UBFC
UNIVERSITÉ
BOURGOGNE FRANCHE-COMTÉ

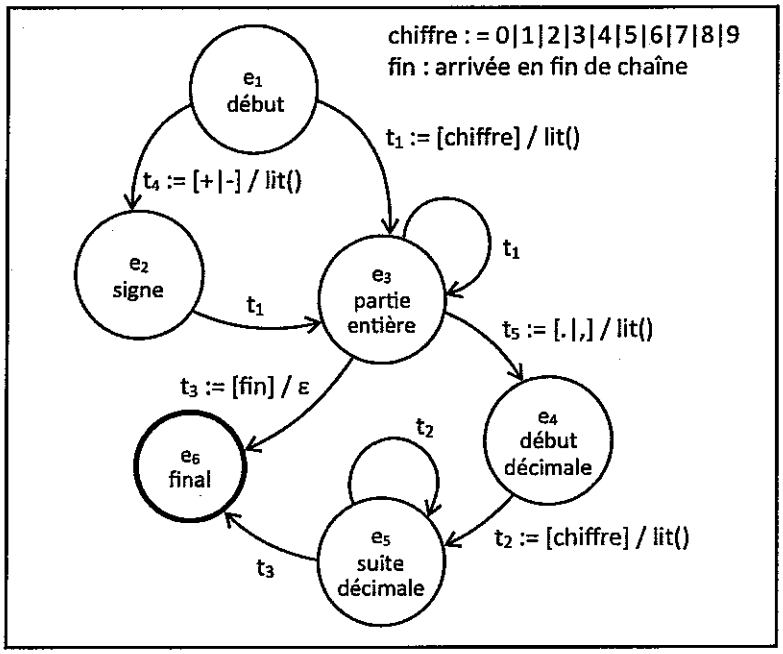
Université de Bourgogne
Année universitaire 2018-2019 - L1
Info 2A — programmation objet
Examen terminal du vendredi 17 mai 2019

Modalités

- ✓ Durée : 2h.
- ✓ Document autorisé : une feuille manuscrite recto-verso non photocopiée.

Description générale

Un automate est une structure dynamique — comme les listes ou les arbres — formée d'*états* (cercles) et de *transitions* (flèches).
Tout processus programmable peut être représenté par un automate, en particulier celui qui permet de vérifier qu'un énoncé respecte bien la syntaxe d'un langage ou d'une partie d'un langage. L'automate de l'exemple permet de lire une chaîne de caractères en vérifiant qu'elle a bien la forme d'un nombre décimal au format américain ou français[1].



Un état représente une étape de la lecture (e_1 : début de la lecture, e_3 : lecture de la partie entière...). L'état terminal (e_6 : final) termine la lecture.
À partir d'un état, il est possible de poursuivre la lecture vers un autre état relié à lui par une transition à condition que la condition associée à la transition soit respectée. Dans l'exemple, la transition t_1 permet de passer de l'état e_1 (état de début) à l'état e_3 (lecture de la partie entière) à condition que le caractère courant (le prochain caractère à lire) soit un chiffre[2]. La même transition permet de continuer la lecture de la partie entière tant que la condition est respectée (qu'il y a un chiffre à lire). Elle est aussi utilisée pour passer de e_2 à e_3, c'est-à-dire de la lecture du signe à celle de la partie entière. Dans la même logique, si la lecture est dans l'état e_3, la lecture peut être continuée vers l'état e_4 à condition que le caractère courant soit '.' ou ',' [3]. En dernier lieu, dans la transition t_3, « [fin] / ε » signifie que, s'il n'y a plus de caractère à lire, c'est-à-dire que la lecture atteint la fin de la chaîne (condition fin), aucune action n'est exécutée au cours de la transition (ε).

---

[1] Un signe optionnel : '+' ou '-', puis un ou plusieurs chiffres pour la partie entière, puis une partie décimale optionnelle commençant par '.' ou ',' et suivie par un ou plusieurs chiffres.

[2] Dans l'exemple, l'écriture nom := expression signifie que nom est un raccourcis d'écriture pour expression.

[3] L'expression « [.|,] / lit() » est sous la forme [condition]/action. Elle signifie que, si la condition [.|,] est remplie, la transition peut être faite et que l'action lit() peut être effectuée. [c_1|c_2|...c_n] est une condition qui exprime que le caractère courant doit être c_1 ou c_2 ou... c_n (dans l'exemple, '.' ou ','). lit() exprime la lecture du caractère courant et l'avancée du curseur de lecture dans la chaîne.

Utilisation d'un automate

Un automate s'utilise en plaçant la chaîne de caractères à tester « dans » l'état de début (ici e_1) associée à un curseur de lecture initialisé à 0 (indice du premier caractère de la chaîne). Les transitions sont ensuite effectuées d'état en état jusqu'à l'état **final** ou, sinon, il y a une erreur. Ce dernier cas se produit quand, dans un état donné, le caractère courant ou l'arrivée en fin de chaîne ne sont prévus par aucune des conditions des transitions disponibles.
Par exemple, avec la chaîne "-123,08", on obtient la séquence :

état	curseur	caractère courant	transition	action
e_1	0	'-'	t_4	lecture, vers e_2
e_2	1	'1'	t_1	lecture, vers e_3
e_3	2	'2'	t_1	lecture, vers e_3
e_3	3	'3'	t_1	lecture, vers e_3
e_3	4	','	t_5	lecture, vers e_4
e_4	5	'0'	t_2	lecture, vers e_5
e_5	6	'8'	t_2	lecture, vers e_5
e_5	7	fin de chaîne atteinte	t_3	ε, vers e_6
e_6				affichage succès

Pour la chaîne "12-3,08", cela donne :

état	curseur	caractère courant	transition	action
e_1	0	'1'	t_1	lecture, vers e_3
e_3	1	'2'	t_1	lecture, vers e_3
e_3	2	'-'	ε	erreur : caractère inattendu

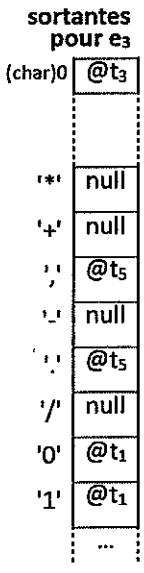
Dernier exemple avec la chaîne "+2." :

état	curseur	caractère courant	transition	action
e_1	0	'+'	t_4	lecture, vers e_2
e_3	1	'2'	t_1	lecture, vers e_3
e_3	2	'.'	t_5	lecture, vers e_4
e_3	3	fin de chaîne atteinte	ε	erreur : fin prématurée

Modélisation objet

Un automate peut être représenté par ses composants : les états et les transitions. Pour modéliser des automates, il est donc possible de considérer deux classes principales : la classe `Etat` et la classe `Transition`, le principe général étant que des instances de ces deux classes se « passent le relais » pour effectuer la lecture jusqu'à arriver à un état final qui la stoppe. Plus précisément :

- ✓ Il y a les états standards (classe `EtatStandard`) et au moins un état final (classe `EtatFinal`). Un état standard comporte des transitions sortantes alors qu'un état final n'en a pas car il termine la lecture. Ils ont tous un nom (début, signe...).
  Pour conserver l'ensemble des transitions sortantes d'un état standard et savoir immédiatement si la lecture d'un caractère est associée à une transition, chaque état standard e_s comporte un tableau de transitions `sortantes` de 128 cases (autant que de caractères possibles en se limitant aux caractères les plus courants). Pour un caractère `c` donné, `sortantes[c]`[4] référence la transition sortante correspondant à `c` ou vaut `null` si ce caractère n'est pas associé à une transition sortante. Dans ce dernier cas, une erreur est affichée et le programme est arrêté.



[4] On rappelle qu'un caractère est traité par java comme un nombre via son code ascii. Il est donc correct d'écrire `sortantes['+']` ; cela revient à écrire `sortantes[43]`, 43 étant le code ascii de '+'.

L'arrivée au bout (en fin) de la chaîne à lire peut aussi déclencher une transition (condition fin dans l'automate de l'exemple). Pour gérer simplement cette situation, un caractère de terminaison est ajouté à la fin de toute chaîne à reconnaître. Par convention, c'est le caractère de code ascii 0 : `(char)0` qui indice la première case du tableau de transitions[5]. Voir ci-dessus ce tableau pour l'état e_3 de l'exemple où la lecture de '.' ou de ',' renvoie à la transition t_5, celle d'un chiffre déclenche la transition t_1, l'arrivée en fin de chaîne déclenche t_3 et tous les autres caractères sont erronés.

- ✓ Les transitions sont aussi de deux sortes : les transitions simples (`TransSimple`) et les transitions de lecture (`TransLecture`) qui font avancer la lecture de la chaîne. Dans tous les cas, elles sont définies par la liste des caractères qui les déclenchent (`plage`) — représentée par une chaîne de caractères — et un état `cible`, standard ou final. Par exemple, la plage pour la transition de lecture t_1 est la chaîne `"0123456789"` et son état cible est e_3.
- ✓ Tous les états ont une méthode de signature `execute(chaîne à lire, curseur)` qui a pour paramètres la chaîne de caractères à lire et le curseur de lecture. La méthode `execute` d'un état final affiche simplement le succès de la lecture. Celle d'un état standard doit passer, si elle existe, le relais à la transition correspondant au caractère courant ou à l'arrivée en fin de chaîne — via la méthode `declenche`, voir ci-dessous. Si aucune transition n'est associée à la lecture, une erreur doit être affichée.
- ✓ Toutes les transitions ont une méthode `declenche(chaîne à lire, curseur)`. Dans tous les cas, cette méthode passe le relais à l'état cible — via leur méthode `execute`. Dans le cas d'une instance de `TransLecture`, elle fait en plus avancer la lecture, c'est-à-dire que la méthode `execute` de l'état cible est lancée avec un curseur incrémenté d'une unité.
  La lecture s'effectue donc en lançant la méthode `execute` de l'état de début. Par exemple : `debut.execute("-123,08"+(char)0,0)`. Cet état lance la méthode `declenche` de la transition t_4 via son tableau `sortantes`, qui, elle-même lance la méthode `execute` de son état cible e_2, et ainsi de suite jusqu'à atteindre l'état final e_6 qui arrête la lecture en affichant le succès.
- ✓ Les états standards sont dotés d'une méthode `ajoute(Transition t)` qui permet de remplir leur tableau `sortantes`. Après l'application de la méthode, la transition est référencée par les cases du tableau correspondant aux caractères de sa plage. Par exemple, `e3.ajoute(t5)` doit faire référencer `t5` par les cases du tableau `sortantes` de `e3` correspondant aux caractères `','` et `'.'` (plage `",."` de `t5`).

Questions

Les classes relatives aux transitions sont considérées comme déjà écrites. Dans ce cadre,

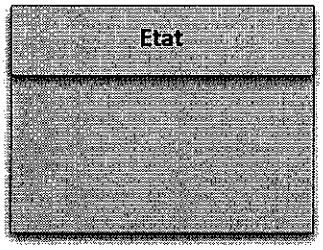
1. Complétez le diagramme de classes de l'annexe — que vous rendrez avec votre copie et qui comptera comme un intercalaire. Précisez les attributs des classes et les relations (utilisation et généralisation/spécialisation : *sorte de* [6]) qu'il y a entre elles.
2. Écrivez les lignes qui permettent de créer l'automate qui lit un nombre entier. Il faut créer les instances correspondant aux états e_1, e_2, e_3 et e_6, puis aux transitions t_1, t_4 et t_3 de l'exemple, et associer les transitions aux états via la méthode `ajoute`. Vous supposerez que vous disposez des constructeurs adéquats.
3. Écrivez complètement le code de la classe `Etat` sachant que la méthode `execute` est « `abstract` » (elle existe pour tous les états mais n'a pas toujours le même code).
4. Écrivez le code de la classe `EtatFinal`.
5. Écrivez le code de la classe `EtatStandard` sachant que, à sa création, une instance de cette classe n'est encore associée à aucune transition. Le code doit contenir la déclaration du tableau `sortantes`, les méthodes d'accès à son contenu, la méthode `ajoute` et la méthode `execute`.

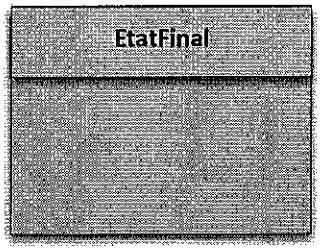
---

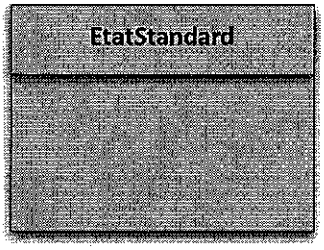
[5] Par exemple, pour lire la chaîne `"-123,08"`, c'est la chaîne `"-123,08"+(char)0` qui sera placée « dans » l'état début.

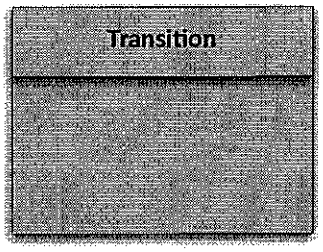
[6] Faites attention au sens des relations. Dessinez les relations d'utilisation en pointillés et les relations *sorte de* en traits pleins.

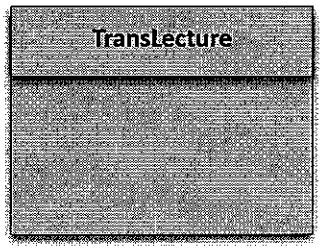
Annexe

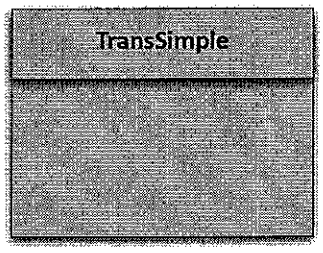












Anonymat :