UNIVERSITÉ LOUIS PASTEUR STRASBOURG

Département d'Informatique
LICENCE 2ème année
Mercredi 10 septembre 2008

Corrigé du contrôle terminal de Programmation Fonctionnelle

I Récursivité

1.
```
let rec inferieur u v = match(u,v) with
    (tu::ru,tv::rv) -> (tu<tv) or ((tu=tv) & (inferieur ru rv))
  | ([],tv::rv)     -> true
  | _               -> false;;
```
2.
```
let rec conjugue u i = match(u) with
    t::r -> if i=1 then u else conjugue (r@[t]) (i-1)
  | _    -> failwith "erreur";;
```
3.
```
let rec lyndon u =
  let rec lyndon_aux u i =
      (i=1) or ((inferieur u (conjugue u i)) & (lyndon_aux u (i-1)))
  in lyndon_aux u (List.length u);;
```

II Tri par insertion

1.
```
let rec insere_mot u lst = match(lst) with
  v::l -> if u=v then lst
                 else if inferieur u v then u::lst
                                      else v::(insere_mot u l)
| []   -> [u];;

let rec insere_lst lst1 lst2 = match(lst1) with
  u::l -> insere_lst l (insere_mot u lst2)
| []   -> lst2;;
```
2.
```
let rec fusion_mot mot = function
  [] -> []
| t::q -> if inferieur mot t then (mot@t)::(fusion_mot mot q)
     else fusion_mot mot q;;

let rec fusion_lst lst1 lst2 =
```

```
match lst1 with
    [] -> []
  | mot::q -> insere_lst (fusion_lst q lst2) (fusion_mot mot lst2);;
```

3.
```
let rec genere n =
  let rec genere_aux n m =
      if n=1 then fusion_lst [[0];[1]] (genere m)
             else insere_lst (fusion_lst (genere n) (genere m))
                  (genere_aux (n-1) (m+1))
  in if n=1 then [[0];[1]]
            else genere_aux (n-1) 1;;
```

III Lambda-calcul

1. $\lambda f.\lambda g.\lambda h.((f\ h)\ (g\ h))_{\curlywedge}\lambda h.\lambda i.h\ \lambda h.h$
$\xrightarrow[\beta]{} \lambda g.\lambda h.((\lambda h.\lambda i.h\ h)\ (g\ h))_{\curlywedge}\lambda h.h$
$\xrightarrow[\beta]{} \lambda h.((\lambda h.\lambda i.h_{\curlywedge}h)\ (\lambda h.h\ h))$
$\xrightarrow[\beta]{} \lambda h.(\lambda i.h_{\curlywedge}(\lambda h.h\ h))$
$\xrightarrow[\beta]{} \lambda h.h$

2. $\lambda f.\lambda g.\lambda h.((f\ h)\ (g\ h))\ (\lambda h.\lambda i.h\ h)\ \lambda h.h$
$\xrightarrow[\alpha]{} \lambda f.\lambda g.\lambda j.((f\ j)\ (g\ j))_{\curlywedge}(\lambda h.\lambda i.h\ h)\ \lambda h.h$
$\xrightarrow[\beta]{} \lambda g.\lambda j.(((\lambda h.\lambda i.h\ h)\ j)\ (g\ j))_{\curlywedge}\lambda h.h$
$\xrightarrow[\beta]{} \lambda j.(((\lambda h.\lambda i.h_{\curlywedge}h)\ j)\ (\lambda h.h\ j))$
$\xrightarrow[\beta]{} \lambda j.((\lambda i.h_{\curlywedge}j)\ (\lambda h.h\ j))$
$\xrightarrow[\beta]{} \lambda j.(h\ (\lambda h.h_{\curlywedge}j))$
$\xrightarrow[\beta]{} \lambda j.(h\ j)$

3. $\lambda f.\lambda g.\lambda h.((f\ h)\ (g\ h))_{\curlywedge}(\lambda h.\lambda i.h\ i)\ \lambda h.h$
$\xrightarrow[\beta]{} \lambda g.\lambda h.(((\lambda h.\lambda i.h\ i)\ h)\ (g\ h))_{\curlywedge}\lambda h.h$
$\xrightarrow[\beta]{} \lambda h.(((\lambda h.\lambda i.h\ i)\ h)\ (\lambda h.h\ h))$
$\xrightarrow[\alpha]{} \lambda h.(((\lambda h.\lambda j.h_{\curlywedge}i)\ h)\ (\lambda h.h\ h))$
$\xrightarrow[\beta]{} \lambda h.((\lambda j.i_{\curlywedge}h)\ (\lambda h.h\ h))$
$\xrightarrow[\beta]{} \lambda h.(i\ (\lambda h.h_{\curlywedge}h))$
$\xrightarrow[\beta]{} \lambda h.(i\ h)$