Simulation de déplacements sur le campus d'Orsay

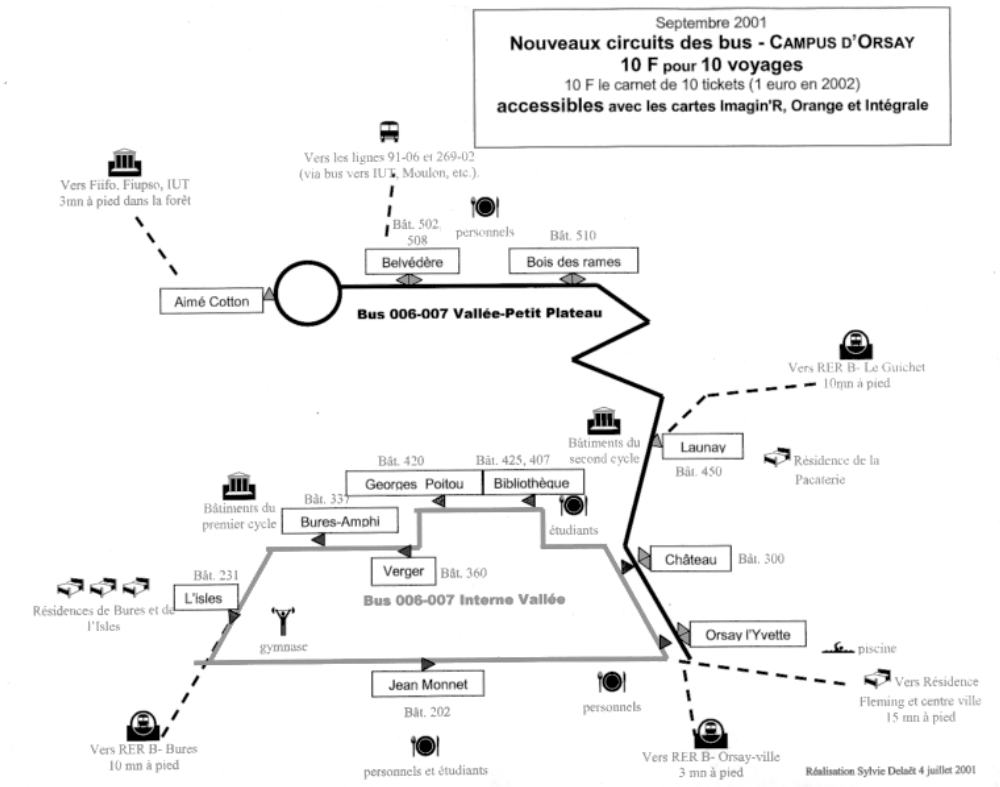
On souhaite écrire un logiciel permettant de savoir comment se déplacer à l'intérieur du campus d'Orsay suivant différents critères, par exemple en minimisant le temps de trajet, ou la marche à pied, ou encore la distance parcourue.

Les lieux pertinents sont des bâtiments de l'Université (par exemple « Maison de l'Ingénieur ») et les arrêts de bus de la ligne 07 (par exemple « Aimé Cotton ») qui dessert le campus. Cependant, votre modèle doit permettre d'ajouter facilement de nouveaux types de lieux (par exemple des intersections de routes ou des points de repères paysagers) ou de nouvelles lignes de bus, avec des correspondances à certains arrêts.

Entre deux lieux décrits comme « voisins » (par exemple « Maison de l'Ingénieur » et « Aimé Cotton ») on peut effectuer le trajet à pied. On suppose que le temps de trajet est lié au couple de lieux voisins, c'est-à-dire que le temps de trajet dépend du sens du trajet. Un trajet à pieds entre deux lieux éloignés sera décomposé en une séquence d'étapes reliant des lieux « voisins » .

Deux arrêts de bus (par exemple « Château » et « Launay ») peuvent être parcourus en bus s'ils font partie du même itinéraire, dans le bon ordre et à un horaire auquel il passe encore des bus. La ligne 07 propose par exemple deux itinéraires : « Vallée » et « Petit Plateau », mais il doit être possible d'ajouter facilement de nouveaux itinéraires à votre modèle. Un itinéraire est simplement une séquence d'arrêts de bus. Le temps de trajet en bus entre deux arrêts successifss est supposé fixé pour chaque couple d'arrêts. A un itinéraire, on associe un ensemble d'horaires, chaque horaire correspond à une tournée et indique pour arrêt l'heure de passage du bus à cet arrêt.

Pour l'utilisateur, on appelle « trajet » une succession d'étapes dont chacune précise le départ, l'arrivée, l'heure de départ ainsi que le moyen de transport et qui s'enchaînent globalement.



Exemple : Considérons que l'on souhaite se rendre du « Bâtiment des Colloques » à la « Maison de l'Ingénieur » à une période où les bus sont fréquents. Un trajet raisonnable serait le suivant :

1. Marcher du « Bâtiment des Colloques » (Bâtiment 338) à l'arrêt « Bures Amphi » ;
2. Prendre le bus circuit « Vallée » de l'arrêt « Bures Amphi » à l'arrêt « Orsay l'Yvette » ;
3. Prendre le bus circuit « Petit Plateau » de l'arrêt « Orsay l'Yvette » à l'arrêt « Aimé Cotton » ;
4. Marcher de l'arrêt « Aimé Cotton » à la « Maison de l'Ingénieur ».

Suivant l'heure de la journée (par exemple s'il n'y a plus de bus) une autre solution comporterait plus de marche à pied.

On dispose d'une classe **`Date`** (comprenant à la fois le jour et l'heure, et munie de toutes les méthodes nécessaires) et d'une classe **`Collection`** générique.

On dispose de la méthode (statique) suivante qui simule la récupération des informations propres aux déplacements en bus :

*// heure d'arrivée du prochain bus de l'itinéraire **nomItineraire** à l'arrêt **nomArret** à partir de*
*// **heureCourante**. Renvoie **null** en cas d'arrêt incorrect ou s'il n'y a plus de bus à cette heure.*

```
Date prochainBus(String nomArret, String nomItineraire, Date heureCourante);
```

1. Modéliser les concepts ci-dessus sous la forme d'une hiérarchie de classes. Précisez le caractère abstrait des classes ou des méthodes. Dans cette question, **il n'est pas demandé** d'écrire le code des méthodes que vous définissez pour vos classes.

2. Écrire une méthode qui vérifie la cohérence globale d'un trajet (chaque étape du trajet est réalisable par le mode de transport préconisé, avec des destinations et horaires cohérents).

3. Écrire une fonction pour évaluer la meilleure manière de se rendre d'un lieu à un autre en minimisant le temps de transport. Vous écrirez également les méthodes de la hiérarchie de classes que vous avez définie à la question 1 *et qui sont pertinentes pour répondre à cette* question. On suppose que l'on dispose d'une fonction qui énumère tous les trajets possibles entre deux lieux du campus d'Orsay dont le prototype est le suivant :

```
Liste<Trajet> enumereTrajets(String départ, String arrivée);
```

On demande simplement de choisir parmi ces trajets prédéfinis le meilleur pour le critère considéré, pas d'inventer le meilleur trajet.

4. Généralisez votre solution précédente pour qu'on puisse appliquer un critère quelconque, pas seulement la minimisation du temps de transport.

5. Commentez votre solution par rapport aux possibilités suivantes :

1. ajouter de nouveaux itinéraires ;
2. ajouter de nouveaux types de lieux ;
3. ajouter de nouveaux modes de transport (par exemple de l'auto-stop à partir d'un ensemble de lieux de départ propices. Le temps d'attente et le temps de trajet étant des valeurs estimées grâce à une méthode prenant en paramètre les lieux et l'heure concernés (le code de cette méthode n'est bien sûr pas demandé !).