Programmation et Structures de Données 1995-96
TD 9
Les pointeurs en Pascal

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

28 novembre 1995

1 Pascal

Le compilateur Pascal s'appelle `pc` et s'utilise comme le compilateur C: `pc -o exécutable source.p`

2 Liste chaînée

On définit, une fois n'est pas coutume, un type liste chaînée. Cette fois-ci nous utilisons des pointeurs:

```
type list = ^ cell;
     cell = record
              car : integer;
              cdr : list
            end;
```

Implanter les fonctions classiques suivantes: `empty, car, cdr, is_empty, print_list`.

Voici une tentative d'implantation de la fonction `cons`:

```
function cons (x:integer; L:list):list
var c:cell
begin
  c.car := x;
  c.cdr := L;
  cons  := ^c
end
```

Expliquer pourquoi cette implantation est incorrecte. Avec un peu de chance, l'appel suivant ne devrait pas afficher ce que l'on attendrait de lui: `print_list(cons(1, cons(2, cons(3, empty()))))`

Écrire une fonction `cons` correcte.

3 Insertion dans une liste chaînée triée

On veut une fonction `insert(x, l)` qui insère l'élément `x` à la bonne place dans la liste triée `l` et qui renvoie la nouvelle liste triée.

Réaliser:

- une version récursive "à la caml"
- une version itérative: avancer dans la liste jusqu'au plus grand élément de `l` inférieur à `x`, générer une nouvelle cellule contenant `x`, insérer cette nouvelle cellule. Voir aussi la figure 1

Vérifier la correction des fonctions à l'aide de `print_list`...

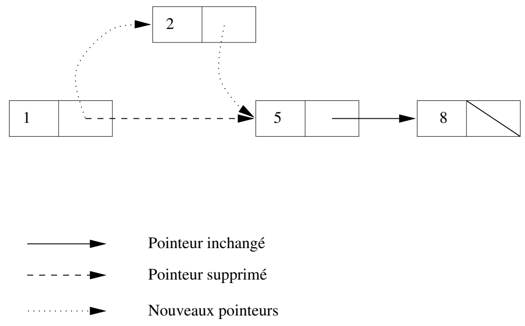
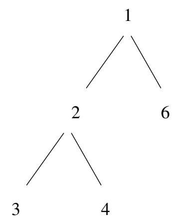


FIG. 1 - *Insertion de "2" dans la liste triée "1 5 8"*

4 Arbre binaire

On suppose que l'on code un arbre binaire d'entier par une chaîne de caractère de la forme: "("^ clé ^ fils_gauche ^ fils_droit ^")" Exemple: "(1 (2 3 4) 6)" code l'arbre de la figure 4.



On définit le type arbre suivant:

```
type tree = ^ node;
     node = record
              cle : int;
              fg  : tree;
              fd  : tree
            end;
```

Définir pour cette structure de données les fonctions: `empty, left, right, is_empty, print_tree`.

On veut définir une fonction `make_tree` qui prend une représentation par chaîne de caractère et qui renvoie l'arbre correspondant.