Travaux Pratiques d'informatique

Algorithmique

Les arbres binaires.

On utilise les types:

```
typedef struct noeud{
  int valeur;
  struct noeud *fg;
  struct noeud *fd;
}Noeud,*Arbre;
```

1. **Un arbre à la main**

   (a) Écrire une fonction `Arbre allouеNoeud(int val)` qui renvoie l'adresse d'un `Noeud` étiquetté avec `val` et avec les champs `fg` et `fd` contenant `NULL`.

   (b) Utiliser directement cette fonction pour construire un arbre de quelques noeuds:

   ```
   a=alloueNoeud(1);
   a->fg=alloueNoeud(2);
   a->fg->fd=(alloueNoeud(42);
   ...
   /*ou*/
   a=alloueNoeud(5);
   a->fg=alloueNoeud(1);
   a->fg->fg=alloueNoeud(11);
   a->fg->fd=  alloueNoeud(2;
   a->fd=  alloueNoeud(21);
   a->fd->fd=          alloueNoeud(42);
   a->fd->fd->fd=  alloueNoeud(15);
   ```

   (*beurk, juste pour avoir un arbre*)

2. **Fonctions de mesure d'arbre**
   Écrire et tester les fonctions vu en td:

   (a) hauteur

   (b) nombre de nœuds

   (c) nombre de nœuds internes

   (d) nombre de feuilles

   (e) ...

3. **Constuction d'arbres étiquettés par des positifs**
   Les fichiers `arbre1, arbre2, arbre3, arbre4` contiennent les suites décrivant 4 arbres, obtenues par parcours en profondeur préfixe: une valeur positive correspond un nœud, 0 correspond à l'arbre vide.

   (a) Reconstruire ces arbres et indiquez les mesures de chacun de ces arbres.

   (b) Écrire et tester sur ces arbres une fonction permettent de déterminer si un arbre est strictement binaire (chaque noeud a 0 ou bien 2 fils).