Simulation de déplacements sur le campus d'Orsay

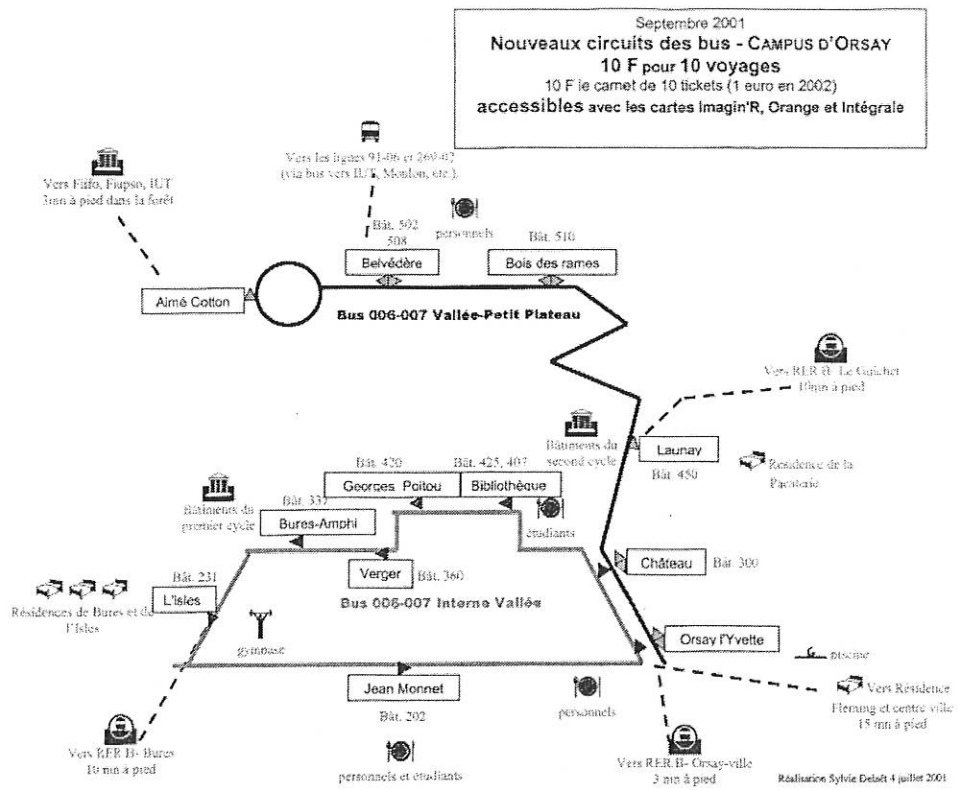
On souhaite écrire un logiciel permettant de savoir comment se déplacer à l'intérieur du campus d'Orsay suivant différents critères, par exemple en minimisant le temps de trajet, la marche à pied, ou encore la distance parcourue.

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).

Entre deux lieux voisins (par exemple « Maison de l'Ingénieur » et « Aimé Cotton ») il est possible d'effectuer le trajet à pied quelle que soit l'heure de la journée. 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 de la traversée.

Deux arrêts de bus voisins (par exemple « Château » et « Launay ») peuvent être parcourus en bus s'ils font partie du même itinéraire. La ligne 07 propose actuellement 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 liste d'arrêts de bus. Le temps de trajet en bus entre deux arrêts de bus voisins est supposé fixé pour chaque couple d'arrêts. A un itinéraire, on associe un ensemble d'horaires, chaque horaire correspondant à une tournée et indiquant pour chaque point de l'itinéraire l'heure de passage du bus en ce point

Pour l'utilisateur, on appelle « trajet » une succession d'étapes (entre deux liens voisins, pour simplifier) dont chacune précise 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 suppose aussi fournie la méthode (statique) suivante qui permet de récupérer les informations propres aux déplacements en bus :

// heure d'arrivée du prochain bus de l'itinéraire ***nomItineraire*** *à l'arrêt* ***nomArret*** *en supposant*
// que l'heure courante est ***heureCourante****. Renvoie* ***null*** *si l'arrêt ne fait pas partie de l'itinéraire.*

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

1. Modéliser les concepts présentés ci-dessus sous la forme d'un ensemble de classes et/ou d'interfaces. 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. Ecrire en pseudo-code une méthode qui vérifie la cohérence globale d'un itinéraire (chaque étape correspond à un trajet réalisable entre les deux lieux par le mode de transport préconisé, avec des horaires cohérents).

3. Ecrire en pseudo-code une méthode pour imprimer un itinéraire. Pour chaque étape on indique les lieux de départ et d'arrivée, le mode de transport, l'heure de départ et l'heure d'arrivée.

4. Ecrire une fonction qui permette d'évaluer la meilleure manière de se rendre d'un lieu à un autre en minimisant le temps de transport. Vous écrirez également sous forme de pseudo-code les méthodes de la hiérarchie de classe 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 (sans cycles) possibles entre deux lieux du campus d'Orsay dont le prototype est le suivant :

```
Liste enumereTrajets(String départ, String arrivée);
```

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 !).
4. ajouter de nouvelles méthodes de comparaison entre trajets pour prendre en compte d'autres critères que la minimisation du temps de transport. Par exemple on pourrait souhaiter minimiser la marche à pieds, ou minimiser le nombre de changements.