Université de Limoges | FACULTÉ DES SCIENCES ET TECHNIQUES

Licence 3e année

Compilation 2

Corrections mai 2012

Durée : 1h30 – Tous documents autorisés

Utilisation de Lex — (6 points)

1 – Contrôle d'un plan de vol pour véhicule spatial expérimental

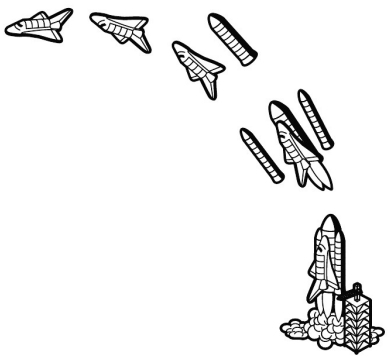
6pts

Une société d'exploration spatiale vous a contacté pour effectuer la vérification d'un plan de vol pour son prototype de navette réutilisable.

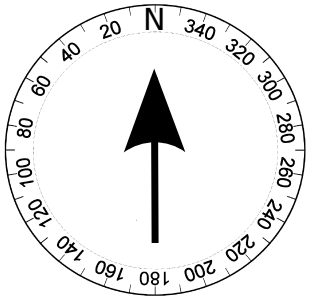
Plan de vol à analyser

```
# Vol avec booster
Direction 40
Acceleration 30
Attente 200
Direction 60
Attente 150
Direction 50
Acceleration 40
Attente 600
# Fin vol initial
Largage booster_double
Direction 50
Attente 200
Direction 80
Attente 100
Largage booster_principal
# fin vol booster
Attente 300
```

Description du vol



Direction de vol



Lors du contrôle du plan de vol, il faut vous assurer que :

- le contenu est une liste de commandes de l'ensemble suivant : `{Direction, Acceleration, Attente, Largage}` ;
- les lignes de commentaires commencent par le symbole « `#` » ;
- la **somme** de tous les paramètres de la commande « `Acceleration` » est inférieure à 100 : $\sum Acceleration(x) < 100$
- lors d'un changement de direction vers un nouveau cap, la différence entre le cap précédent et le nouveau cap est inférieure à 30° : $|cap_{ancien} - cap_{nouveau}| < 30$

Travail :
Écrivez le code d'un interprète au **format Lex** réalisant ce travail de contrôle et affichant un rapport sur le résultat des différents contrôles (succès ou échec).

```
%{ #include <stdio.h>
   #include <stdlib.h>
   #include <math.h>
   int acceleration = 0;
   int direction = -1;
%}
%start ACCELERATION DIRECTION
valeur [0-9]+
espaces [\t ]*
autres (Attente|Largage)
%%

Direction{espaces}  {  BEGIN DIRECTION; }
Acceleration{espaces} { BEGIN ACCELERATION; }
<DIRECTION>{valeur} {
   int conversion = atoi(yytext);
   if ((direction != -1) && (abs(conversion-direction)> 30))
   {
    printf("Changement de direction trop important !\n");
    exit(1);
   }
   direction = conversion;
            printf ("Direction %d\n", direction);
            BEGIN 0;
           }
<ACCELERATION>{valeur} {
   int conversion = atoi(yytext);
   acceleration += conversion;
   if (acceleration > 100)
   {
    printf("Acceleration superieure a 100 !\n");
    exit(1);
   }

            printf ("Acceleration %d\n", acceleration);
            BEGIN 0;
            }

{autres}{espaces}{valeur}{espaces}\n {
             printf("Commande ok : %s",yytext);
           }
#.*\n { printf("Commentaire %s", yytext);}
\n /* rien */
%%
```

Vous compilerez à l'aide des commandes suivantes :

```
pef@solaris:~$ flex space.l
pef@solaris:~$ gcc -o SPACE lex.yy.c -lm -lfl
pef@solaris:~$ ./SPACE
```

Utilisation de Lex & YACC — (6 points)

2 – 6pts Vous avez été recruté par une célèbre entreprise de jeux vidéo pour participer à la gestion des fichiers de sauvegarde d'une jeu de rôle médiéval, « Diablito IX ».

Vous allez gérer les personnages de l'équipe du joueur :

- chaque équipe dispose d'au plus 4 personnages ;
- chaque personnage est décrit par une classe `{Magicien, Guerrier, Voleur}` ;
- chaque personnage dispose de 4 caractéristiques : `{Santé, Magie, Dextérité, Constitution}` ;
- pour une classe donnée, un personnage peut majorer ses caractéristiques avec un « bonus de classe » :
  - `Magicien` : il peut répartir 5 points sur les caractéristiques `Magie` et `Dextérité` ;
  - `Guerrier` : il peut répartir 5 points sur les caractéristiques `Constitution` et `Santé` ;
  - `Voleur` : il peut répartir 5 points sur les caractéristiques `Santé` et `Dextérité` ;
- exemple de contenu :

```
Equipe
Magicien [ Santé : 10, Magie : 12, Dextérité : 13, Constitution : 9 ]
   Bonus [ Magie : 3, Dextérité : 2 ]
Guerrier [ Santé : 12, Magie : 5, Dextérité : 11, Constitution : 13 ]
   Bonus [ Constitution : 4, Santé : 1 ]
Guerrier [ Santé : 11, Magie : 6, Dextérité : 15, Constitution : 15 ]
   Bonus [ Constitution : 1, Santé : 4 ]
```

Questions :

a. Dans un premier temps, votre travail consiste à vérifier que le fichier contient une équipe « correcte » du point de vue syntaxique (utilisation de lexèmes connus, chaque personnage possède 4 caractéristiques et 2 bonus) :

i. Quel pourrait être le type le plus simple pour votre « `yylval` » ?
Le type « `int` ».

ii. Donnez la liste des terminaux que vous allez utiliser pour faire votre **analyse lexical** le plus simplement possible ;
La liste suivante : `EQUIPE CLASSE BONUS CARACTERISTIQUE VALEUR`

b. Dans un second temps, la société aimerait que vous contrôliez la présence, pour un personnage, de chacune des caractéristiques `{Santé, Magie, Dextérité, Constitution}` :

i. Pour simplifier votre travail devez vous faire des modifications et donner des contraintes sur le format du fichier ?
Pour effectuer le contrôle demandé, il faut pouvoir distinguer chacune des caractéristiques.

ii. Devez vous modifier le type associé à « yylval » ?
Deux possibilités :
   - ⋆ *Oui : il faut donner un type « `int` » pour la valeur numérique, et un type « `char *` » pour le nom de la caractéristique et le contrôle sera fait dans les actions sémantiques de la grammaire ;*
   - ⋆ *Non : on peut utiliser différents lexèmes pour chaque caracteristique et le contrôle sera fait dans la grammaire elle-même.*

iii. Donnez la liste des terminaux que vous allez utiliser pour faire votre **analyse lexical** ;
La liste privilégiant un contrôle par la grammaire : `EQUIPE CLASSE BONUS VALEUR SANTE MAGIE DEXTERITE CONSTITUTION`

c. Donnez la grammaire au format YACC permettant de faire l'analyse de « l'équipe » :
   - qui vérifiera que

- ⋆ chaque caractèristique possède une valeur < 20 (pour éviter les modifications des fichiers de sauvegarde par des joueurs indélicats) ;
- ⋆ la somme des points de bonus est égale à 5 ;

```
%{
 #include <stdio.h>
 #include "persos_diablito.h"
%}

%token CROCOUV CROCFERM VALEUR CLASSE BONUS EQUIPE SANTE DEXTERITE CONSTITUTION
MAGIE DEUXPTS VIRGULE
%start Equipe

%%

Equipe : EQUIPE ListePersos
       ;

ListePersos : Perso
   | ListePersos Perso
   ;

Perso : CLASSE Caracteristiques Bonus;

Caracteristiques : CROCOUV Sante Magie Dexterite Constitution CROCFERM;

Sante : SANTE DEUXPTS VALEUR VIRGULE { if ($3 >= 20) printf("Erreur caracteristique\n");
}
Magie : MAGIE DEUXPTS VALEUR VIRGULE { if ($3 >= 20) printf("Erreur caracteristique\n");
}
Dexterite : DEXTERITE DEUXPTS VALEUR VIRGULE { if ($3 >= 20) printf("Erreur
caracteristique\n"); }
Constitution : CONSTITUTION DEUXPTS VALEUR { if ($3 >= 20) printf("Erreur caracteristique\n");
}

Bonus : BONUS CROCOUV MAGIE DEUXPTS VALEUR VIRGULE DEXTERITE DEUXPTS VALEUR
CROCFERM
  { if (($5 + $9) > 5) printf("Erreur bonus\n"); }
   | BONUS CROCOUV CONSTITUTION DEUXPTS VALEUR VIRGULE SANTE DEUXPTS VALEUR
CROCFERM
    { if (($5 + $9) > 5) printf("Erreur bonus\n"); }
   | BONUS CROCOUV SANTE DEUXPTS VALEUR VIRGULE DEXTERITE DEUXPTS VALEUR CROCFERM
    { if (($5 + $9) > 5) printf("Erreur bonus\n"); }
   ;

%%
int yyerror(char *s) { printf("%s\n",s); }

void main()
{
   yyparse();
}
```

Le contenu du fichier Lex associé (non demandé dans l'examen) :

```
%{
#include <stdlib.h>
#include <string.h>
#include "persos_diablito.h"
%}
classe (Magicien|Guerrier|Voleur)
valeur [0-9]+
%%

Equipe    return EQUIPE;
Bonus    return BONUS;
{classe}  return CLASSE;
"["    return CROCOUV;
"]"    return CROCFERM;
Sante   return SANTE;
Magie   return MAGIE;
Dexterite  return DEXTERITE;
Constitution return CONSTITUTION;
":"    return DEUXPTS;
","    return VIRGULE;
{valeur} { yylval = atoi(yytext); return VALEUR;
   }
. /* rien */
\n /* rien */
```

◇ Est-ce que la gestion d'une erreur présente dans le fichier est possible et comment pourrait-elle être gérée dans votre analyseur ?
Ne donnez pas de code, mais discutez seulement comment cela pourrait être fait par rapport au format de fichier proposé.
On pourrez « gérer » des erreurs dans un personnage en relançant l'analyse syntaxique à partir du personnage suivant, c-à-d à partir du prochain lexème « CLASSE ».

XML, DTD & XSLT – (8 points)

3 – 8pts La même société que dans l'exercice 2, vous demande maintenant de leur proposer une solution à base d'XML pour une version massivement multijoueur de leur jeu :
Dans cette version, vous gérerez une collection de **nombreux** joueurs.

a. Donnez un DTD pour définir un fichier au format XML permettant d'exprimer cette **liste de joueurs** :
   - ◇ chaque joueur est identifié par un numéro unique ;
   - ◇ pour chaque joueur, son équipe est détaillée :
     - ⋆ liste de personnages ;
     - ⋆ chaque personnage avec sa classe, ses caractéristiques et ses bonus de classe.

```
<!ELEMENT ListeJoueurs (Joueur+)>
<!ELEMENT Joueur (Perso+)>
<!ATTLIST Joueur numero ID #REQUIRED>
<!ELEMENT Perso (Caracteristique,Caracteristique,Caracteristique,Caracteristique,Bonus)>
<!ATTLIST Perso
 Classe (Guerrier|Magicien|Voleur) #REQUIRED>
<!ELEMENT Caracteristique EMPTY>
<!ATTLIST Caracteristique
 Nom (Sante|Dexterite|Constitution|Magie) #REQUIRED
 Valeur CDATA #REQUIRED>
<!ELEMENT Bonus (Caracteristique,Caracteristique)>
```

Soit un exemple de déclaration (non demandé dans l'examen) :

```
<?xml version='1.0'?>
<!DOCTYPE ListeJoueurs SYSTEM "ListeJoueurs.dtd">
<ListeJoueurs>
<Joueur numero='J1'>
 <Perso Classe='Magicien'>
  <Caracteristique Nom='Sante' Valeur='10'/>
  <Caracteristique Nom='Dexterite' Valeur='13'/>
  <Caracteristique Nom='Constitution' Valeur='9'/>
  <Caracteristique Nom='Magie' Valeur='12'/>
  <Bonus>
   <Caracteristique Nom='Magie' Valeur='3'/>
   <Caracteristique Nom='Dexterite' Valeur='2'/>
  </Bonus>
 </Perso>
 <Perso Classe='Guerrier'>
  <Caracteristique Nom='Sante' Valeur='12'/>
  <Caracteristique Nom='Dexterite' Valeur='11'/>
  <Caracteristique Nom='Constitution' Valeur='13'/>
  <Caracteristique Nom='Magie' Valeur='5'/>
  <Bonus>
   <Caracteristique Nom='Magie' Valeur='4'/>
   <Caracteristique Nom='Dexterite' Valeur='1'/>
  </Bonus>
 </Perso>
</Joueur>
</ListeJoueurs>
```

b. Donnez un fichier au format XSLT permettant d'afficher sous forme de tableau HTML :
   - ◇ la liste des joueurs et pour chaque joueur son équipe :
     - ⋆ par personnage : ses caractéristiques et sa classe ;
     - ⋆ mais **sans indication** des « bonus de classe » ;

     Vous disposez pour améliorer votre affichage d'un serveur incorporant des icones de classe :
     - ⋆ `http://combat.net/icone/guerrier.png`
     - ⋆ `http://combat.net/icone/magicien.png`
     - ⋆ `http://combat.net/icone/voleur.png`

```
1  <xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="2.0">
2  <xsl:template match="/ListeJoueurs">
3  <html>
4  <head><title>Liste des Joueurs</title></head>
5  <body>
6  <table>
7  <xsl:for-each select="Joueur">
8  <xsl:if test="Perso[@Classe='Magicien']">
9   <tr><h1>
10  <th>Numéro</th>
11  <th><xsl:value-of select="@numero"/></th>
12  </h1>
13  </tr>
14  <xsl:for-each select="Perso">
15   <tr>
16   <td><h2><xsl:value-of select="@Classe"/></h2></td>
17   <td>
18   <xsl:choose>
19   <xsl:when test="@Classe='Magicien'"><IMG src='http://combat.net/magicien.png'>
20   </IMG></xsl:when>
21   <xsl:when test="@Classe='Guerrier'"><IMG src='http://combat.net/guerrier.png'>
22   </IMG></xsl:when>
23   <xsl:when test="@Classe='Magicien'"><IMG src='http://combat.net/voleur.png'>
24   </IMG></xsl:when>
25   </xsl:choose>
26   </td>
27   </tr>
28   <xsl:for-each select="Caracteristique">
29    <tr>
30    <td><xsl:value-of select="@Nom"/></td>
31    <td><xsl:value-of select="@Valeur"/></td>
32    </tr>
33   </xsl:for-each>
34   <tr>
35   <td><h3>Bonus</h3></td>
36   <td></td>
37   </tr>
38   <xsl:for-each select="Bonus/Caracteristique">
39    <tr>
40    <td><xsl:value-of select="@Nom"/></td>
41    <td><xsl:value-of select="@Valeur"/></td>
42    </tr>
43   </xsl:for-each>
44  </xsl:for-each>
45  </xsl:if>
46 </xsl:for-each>
47 </table>
48 Nombre total de joueurs <xsl:value-of select='count(Joueur)'/>
49 </body>
50 </html>
51 </xsl:template>
52 </xsl:stylesheet>
```

c. Ajoutez l'affichage du **nombre total** de joueurs.
L'affichage se fait à la ligne 48 de la question précédente.

d. Modifiez le fichier de la question (b) pour n'afficher que les joueurs dont l'équipe contient un « Magicien ».
Ce traitement est réalisé par les lignes 8 et 45 avec la balise `<xsl:if>`.