

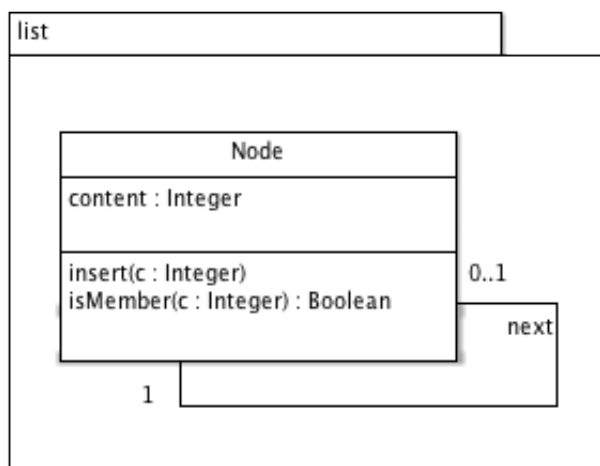
Nada Abdallah, Cédric Saule, Delphine Longuet
Prof. Burkhardt Wolff
Parc Orsay Université
4, rue Jacques Monod
Building H / Room 012

TD 6

Date : 09/03/2010

Exercice 1 (Test basé sur la spécification)

Engendrer un ensemble de cas de test puis des tests pour l'opération `insert` décrite dans le modèle suivant :



```
package list
```

```
context Node
```

```
inv wff : self.next.oclIsDefined()          --- we ignore this part
inv asc : self.next <> null implies content < next.content
```

```
context Node::nodes():Set(Node)
```

```
pre : true
```

```
post: if self = null then result = Set{}
      else result = self.next.nodes()->including(self)
```

```

endif

context Node::insert(i:Integer):OclVoid
pre : self <> null
post: self.nodes().content = (self.nodes@pre().content)->including(i)

endpackage

```

En particulier, on demande de :

1. Déplier les prédicats récurrents avec une profondeur 3.
2. Calculer la forme DNF et construire les cas de test.
3. Sélectionner des instances pour chaque cas de test.

Indications.

- Vous pouvez utiliser des notations mathématiques pour la dérivation (voir cours).
- Le prédicat récurrent représentant l'invariant prend la forme suivante :

$$\begin{aligned}
Inv(s) \equiv s = null \vee \\
s.next \langle \rangle null \Rightarrow s.content < s.next.content \wedge Inv(s.next)
\end{aligned}$$

- Vous pouvez supposer une profondeur de test de 3, correspondant à :

$$\begin{aligned}
H(s) = s = null \vee \\
s \langle \rangle null \wedge s.next = null \vee \\
s \langle \rangle null \wedge s.next \langle \rangle null \wedge s.next.next = null \vee \\
s \langle \rangle null \wedge s.next \langle \rangle null \wedge s.next.next \langle \rangle null \wedge s.next.next.next = null
\end{aligned}$$

- Commencez le calcul de la DNF avec $H(s) \wedge INV(s)$.
- On rappelle les lois booléennes
 $\neg\neg x = x, x \rightarrow y = \neg x \vee y, x \vee y = y \vee x, x \wedge y = y \wedge x, x \vee (y \wedge z) = (x \vee y) \wedge (x \vee z),$
 $x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z), \neg(x \wedge y) = \neg x \vee \neg y, \neg(x \vee y) = \neg x \wedge \neg y, x \wedge \neg x = False,$
 $x \vee \neg x = True$ etc.
- On rappelle la définition de `including` :
 $(S \rightarrow including(a)) \rightarrow includes(b) = (a=b) \text{ or } (S \rightarrow includes(b))$