TP AUTOMATE

Une société de transports livre, par camions-citernes, du vin à différents magasins. La société emploie au plus 10 chauffeurs et chaque camion-citerne possède un volume compris entre 100 et 200 hectolitres.

Lors d'une livraison, le vin contenu dans la citerne appartient à l'une des trois qualités suivantes : BEAUJOLAIS, BOURGOGNE ou ORDINAIRE et peut être destiné à un ou plusieurs magasins.

De temps à autre, chaque chauffeur remplit des fiches de livraison et, en fin de mois, les fiches sont exploitées pour établir des statistiques, par un programme qui admet en entrée une chaîne du langage régulier engendré par la grammaire suivante :

<suite_fiches>	→	{ <fiche> ; }* /
<fiche>	→	<chauffeur> { <volume_citerne> }$^{0/1}$ <livraisons>
<chauffeur>	→	ident
<volume_citerne>	→	nbentier
<livraisons>	→	<livraison> { , <livraison> }*
<livraison>	→	<qualité> { <livraison_magasin> }$^{+}$
<qualité>	→	{ BEAUJOLAIS | BOURGOGNE }$^{0/1}$
<livraison_magasin>	→	<magasin> <quantité>
<magasin>	→	ident
<quantité>	→	nbentier

Remarques :

- les identificateurs sont composés de lettres, les minuscules et les majuscules sont équivalentes,
- on suppose qu'aucun chauffeur, ni aucun magasin, ne s'appelle BEAUJOLAIS ou BOURGOGNE ; par contre, un chauffeur et un magasin peuvent porter le même nom (cf POTIN dans l'exemple ci-contre),
- on suppose, également, que la société n'emploie pas deux chauffeurs portant le même nom,
- lorsque le volume de la citerne n'est pas indiqué en début de fiche, il est considéré comme égal à 100,
- si dans <livraison> la qualité du vin n'est pas fournie, il s'agit de vin ordinaire.

Exemple :

```
DURAND      120
        BEAUJOLAIS      vieillesgaleries    50
                        prixmultiples       70 ,
                prixmultiples 110,
        BOURGOGNE       potin               50
                        vieillesgaleries    40
                        elephant            10;
DUPONT      110
        BOURGOGNE       croisement          70
                        superv              40;
POTIN       140
        BEAUJOLAIS      potin               90 ;
DURAND
                elephant 90;
/
```

⇒

3 chauffeurs : DURAND DUPONT POTIN

6 magasins : vieillesgaleries prixmultiples potin elephant croisement superv

Rq: chaque fiche peut comprendre plusieurs livraisons, séparées par des ',' pour lesquelles la catégorie de vin livrée peut changer (mais la capacité de la citerne est inchangée).

Vous disposez sur le réseau, dans **g:\l3info\comp\tpauto\phase1**, de :

- **Vin.java** programme principal contenant l'interpréteur de tables
- **Autovin.java** squelette de la classe destinée à recevoir la table **transit**
- **Actvin.java** squelette de la classe pour la table **action** et la procédure **executer**
- **Lexvin.class** analyseur lexical proposé pour faciliter les tests des phases 1 et 2
- **SmallSet.class** pour les méthodes ensembles vues en PROG1.

Démarche à suivre

Phase 1

• Trouvez un automate acceptant le langage régulier précédent, complétez ensuite **Autovin.java** en remplissant la table **transit**.

L'analyseur lexical **Lexvin.class** qui vous est proposé "saute" les espaces, convertit toute lettre minuscule en majuscule et **impose le codage** suivant :

BEAUJOLAIS = 0, BOURGOGNE =1, ident = 2, nbentier = 3,
virgule = 4, ptvirg = 5, barre = 6, autres = 7

• Testez, "à vide" (ie. sans actions), votre analyseur syntaxique en compilant et exécutant **Vin.java**

Phase 2

• Prévoyez des actions afin :

- d'afficher, à chaque fin de fiche (c'est à dire sur chaque ;), pour chaque chauffeur présent :
  - le nom cadré sur 20 caractères,
  - la quantité cumulée livrée dans chaque catégorie de vin, présentée dans l'ordre : Beaujolais, Bourgogne, ordinaire,
  - le nombre de magasins différents livrés depuis le début de l'analyse.

Ainsi, pour l'exemple, on veut afficher :

à la fin de la 1ère fiche :

CHAUFFEUR	BJ	BG	ORD	NBMAG
DURAND	120	100	110	4

à la fin de la 2ème fiche :

CHAUFFEUR	BJ	BG	ORD	NBMAG
DURAND	120	100	110	4
DUPONT	0	110	0	2

à la fin de la 3ème fiche :

CHAUFFEUR	BJ	BG	ORD	NBMAG
DURAND	120	100	110	4
DUPONT	0	110	0	2
POTIN	90	0	0	1

à la fin de la 4ème fiche :

CHAUFFEUR	BJ	BG	ORD	NBMAG
DURAND	120	100	200	4
DUPONT	0	110	0	2
POTIN	90	0	0	1

- de forcer à 100 la capacité d'une citerne si la capacité fournie dans la fiche n'appartient pas à l'intervalle 100:200,
- de contrôler que le volume livré à chaque magasin est ≠ de 0 (erreur non fatale),
- de vérifier, pour chaque livraison, que le volume livré ne dépasse pas la capacité de la citerne (erreur non fatale),
- d'abandonner l'analyse s'il y a plus de 10 chauffeurs (erreur fatale),
- de déterminer le nom du (d'un) chauffeur ayant globalement livré le plus de magasins différents.

• Supprimez, dans **Vin.java**, les commentaires masquant les appels à **Actvin**.

• Complétez **Actvin.java**, qui contient déjà la classe *Chauffeur* et l'instanciation du tableau *tabchauf* des chauffeurs, puis testez votre automate incluant les actions.

Remarques

Les attributs lexicaux fournis, en **static**, dans la classe **Lexvin** sont :

- **valNb** pour la valeur du dernier nombre entier lu,
- **numId** pour le codage (entre 2 et 201) du dernier identificateur non réservé lu,
- **String repId** (**int** nid) qui délivre l'identificateur codé par nid.

Phase 3

• Complétez **Lexvin.java,** fourni dans **g:\l3info\comp\tpauto\phase3**, en :

- définissant la fonction **liresymb** (qui doit respecter le codage indiqué dans la phase 1) et la procédure **repId**,
- positionnant les attributs **valNb** et **numId**.

• Testez à nouveau votre programme.

Rendre

• Un dossier contenant le dessin de l'automate et les numéros d'actions.

• Les sources **Vin.java, Autovin.java**, **Actvin.java** et **Lexvin.java** de votre application.