UM 2 - IUT Montpellier **Programmation Fonctionnelle** 2ème année
Département Informatique

Exercices 6

Expressions arithmétiques

1) Lecture et traitement des caractères

Les chaînes sont délimitées par des ". Le prédicat `(string? x)` retourne `#t` si `x` est une chaîne de caractères. Les caractères sont différents des chaînes. Le prédicat `(char? x)` retourne `#t` si `x` est un caractère. Le caractère `a` est noté par `#\a`.

Q1) Essayez les cas suivants :

```
(define st "hello")
(define c #\F)
(define p (substring st 0 1))
st
c
p
(string? st)
(string? c)
(char? c)
(char? p)
(string? p)
```

2) Arbres syntaxiques

Un arbre syntaxique binaire non vide peut être un `constant numérique` ou un arbre construit à partir de trois éléments : `opérateur sous-arbre_gauche sous-arbre_droit.` Un constant est représenté par lui-même et pour la représentation d'un arbre composé on choisie une paire qui contient : l'opérateur en première position et une paire des deux sous-arbres en deuxième position. Naturellement, les sous-arbres peuvent être des arbres binaires ou des constants.

On veut construire l'arbre syntaxique d'une expression binaire qui est donnée par sa notation préfixée. Les éléments dans cette notation sont séparés par des blancs. Seuls les constants entiers et les quatre opérateurs (+,-,*,/) sont autorisés. S'il y a un élément illégal, on attend un message d'erreur.

Q2) Ecrire la fonction `(unarbre)` qui lit **une expression préfixée** (élément par élément) du clavier et qui retourne l'arbre binaire correspondant. Un exemple de l'entrée :

```
#\+
2
3
```

ce qui correspond à l'expression préfixée `+ 2 3` qui doit impliquer la construction de l'objet `(#\+ . (2 . 3))`

Q3) Écrire la fonction `(evaluation A)` qui retourne la valeur évaluée de l'arbre binaire `A`.

Q4) Soit `A` un arbre binaire défini comme précédemment. Écrivez une fonction `(postfixe A)` **qui affiche l'expression post-fixée** correspondante à l'arbre. Indication : pour afficher une valeur (numérique ou caractère), on peut utiliser la fonction `(display …)`

Q5) On souhaite **construire une chaîne de caractères (retourner un objet)** par une fonction `(chaine_postfixe A.` On peut utiliser les fonctions :

`(string-append str1 str2)`
`(string-append "brou" "haha")` retourne `"brouhaha"`
`(number->string x)` qui construit la chaîne représentant la valeur numérique `x.`
`(number->string 31)` donne "31"
Pour transformer un caractère `c` en une chaîne, on a le constructeur `(string c)`
`(string #\a)` donne `"a"`

Q6) Soit `A` un arbre binaire et `op` un opérateur défini comme précédemment. Écrivez une fonction `(compter op A)` qui retourne le nombre de fois l'opérateur est utilisé dans l'arbre.

Q7) Une expression symétrique est donnée avec les parenthèses. Construire l'arbre à partir du clavier.
Exemple :
#\(#\(#\(7 #* 4 #\) #\+ #\(33 #\- 8 #\) #\) #* 10 #\)

3) Fonctions comme arguments, compositions

Le nom d'une fonction peut être l'argument d'une autre. Par exemple :
`(define (appliquer f x) (f x))`
puis appeler avec :
`(appliquer abs -3)` qui donne `3`

Q8) Proposez une fonction `(cascade f g x)` qui applique la fonction `g` sur `x` puis la fonction `f` sur le résultat (qui retourne `f(g(x))`).

Testez avec `(cascade abs sin (* 1.5 pi))`

Q9) Écrivez une fonction `(maxlist l)` qui retourne la valeur maximale de la liste l qui ne contient que des valeurs numériques.

Q10) `(map p l)` est une fonction qui exécute la fonction `p` sur chaque élément de la liste `l`. Ecrivez une fonction `(manip f g l)` qui exécute la fonction `f` sur la liste obtenue par l'application de `g` sur les éléments de `l`.

En utilisant cette fonction, calculez la valeur maximale parmi les valeurs de `sin(x)` où `x` est un élément de la liste `(1 2 3 4 5 6 7 8 9)`