

TP Automates finis et actions

Gestion d'une entreprise de location de vélos

Nous nous situons ici dans le cadre d'une application de gestion d'une entreprise de location de vélos pour enfants et pour adultes. L'entreprise possède un stock limité de vélos des deux catégories. L'objectif de ce TP est d'analyser des saisies d'opérations de location afin de facturer ces locations et d'effectuer certains affichages. Un exemple de donnée et des résultats attendus correspondants est fourni section 4.

Les saisies d'opérations de location sont organisées de la manière suivante :

- Chaque jour, une *validation* des opérations est effectuée (le séparateur entre deux validations est le caractère ';').
- Chaque *validation* est constituée d'une ou plusieurs *opération*(s) de début (*début_loc*) ou de fin (*fin_loc*) de location. Le séparateur entre deux opérations est le caractère ','.
- La fin de donnée, symbolisée par le caractère '/' permet d'effectuer un bilan des locations (par exemple mensuellement).

Le langage d'entrée peut ainsi être représenté par les règles de production suivantes :

$\langle \text{suite_validation} \rangle$	$\rightarrow (\langle \text{validation} \rangle ;)^* /$
$\langle \text{validation} \rangle$	$\rightarrow \langle \text{opération} \rangle (, \langle \text{opération} \rangle)^*$
$\langle \text{opération} \rangle$	$\rightarrow \langle \text{début_loc} \rangle$ $\quad \mid \langle \text{fin_loc} \rangle$
$\langle \text{début_loc} \rangle$	$\rightarrow \langle \text{client} \rangle \langle \text{horaire} \rangle^{0/1} \text{ debut } \langle \text{quantité_louée} \rangle$
$\langle \text{quantité_louée} \rangle$	$\rightarrow \langle \text{qté} \rangle \text{ adulte } \langle \text{qté} \rangle \text{ enfant}$ $\quad \mid \langle \text{qté} \rangle \text{ adulte}$ $\quad \mid \langle \text{qté} \rangle \text{ enfant}$
$\langle \text{fin_loc} \rangle$	$\rightarrow \langle \text{client} \rangle \langle \text{horaire} \rangle^{0/1} \text{ fin}$
$\langle \text{client} \rangle$	$\rightarrow \text{ident}$
$\langle \text{horaire} \rangle$	$\rightarrow \text{nbentier heures}$
$\langle \text{qté} \rangle$	$\rightarrow \text{nbentier}$

1 Analyse lexicale

L'analyse lexicale (des données de saisie de location) est à réaliser par un automate fini déterministe avec actions, mis en œuvre par programmation directe (*cf.* TD). L'analyseur doit ainsi reconnaître les items lexicaux correspondant :

- aux mots réservés (items ADULTE, DEBUT, ENFANT, FIN, HEURES),
- aux identificateurs, ici de clients, composés uniquement de lettres minuscules ou majuscules (item ident),
- aux séparateurs ';' , '/' et ',' (items PTVIRG, BARRE et VIRG),
- aux nombres entiers, composés uniquement de chiffres de '0' à '9' (item nbentier).

2 Analyse syntaxique

L'analyse syntaxique est à réaliser par un automate fini déterministe avec actions, mis en œuvre par tables et interpréteur de tables (*cf.* TD).

Le langage d'entrée, décrit ci-dessus, peut ainsi être représenté par l'expression régulière (ou rationnelle) suivante :

```
(( ident (nbentier HEURES)0/1 DEBUT (nbentier ADULTE nbentier ENFANT
| nbentier ADULTE | nbentier ENFANT)
| ident (nbentier HEURES)0/1 FIN )
( VIRG
( ident (nbentier HEURES)0/1 DEBUT (nbentier ADULTE nbentier ENFANT
| nbentier ADULTE | nbentier ENFANT)
| ident (nbentier HEURES)0/1 FIN )
)*
PTVIRG )* BARRE
```

Les actions associées à l'automate d'analyse syntaxique doivent effectuer des traitements sur les opérations de location correctes, et traiter également un certain nombre d'erreurs possibles. L'ensemble des classes Java à utiliser et à compléter est disponible sur Moodle.

2.1 Traitement des données correctes

Les actions doivent permettre :

- de calculer la facturation de la location de chaque client et d'écrire, à chaque fin de location, la somme due par le client venu rendre ses vélos ainsi que son nom,
- d'afficher, à la fin de chaque validation journalière, les noms des clients en cours de location et, pour chacun, le nombre de vélos non rendus de chaque catégorie (adulte et enfant),
- d'indiquer, en fin d'analyse (sur l'item BARRE), le nombre total de vélos non ramenés pour chacune des deux catégories (enfant et adulte) ainsi que le nombre maximum de clients différents servis en une journée,
- d'afficher, en fin d'analyse, le nombre d'opérations correctes (de début ou de fin de location) et le nombre total d'opérations saisies.

Pour faciliter ces actions, certaines structures de données sont fournies. La classe *BaseDeLoc.java* permet de stocker, et de mettre à jour, les locations en cours (clients n'ayant pas encore rendu les vélos loués). De plus, un tableau extensible d'ensembles *clientsParJour* permet de stocker l'ensemble des numéros de clients différents pour chaque jour. Ainsi *clientParJour.get(2)* permet de mémoriser, pour le jour 2, l'ensemble des *numIdent* associés aux clients par l'analyseur lexical.

2.1.1 Horaires de location

L'entreprise reçoit des clients tous les jours entre 8h et 19h. Lors d'une opération de début de location, si l'horaire n'est pas fourni, l'heure de début doit être forcée à 8h. De même, lors d'une opération de fin de location, si l'horaire n'est pas fourni, l'heure de fin doit être forcée à 19h.

La validation d'une journée permet d'enregistrer toutes les opérations de cette journée. Ces opérations sont forcément saisies dans l'ordre dans lequel les clients ont été servis et la fin de location d'un client est toujours saisie après le début de location de ce même client. **Vous n'avez pas besoin de prévoir ces vérifications.**

2.1.2 Facturation des locations

Un client venant rendre des vélos est immédiatement facturé pour tous les vélos empruntés par lui; il ne peut pas venir emprunter de nouveaux vélos sans avoir rendu tous ceux d'un emprunt précédent (le cas échéant).

Le tarif de location est 4€/heure pour les vélos adulte et 2€/heure pour les vélos enfant. Une location de moins d'une heure (si heure de fin = heure de début) est facturée une heure. Les locations peuvent courir sur plusieurs jours consécutifs (mais les heures de 19h à 8h le lendemain ne sont pas comptabilisées).

2.2 Traitement des erreurs

Une reprise du traitement après les erreurs non fatales est exigée (*cf.* TD). Les points de reprise souhaités correspondent aux items *VIRG*, *PTVIRG* et *BARRE*.

Les erreurs non fatales (opérations de début ou de fin de location à ignorer) correspondent aux cas suivants :

- L'heure indiquée n'appartient pas à [8..19].
- Le nombre de vélos demandés (adulte ou enfant ou les deux) est nul.
- La demande de location ne peut pas être complètement satisfaite : dans une catégorie, le nombre de vélos demandés dépasse le nombre de vélos disponibles.
- Toutes les erreurs de syntaxe doivent être signalées et gérées comme erreurs non fatales. Notez que si l'erreur de syntaxe est l'oubli du dernier PTVIRG avant BARRE, les affichages de fin d'analyse (dont la journée avec le maximum de clients différents) doivent être réalisés sans tenir compte de la dernière journée (reprise d'erreur sur BARRE).

Un seul cas d'erreur fatale est à envisager :

- L'opération de restitution de vélos indique une heure inférieure à l'heure de début de location, dans la même journée.

N.B. On considère qu'un client ne peut ni louer de nouveaux vélos alors qu'il en a déjà en location, ni rendre des vélos qu'il n'a pas emprunté. Les noms de clients sont supposés tous distincts (pas d'homonymes).

3 Travail à faire et à rendre

Vous devez mettre en œuvre l'analyseur lexical et l'analyseur syntaxique du langage de cette application de location de vélos selon les consignes données dans les sections 1 et 2. Pour cela, il vous faut définir les automates finis **déterministes** correspondants, et compléter les squelettes de classes Java mis à votre disposition sur Moodle.

L'**analyseur lexical** est à rendre **au cours de la première séance de TP** (fichier LexVelo.java uniquement). Le dessin de l'automate d'analyse syntaxique, avec les numéros d'action, et l'**analyseur syntaxique** lui-même (fichiers AutoVelo.java et ActVelo.java uniquement), ainsi qu'un jeu de test personnel commenté sont à rendre sur Moodle, par trinôme, pour le **mercredi 28 janvier 2026**. Le jeu de test doit mettre en évidence tous les cas testés, corrects et erronés, en commentant ce qui est testé.

4 Exemple

Voici un exemple simple de donnée analysée (fourni dans les fichiers distribués).

```
Durand 8 HEURES DEBUT 5 ADULTE 5 ENFANT,
Dupont 8 HEURES DEBUT 2 ADULTE 1 ENFANT,
Dupont 13 HEURES FIN,
Durand 18 HEURES FIN,
Legros 18 HEURES DEBUT 20 ADULTE 10 ENFANT ,
Legros 19 HEURES FIN; /
```

Cette donnée décrit une seule journée de location, pendant laquelle trois clients louent et rendent une seule fois chacun.

- Dupont 10 euros par heure pendant 5 heures = 50 euros
- Durand 30 euros par heure pendant 10 heures = 300 euros
- Legros 100 euros par heure pendant 1 heure = 100 euros

Pour cette donnée, le traitement et l'affichage attendu sont de la forme suivante.

```
Le client:  DUPONT  doit payer : 50 euros pour 2 vélo(s) adulte et 1 vélo(s) enfant
Le client:  DURAND  doit payer : 300 euros pour 5 vélo(s) adulte et 5 vélo(s) enfant
Le client:  LEGROS  doit payer : 100 euros pour 20 vélo(s) adulte et 10 vélo(s) enfant
```

```
*****
BILAN DU JOUR  1
      CLIENT      JOUR      HEURE_DEBUT  ADULTE      ENFANT
-----
Nombre de vélos adulte manquants :  0
Nombre de vélos enfant manquants :  0
Opérations correctes : 6 - Nombre total d'opérations : 6
```

Voici les clients qui doivent encore rendre des vélos

```
      CLIENT      JOUR      HEURE_DEBUT  ADULTE      ENFANT
-----
```

```
***** BILAN AFFLUENCE *****
Le jour de plus grande affluence est : 1  avec 3 clients servis
```