Programmation et Structures de Données 1995-96
TD 8
Itérateurs

Frédéric Vivien, bureau 306, e-mail: Frederic.Vivien

17 novembre 1995

1 Itérateurs `it_list` et `list_it`

1.1 Fonction de visualisation

Pour pouvoir visualiser plus facilement l'exécution des itérateurs, on définit une fonction `f` qui prend deux chaînes `x` et `s` comme arguments et qui renvoie la chaîne de caractères :

```
"f (" ^ x ^ " " ^ s ")"
```

1.2 Itérateurs pré-définis `it_list` et `list_it`

Comprendre le fonctionnement de ces deux itérateurs. On pourra s'aider de la fonction `f` précédemment définie…

1.3 `append` et `reverse`

Écrire les fonctions `append` et `reverse` d'abord à l'aide de l'itérateur `it_list` puis à l'aide de l'itérateur `list_it`.

1.4 Nouveaux itérateurs `it_list` et `list_it`

Écrire les fonctions `it_list` et `list_it`. Tester le résultat grâce à `f`.

1.5 `it_list` en fonction de `list_it`

Écrire l'itérateur `it_list` en fonction de l'itérateur `list_it` et de la fonction `reverse`.
Écrire l'itérateur `list_it` en fonction de l'itérateur `it_list` et de la fonction `reverse`.

1.6 `it_list` terminale récursive

Écrire une version terminale récursive de `it_list`.

2 Homomorphisme sur arbre

2.1 Définir un type *arbre binaire*

cf. titre.

2.2 Itérateur sur arbre

Écrire un homomorphisme `btree_hom` sur les arbres binaires.

2.3 Utilisations simples de `btree_hom`

Écrire au moyen de `btree_hom` des fonctions calculant la hauteur d'un arbre, le nombre de sommets d'un arbre, l'arbre miroir, et recherchant une clé dans un arbre.

2.4 Itérateurs `btree_it` et `it_btree`

Implémenter des itérateurs `btree_it` et `it_btree` inspirés de `it_list` et `list_it`.
Utiliser ces itérateurs pour lister suivant un parcours en largeur (infixe gauche) les clés de l'arbre.