Université Toulouse III – Paul Sabatier
L2 Informatique

**Algorithmique**
*12 janvier 2013*
Semestre 3

# Projet — Le problème du voyageur de commerce

Antoine de Roquemaurel (Groupe 1.1)

## 1 Le projet

### 1.1 Compilation

- `make` ou `make build` Compile le projet
- `make clean` supprime les fichiers binaires (.o)

Une fois un make effectué le fichier executable est disponible en racine du projet, il se nomme `voyageurDeCommerce`.

### 1.2 Execution

Comme demandé dans le cahier des charges, le programme doit respecter une liste d'argument précis, exemples d'utilisation :

```
./voyageurDeCommerce -v -f resources/inputFiles/essai8.txt -bf
./voyageurDeCommerce -f resources/inputFiles/essai8.txt -bf -lsr 50
./voyageurDeCommerce -f resources/inputFiles/ulysse16.txt -lsnr 20 -lsr 50
./voyageurDeCommerce -v -f resources/inputFiles/ulysse16.txt -ga 15 0.8
```

Listing 1 – Exemple d'execution du programme

### 1.3 Organisation des fichiers

Afin d'avoir une meilleur clarté, le projet est organisé en plusieurs fichiers, qui sont répartis dans différents dossiers, ci-dessous l'utilité de chacun des dossiers.

**build** Ce dossier contient les fichiers binaires (.o) du projet, ceux-ci seront supprimés en utilisant la commande `make clean`

**doc** Ce dossier contient la documentation générée à l'aide de `doxygen` du projet. [1]
La documentation du projet est également disponible à l'adresse suivante :
▷ `http://documentation.joohoo.fr/L2/voyageurCommerce/`

**lib** Ce dossier contient les fichiers headers (.h)

**src** Ce dossier contient les sources du projet (.c)

**resources** Ici sont entreposés les fichiers de ressources pouvant être utiles au programmes

**tests** Les fichiers de tests.

1. Celle-ci est disponible en format HTML ou pdf.

# 2 Modifications apportés depuis la validation du 7 Janvier 2013

Les modifications depuis la validation :
– Correction bug des recherches locales.

# 3 Programmation avec preuve de `util_searchFirstOccurence`

## 3.1 Spécification

```
1  /* N > 0 */
2  searchFirstOccurence(T, R, N, p);
3  /* ((∀ I : 0 ≤ I < N → T[i] ≠ R)→ p = -1) ∨
4   *  ((∃ J : 0 ≤ J < N ∧ T[J] = R) → p = J)
5   */
```

## 3.2 Programmation

```
1  /* N > 0 */
2  p = 0;
3  /* INV = ∀ J : 0 ≤ J ≤ p → T[J] ≠ R */
4  while(p < N && T[p] != R) {
5    /* p < N ∧ T[p} ≠ R ∧ INV */
6    ++p;
7    /* INV */
8  }
9  /* p > N ∧ T[p] = R ∧ ∀J: 0 ≤ J < p → T[J] ≠ R */
10 if(p == N-1) {
11   p = -1;
12 }
13 /* (( ∀ I: 0 ≤ I < N → T[I] ≠ R) → p = -1) ∧
14  * ((∃ p : 0 ≤ < N ∧ T[p] = R) ∧ ∀ J : 0 ≤ J < p → T[J] ≠ R)
15  */
```

## 3.3 Preuve de programme

### 3.3.1 Initialisation

$$\begin{aligned} N > 0 \quad &\rightarrow \quad \texttt{pfp}(\text{"}p = 0\text{"}, \forall J : 0 \leq J < p \rightarrow T[J] \neq R) \\ N > 0 \quad &\rightarrow \quad \underbrace{\forall J : 0 \leq J < 0}_{\bot} \rightarrow T[J] \neq R) \end{aligned}$$

L'initialisation est donc correct.