Maîtrise d'Informatique **J.Bétréma**
Compilation (tronc commun) **Travaux dirigés**

Automates

1. Décrire en français les langages suivants ($A = \{a, b\}$) :
   - $((\epsilon|a)b)^*$
   - $a^*ba^*ba^*ba^*$
   - A^*aA^2
2. On utilise souvent, dans les expressions rationnelles, les notations abrégées suivantes :
   - L^+ pour désigner les mots formés d'un ou plusieurs mots, concaténés, du langage L ;
   - $L?$ pour indiquer que L est une partie *optionnelle* de l'expression.

   Donner les définitions formelles de ces raccourcis, qu'on pourra utiliser pour les exercices suivants.
3. Les identificateurs en ADA ressemblent à ceux du langage C, sauf que, pour éviter que des identificateurs distincts soient confondus par des humains, le caractère '_' (souligné, ou underscore pour certains) ne doit apparaître, ni doublé, ni en tête ou en fin d'un identificateur. Donner une expression rationnelle et un automate déterministe pour ces identificateurs.
4. Constantes numériques :
   (a) Donner une expression rationnelle et un automate déterministe pour les constantes numériques positives exprimées en virgule fixe (exemples : 421, 3.14) ; discuter selon qu'on admet ou non une constante comme "3." (quel peut être l'intérêt de cette dernière notation ?) ; considérer aussi le cas de ".5".
   (b) Mêmes questions en admettant des constantes numériques en virgule flottante, avec un exposant positif (exemple : 6.02E23).
   (c) Mêmes questions en tenant compte des signes éventuels.
5. Compression de tables : la table d'un automate déterministe peut être comprimée en stockant, pour chaque état q, le numéro d'un état q' tel que les transitions issues de q soient aussi voisines que possible de celles issues de q' ; il suffit ensuite de spécifier les transitions issues de q distinctes de celles issues de q'.

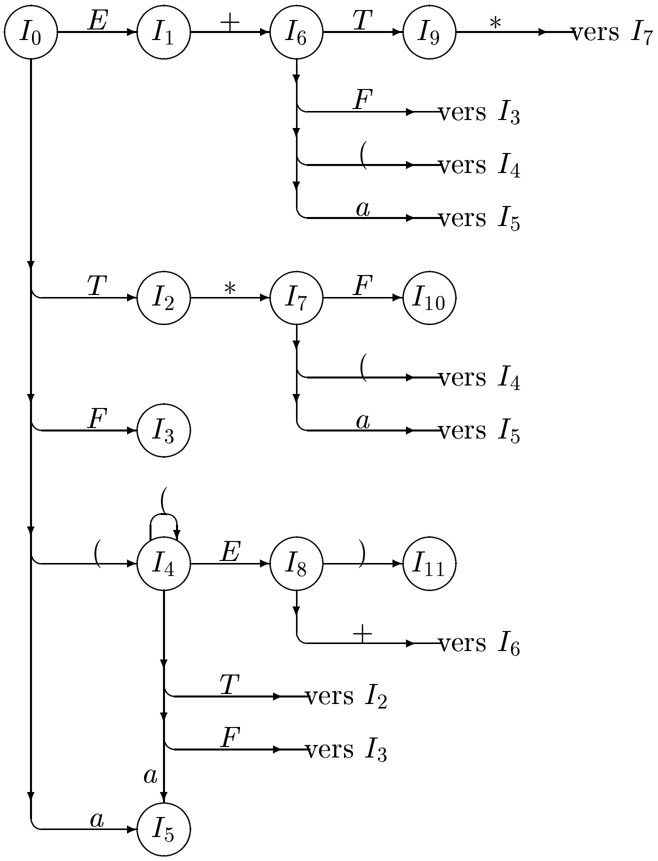
   Une réalisation possible est la suivante ; on utilise :

- deux tableaux, indexés par les états, *base* et *defaut*;
- deux tableaux, de taille variable, *suivant* et *controle*.

On numérote les lettres de l'alphabet utilisé à partir de 0, et la fonction de consultation de la table de l'automate s'écrit :

```
etat transition (etat q, int a) {
  int i = base [q] + a;
  if (controle [i] == q)
    return suivant [i];
  else
    return transition (defaut [q], a);
}
```

Remplir intelligemment les tables pour l'automate suivant, qui sert en analyse syntaxique ascendante (fig. 4.36 du dragon) :



Corrigé

1. Ce sont, respectivement, les langages formés des mots u tels que :
   - tout a est suivi d'un b; si l'on choisit la formulation : pas de facteur a^2 dans u, il faut ajouter $u \neq a$;
   - u comporte exactement trois occurrences de b;
   - quand on lit u de droite à gauche, la troisième lettre est un a.

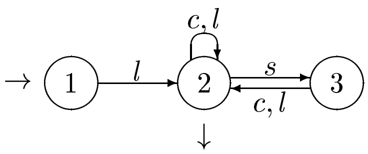
2. $L^+ = LL^* = L^*L$. Expliquer les problèmes posés par la réponse (incorrecte) $L^+ = L^* \backslash \epsilon$ (complémentaire et intersection ne font pas partie des opérations autorisées dans les expressions rationnelles; le fait que le complémentaire d'un langage rationnel – resp. l'intersection de deux langages rationnels – soit un langage rationnel est un théorème). Digressions sur des formules comme : $L^* = \epsilon | L^+ = \epsilon | LL^*$.

   $L? = \epsilon | L$.

3. Soit L l'ensemble des lettres, C celui des chiffres, et S le caractère _ (souligné); je sais, C, L, S sont aussi des lettres … La spécification demandée est :

$$L(S?(L|C))*$$

   Automate :



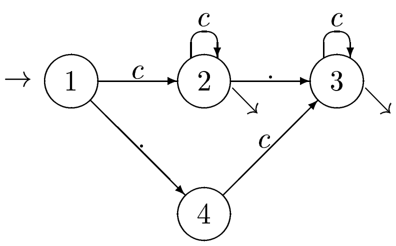
4. (a) Soit C l'ensemble des chiffres. $C^+(.C^+)? = C^+|C^+.C^+$ n'autorise pas les constantes qui débutent ou finissent par un point. L'expression

$$C^+|C^+.C^+|C^+.|.C^+$$

   peut se simplifier de plusieurs manières, par exemple :

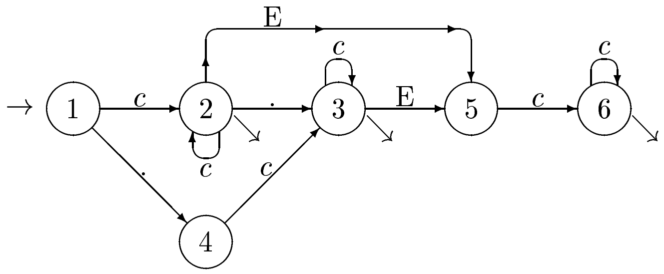
$$C^+|C^+.C^*|.C^+$$

   Automate :



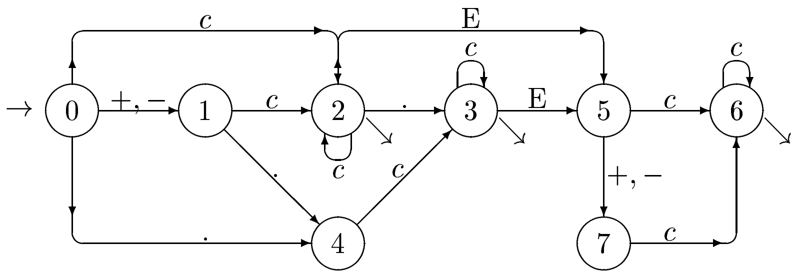
   (b)

$$(C^+|C^+.C^*|.C^+)(EC^+)?$$

Automate : 

(c) En notant $S = (+|-)$, l'expression peut s'écrire :

$$S?(C^+|C^+.C^*|.C^+)(ES?C^+)?$$

Automate : 

Remarquer l'identité, visible sur l'automate (états 5, 6 et 7) :

$$S?C^+ = (\epsilon|S)CC^* = (C|SC)C^*$$

5. Numérotation de l'alphabet :

0	1	2	3	4	5	6	7
E	T	F	$+$	$*$	(	)	a

On ajoute un état puits I_{12}.

- Etat I_0 : base[0] = defaut[0] = 0 (defaut[0] ne sera pas utilisé) ; et on enregistre toutes les transitions pour ce premier état :

	0	1	2	3	4	5	6	7
suivant	1	2	3	12	12	4	12	5
controle	0	0	0	0	0	0	0	0

- Etat I_1 : ses seules transitions communes avec I_0 concernent $*$ et) ; base[1] = 8, defaut[1] = 0, et on enregistre les transitions, sauf pour $*$ et) :

	8	9	10	11	12	13	14	15
suivant	12	12	12	6		12		12
controle	1	1	1	1		1		1

Les entrées 12 et 14, correspondant aux transitions $*$ et), sont laissées libres.

- Etat I_2 : identique à I_1, sauf pour les transitions $+, *$; base[2] = 13 (si, si!), defaut[2] = 1, et on enregistre seulement les transitions pour $+, *$, à partir de l'adresse 13+3=16!

	16	17
suivant	12	7
controle	2	2

- Etat I_3 : identique à I_2, sauf pour la transition $*$; base[3] = 8 (si, si!), defaut[3] = 2, et on enregistre seulement la transition pour $*$, à l'adresse 8+4=12, qui est libre!
- Etat I_4 : identique à I_0, sauf pour la transition E; base[4] = 14, defaut[4] = 0, et on enregistre la transitions pour E, à l'adresse 14, qui est libre.
- Et ainsi de suite; $I_5, I_{10}, I_{11}, I_{12}$ sont identiques à I_3, et n'occupent donc pas de place dans les tables *suivant* et *controle* (d'ailleurs on pourrait les confondre en un seul état dans l'automate; dans le contexte de l'analyse syntaxique ascendante, ils correspondent à des actions sémantiques distinctes).
  - I_6 est identique à I_0, sauf pour les transitions E, T;
  - I_7 est identique à I_6, sauf pour les transitions T, F;
  - I_8 est identique à I_1, sauf pour la transition);
  - I_9 est identique à I_2.

Tables finales :

	base	defaut
0	0	
1	8	0
2	13	1
3	8	2
4	14	0
5	0	3
6	18	0
7	19	6
8	16	1
9	0	2
10	0	3
11	0	3
12	0	3

	suivant	controle
0	1	0
1	2	0
2	3	0
3	12	0
4	12	0
5	4	0
6	12	0
7	5	0
8	12	1
9	12	1
10	12	1
11	6	1
12	12	3
13	12	1
14	8	4
15	12	1
16	12	2
17	7	2
18	12	6
19	9	6
20	12	7
21	10	7
22	11	8
23		-1
24		-1
25		-1
26		-1

Un étudiant intelligent a fait remarquer que les tables *suivant* et *controle* doivent être remplies jusqu'à la plus grande base, augmentée de la plus grande lettre de l'alphabet (ici $19 + 7 = 26$), avec une valeur “impossible” pour *controle*.

On peut, pour confirmation, faire tourner l'algorithme, par exemple pour calculer les transitions depuis I_9.