Corrigé de l'examen de programmation avancée

ENSIIE, semestre 2

mercredi 28 mars 2011

Exercice 1 : Makefile (4 points)

Pour C :

```
entrees_sorties.o: entrees_sorties.h
donnees.o: donnees.h
lecture.o: entrees_sorties.h donnees.h lecture.h
traitement.o: donnees.h traitement.h
main.o: lecture.h traitement.h

prog: entrees_sorties.o donnees.o lecture.o traitement.o main.o
	gcc -o $@ @^
```

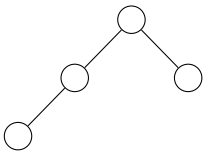
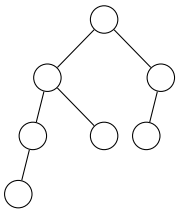
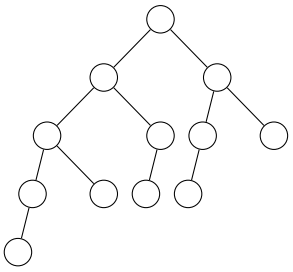
Pour OCaml

```
.SUFFIXES: .ml .mli .cmo .cmi

.ml.cmo:
	ocamlc -c $<
.mli.cmi:
	ocamlc -c $<

entrees_sorties.cmo: entrees_sorties.cmi
donnees.cmo: donnees.cmi
lecture.cmo: entrees_sorties.cmi donnees.cmi lecture.cmi
traitement.cmo: donnees.cmi traitement.cmi
main.cmo: lecture.cmi traitement.cmi

prog: entrees_sorties.cmo donnees.cmo lecture.cmo traitement.cmo main.cmo
	ocamlc -o $@ @^
```

Exercice 2 : Modularité (3 points)

1. Abstraction
2. Séparation des tâches
3. Encapsulation
4. Réutilisation
5. Maintenabilité
6. Création d'un espace des noms

Exercice 3 : Arbres de Fibonacci (9 points)

1. A_2 :  A_3 :  A_4 :  A_5 : 
2. En C :

```
int hfibo(abr a) {
  int hg, hd;
  if (a == NULL) return 0;
  if (a->fg == NULL && a->fd == NULL) return 1;
  hg = hfibo(a->fg);
  hd = hfibo(a->fd);
  if (hg == -1 || hd == -1) return(-1);
  if (hg != hd + 1) return(-1);
  return(hg+1);
}
```

   En OCaml :

```
let rec hfibo a =
  match a with
  | Empty -> 0
  | Node(Empty, _, Empty) -> 1
  | Node(fg, _, fd) ->
      let hg = hfibo fg in
      let hd = hfibo fd in
      if hg = -1 || hd = -1 then -1
      else if hg <> hd + 1 then -1
      else hg + 1
```

3. Si on note c_n la complexité en fonction du nombre de noeuds, on a la relation de récurrence $c_n = 1 + c_{n_1} + c_{n_2}$ avec $n = n_1 + n_2$ (temps constant plus les deux appels récursifs) et $c_1 = 1$. On a donc $c_n = n$.
4. Hauteur : $h_0 = 0$, $h_1 = 1$, $h_2 = 2$, $\ldots h_n = n$.

   Démonstration par récurrence sur n.

   Noeuds : $N_0 = 0$, $N_1 = 1$,$\ldots N_n = N_{n-1} + N_{n-2} + 1$.

   La démonstration n'est pas demandée, elle s'appuie sur le fait que N_n reprend le schéma de Fibonacci auquel il faut ajouter les $+1$. Il y a autant de $+1$ que « d'appels » à Fibonacci, or le nombre d'appels à la fonction Fibonacci fait intervenir la fonction de Fibonacci elle-même.

   On a donc $N_n = F_{n+2} - 1$.
5. On montre par induction que si un arbre binaire est un arbre de Fibonacci alors il est équilibré.

   Pour l'arbre vide, c'est trivial.

Supposons qu'on a un arbre a formé d'un noeud avec deux sous-arbres fg et fd. Par hypothèse d'induction on suppose que si fg est un arbre de Fibonacci alors il est équilibré et si fd est un arbre de Fibonacci alors il est équilibré.

Supposons que a est un arbre de Fibonacci. Il y a deux cas :

- $a = A_1$, dans ce cas il est bien équilibré ;
- $a = A_n$ pour $n > 1$. Dans ce cas, fg et fd sont les arbres de Fibonacci A_{n-1} et A_{n-2}. Par hypothèse d'induction ils sont équilibrés, donc pour tous leurs noeuds les hauteurs des sous-arbres gauche et droit ne diffèrent au maximum que de 1. Reste le cas du noeud à la racine. On utilise le résultat de la question précédente qui nous dit que leur hauteur est respectivement $n-1$ et $n-2$. Donc, en tout noeud, les hauteurs des sous-arbres gauche et droit ne diffèrent au maximum que de 1, l'arbre est bien équilibré.

NB : la démonstration est plus élégante par récurrence sur n l'indice de l'arbre de Fibonacci, mais il était explicitement demandé un preuve par induction.

6. Par définition, les arbres de Fibonacci sont les plus petits AVL possibles : pour chaque noeud on maximise la différence de hauteur entre les sous-arbres.

   Pour une hauteur h, la taille minimum est celle de l' arbre de Fibonacci de hauteur h soit $F_{h+2} - 1$. La taille maximum est celle l'arbre complet de hauteur h soit $2^h - 1$.

Exercice 4 : Fonction de hachage (4 points)

1. f ne peut être que 0 ou $0,5$. Par conséquent $h_{mult}(k, m)$ ne peut être que 0 ou $\lfloor \frac{m}{2} \rfloor$. $A = 0,5$ n'est donc pas un bon choix car seules deux cases de la table de hachage seront utilisées.
2. f peut prendre comme valeur $\frac{i}{q}$ pour i variant de 0 à $q-1$. Par conséquent $h_{mult}(k, m)$ ne peut être que $\lfloor \frac{im}{q} \rfloor$ pour i variant de 0 à $q-1$. Au maximum q cases de la table seront utilisées, ce qui est problématique si $q \ll m$. Il n'est donc pas judicieux de choisir un rationnel pour A.