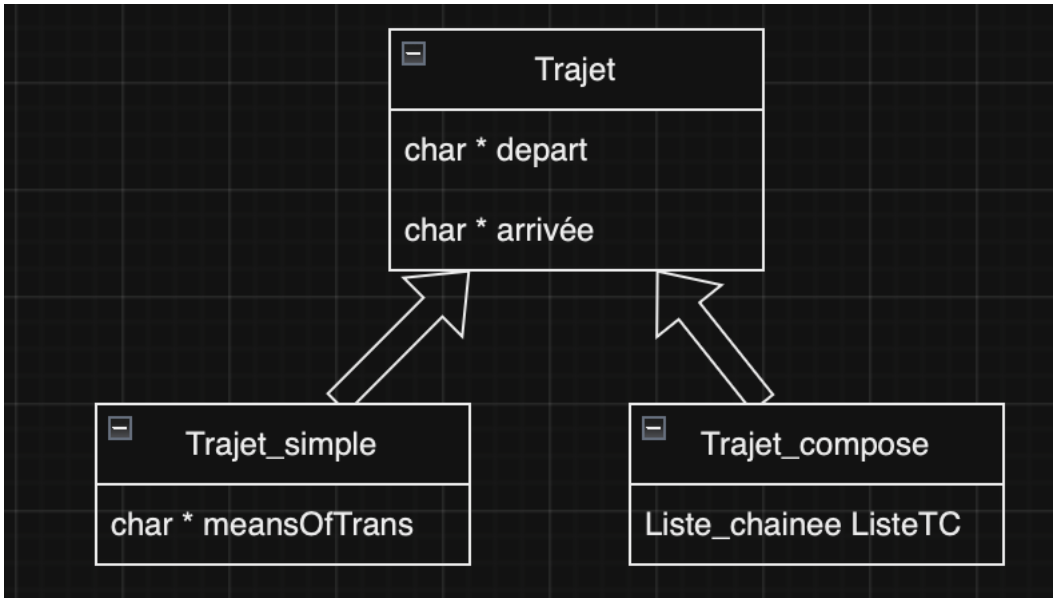
Cette classe hérite de la classe Trajet en héritage publique. Elle a, en plus des caractéristiques d'un trajet, un attribut propre qui est un pointeur pointant une chaine de caractères définissant le moyen de transport d'un trajet simple. Elle possède un constructeur (classique), un destructeur et une méthode publique membre de la classe qui permet d'afficher les villes de départ et d'arrivée d'un trajet simple ainsi que le moyen de transport utilisé.

3) LA CLASSE TRAJET_COMPOSE

De même que la classe Trajet_simple, la classe Trajet_compose hérite de la classe Trajet en héritage publique. Elle représente une collection **ordonnée** de trajets qui peuvent-être simples ou composés. La structure de données retenue pour bâtir cette collection ordonnée est la liste chainée qui sera traitée dans les parties qui suivent. La liste chainée est un attribut protégé de la classe et elle contient des pointeurs sur des trajets.
La classe Trajet_compose a deux méthodes publiques membres de la classe : une méthode Afficher qui affiche la ville de départ, la ville d'arrivée et la ville intermédiaire et une méthode Ajouter qui ajoute un trajet dans la liste chainée. Cette méthode Ajouter tâchera de vérifier que le trajet qu'on veut ajouter est possible c'est-à-dire qu'il respecte le caractère ordonné de la liste. En plus, elle contient un constructeur par défaut et un destructeur.

4) GRAPHE D'HERITAGE

Un trajet simple et un trajet composé sont des sortes de trajets. C'est la raison pour laquelle nous avons mis en place un héritage publique.
La répartition des caractéristiques est dirigée par le sujet. En effet, un moyen de transport n'est défini que pour un trajet simple alors il est tout à fait naturel que l'on mette un attribut « *meansOfTrans » que pour cette classe. De meme, l'attribut de type « Liste_chainee » dans la classe trajet_compose se justifie par le fait qu'un trajet composé est une collection ordonnée de trajets.

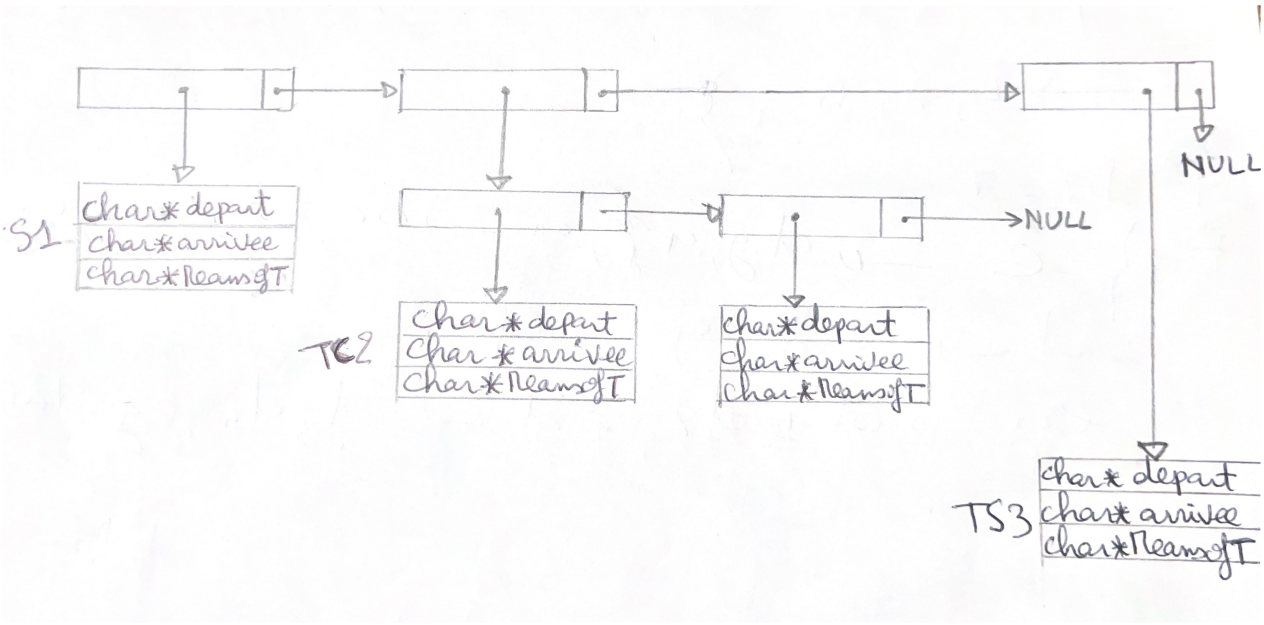


Graphe d'héritage de la classe Trajet

5) LA CLASSE LISTE_CHAINEE

Le développement de notre application requiert une structure de données adaptée à son contexte. Nous avons choisi une liste chainée composée de nœuds et chaque nœud contient un pointeur vers un trajet et un pointeur vers le nœud suivant. Cette classe a un attribut protégé qui est un pointeur vers le premier nœud : c'est le point d'entrée de la liste chainée. Elle possède un constructeur par défaut, un destructeur et des méthodes publiques membres de la classe. Ces méthodes sont principalement au nombre de trois : une méthode append, une méthode Afficher et un getter. Ce dernier permet d'accéder à la tête de la liste chainée qui est un attribut protégé. La méthode append permet d'ajouter un trajet dans la liste chainée en allouant dynamiquement un nœud pour le rattacher au dernier.

6) DESSIN DE LA MÉMOIRE



7) LA CLASSE CATALOGUE

La classe Catalogue utilise toutes les classes de l'application. Catalogue est une collection de trajets mais à la différence de la classe trajet_compose, sa collection n'est pas ordonnée. Elle contient un attribut protégé de type « Liste_chainee », un constructeur par défaut, un destructeur et des méthodes publiques membres de la classe. Elle possède deux principales méthodes : la recherche de trajets et l'ajout de trajets dans le catalogue. La méthode ajouter fait appel à la méthode append de la classe Liste_chainee tandis que la méthode « search » permet de recueillir une ville de départ et une ville d'arrivée sur l'entrée standard, de vérifier qu'il existe un trajet dans le catalogue qui donne les mêmes villes de départ et d'arrivée et les afficher. Cette dernière effectue une recherche simple.