

INFO0947 : TAD

Groupe XX : Simon Lorent, Corentin Jemine

Avril/Mai 2015

1 Introduction

Dans ce troisième projet nous avons défini deux structures de données, List et Array, pour lesquelles nous avons implémenté des fonctions de base. Ces fonctions sont des constructeurs, des observateurs ou des transformateurs. Nous nous pencherons sur une définition théorique de ces structures ainsi que leurs avantages et inconvénients respectifs.

2 Définition du type abstrait

2.1 Signature

Type :

Multi¹

Utilise :

Natural, Boolean, Element²

Opérations :

create_empty : \rightarrow Multi
is_empty : Multi \rightarrow Boolean
count : Multi \rightarrow Natural
occurrences : Element x Multi \rightarrow Natural
part_of : Element x Multi \rightarrow Boolean
equals : Multi x Multi \rightarrow Boolean
join : Multi x Multi \rightarrow Multi
add_to : Element x Multi \rightarrow Multi
remove_from : Element x Multi \rightarrow Multi

2.2 Sémantique

Préconditions :

$\forall m \in \text{Multi}, \forall e \in \text{Element} :$
remove_from(e, m) est défini ssi is_empty(m) = False

Axiomes :

Notations : #m désigne le nombre d'Elements dans m
m[i] désigne le i-ème Element de m

$\forall m, m' \in \text{Multi}, \forall e \in \text{Element} :$

is_empty(create_empty()) = True
is_empty(add_to(e, m)) = False
count(m) = #m

$\text{occurrences}(e, m) = \sum_{i=1}^{\#m} (m[i] == e)$

part_of(e, m) = ($\text{occurrences}(e, m) > 0$)

$\text{equals}(m, m') = (\#m == \#m') \ \&\& \ \prod_{i=1}^{\#m} ((\text{occurrences}(m[i], m) == \text{occurrences}(m[i], m'))$

$\text{occurrences}(e, \text{join}(m, m')) = \text{occurrences}(e, m) + \text{occurrences}(e, m')$

$\text{occurrences}(e, \text{add_to}(e, m)) = \text{occurrences}(e, m) + 1$

Si part_of(e, m) **alors** $\text{occurrences}(e, \text{remove_from}(e, m)) = \text{occurrences}(e, m) - 1$

Sinon $\text{equals}(m, \text{remove_from}(e, m))$

1. Multi désigne soit le type List, soit le type Array

2. Element désigne une type générique

2.3 Jusitification des axiomes