

## Contrôle continu 2 (CC-TP) SUJET Exemple - Durée 20 min

### Éléments de correction

## 1 Manipulations préliminaires

- Sauvegardez vos modifications faites dans le git étudiant, par exemple :

```
git commit -a -m "mes modifs"
```

- Récupérer le sujet de contrôle :

```
git pull
```

- Vous travaillez dans le répertoire CC-TPEx. Un Makefile, un main et une grammaire commentée est fournie :

```
grammar AnB2n;
```

```
prog : EOF {print("sujet de test!");}
```

```
WS : [ \t\r\n]+ -> skip ; // skip spaces, tabs, newlines
```

- Ouvrir le Makefile et changer JohnDoe en votre prénom suivi de votre nom (sans accent, sans espace, sans caractère spécial, tirets autorisés).

## 2 Exercice - grammaire avec ANTLR

L'objet de cet exercice est d'écrire un analyseur qui reconnaît le langage  $a^n b^{2n}$ , ( $n > 0$ ). Les autres caractères alphabétiques (a,b, ...z, A, ...Z) ainsi que les blancs et tabulation seront ignorés, les autres caractères feront une erreur de syntaxe.

1. Écrire un premier fichier de test `tests/ex0.txt`.
2. Éditer le `.g4` pour coder l'analyseur (lexical/syntaxique). Tester avec :

```
make  
python3 sujetEx.py tests/ex0.txt
```

**On rappelle qu'un fichier accepté par la grammaire ne cause aucun affichage sur la sortie standard, seules d'éventuelles erreurs lexicales ou syntaxiques sont affichées.**

3. Dans le répertoire `tests/` ajouter 5 tests pertinents pour cet analyseur (positifs, négatifs).
4. Pour déposer :

```
make clean  
make tar
```

vous fournit un tgz à déposer sur TOMUSS.

## Solution:

---

```
grammar AnB2n;

start: akbk EOF;

akbk:
    A akbk B B
    |
;

A: 'a' ;
B: 'b' ;
CHARS: [c-z,A-Z] -> skip ; //skip chars
WS : [ \t\r\n]+ -> skip ; // skip spaces, tabs, newlines
```

---