Contrôle continu 3
Pointeurs - Récursivité – Listes – Piles – Files– Arbres
(3 pages)

Questions de cours (5 points)

- Peut-on créer un arbre binaire dont la valeur du sommet est une liste?

Oui la valeur d'un sommet d'un arbre binaire est un objet quelqconque.

- Doit-t-on utiliser une file pour parcourir un arbre binaire en ordre infixe ? Si oui, pourquoi ? Si non, quelle structure de données est utilisée ?

Non, on utilise une file pour le parcours hiérarchique. Pour le parcours infixe, la procédure s'écrit, en général, récursivement et donc utilise implicitement une pile.

- Quel sont les parcours qui existent pour les arbres binaires et pour les arbres planaires ?

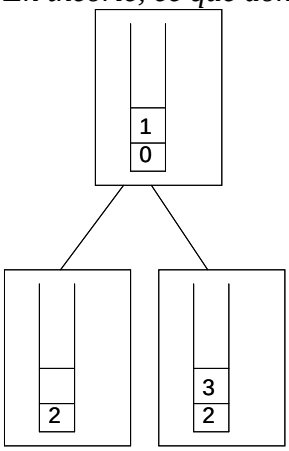
Les parcours préfixe, postfixe et hiérarchique.

- On considère la suite d'opérations suivante :

```
var A : arbreBinaire de pile d'entier;
var s :pile d'entier;
creerPile(s) ;
empiler(s,0);
empiler (s,1);
A=creerArbreBinaire(s) ;
detruirePile(s))
 creerPile(s) ;
empiler(s,2) ;
ajouterFilsGauche(A,s)
empiler(s,3) ;
ajouterFilsDroit(A,s)
```

Dessinez la structure de donnée ainsi obtenue.

En théorie, ce que donne cette suite d'instruction est la structure ci-dessous.



Y-a-t-il des problèmes liés à l'implémentation de la pile ? en implémentation statique ?

Non car la copie de la valeur est une copie d'une structure ne contenant pas de pointeur.

en implémentation dynamique ? On justifiera les réponses.

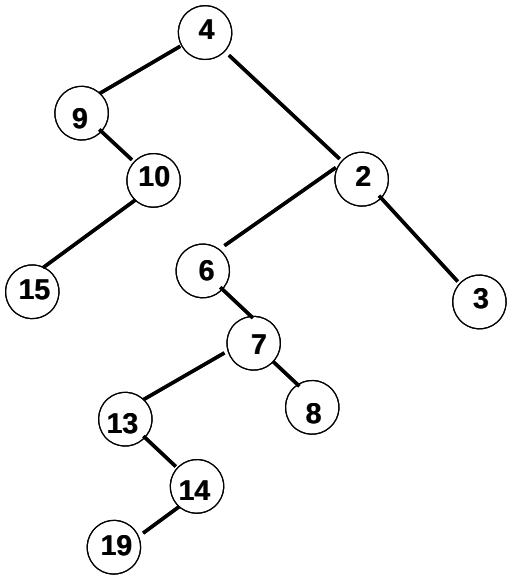
Oui, car la copie de la valeur va copier seulement les pointeurs de la liste SC. Donc quand on exécute, la primitive detruirePile, tous les éléments vont disparaître et de fait la structure stockée dans l'arbre va pointer vers des pans de mémoires non attribué. Pour éviter ce problème il faut créer une fonction d'affectation propre à ces piles (non traité en cours).

fonction op ::=(val P :pile d'objet) : pile d'objet ;

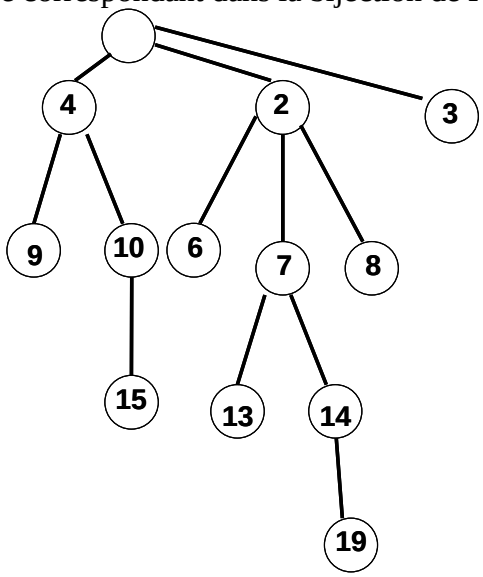
```
var F :pile d'objet ;
var R :pile d'objet ;
début
    creerPile(F) ;
    creerPile(R) ;
    tantque !pileVide(P) faire
            empiler(F,valeur(P)) ;
            depiler(P) ;
    fintantque
    tantque !pileVide(F) faire
            empiler(R,valeur(F)) ;
            depiler(F) ;
    fintantque
    detruirePile(F) ;
    retourner(R)
fin
```

Exercice 2 (5 points)

Soit l'arbre binaire suivant



1. Donnez sa hauteur
   La hauteur est 6.
2. Donnez la liste des sommets dans l'ordre postfixe (ou suffixe)
   15,10,9,19,14,13,8,7,6,3,2,4
3. Donnez la liste des sommets dans l'ordre hiérarchique
   4,9,2,10,6,3,15,7,13,8,14,19
4. Donnez l'arbre planaire correspondant dans la bijection de Knuth



5. Pour l'arbre planaire ainsi obtenu, donner la liste des entiers en ordre préfixe.
   4,9,10,15,2,6,7,13,14,19,8,3

Exercice 2 (10 points)
Ecrire une fonction qui renvoie la liste de tous les sommets internes d'un arbre binaire en ordre infixe :
1 – En utilisant les primitives du type arbreBinaire et listeSC

```
fonction listeFeuille(val A :sommet ;ref L : listeSC de sommet) : vide ;
  début
    si !estFeuille(A) alors
     insererEnTete(L,A) ;
     si FilsGauche(A) !=NIL alors
       listeFeuille(FilsGauche(A),L)
     finsi
      si Filsdroit(A) !=NIL alors
       listeFeuille(Filsdroit(A),L)
     finsi
   finsi
 fin
```

2 – En implémentant directement en allocation dynamique pour le type Arbre Binaire et le type listeSC.

```
fonction listeFeuille (val A : sommet ;ref L :listeSC de sommet) :vide ;
  var c :curseur ;
  début
    si A^.gauche !=NIL ou A^.droit=NIL alors
      new( c) ;
      c^.info=A ;
```

```
        c^.suivant=L.premier;
        L.premier=c;
        si A^.gauche !=NIL alors
         listeFeuille(A^.gauche,L)
       finsi
        si A^.droit !=NIL alors
         listeFeuille(A^.droit,L)
       finsi
    finsi
  fin
```

Liste Simplement Chainées

fonction valeur(val L:liste d'objet):objet;
fonction debutListe(ref L:liste d'objet):vide;
fonction suivant(ref L:liste d'objet):vide;
fonction listeVide(val L:liste d'objet): booleen;
fonction créerListe(ref L:liste d'objet):vide;
fonction insérerAprès(ref L:liste d'objet; val x:objet;):vide;
fonction insérerEnTete(ref L:liste d'objet ;val x:objet):vide;
fonction supprimerAprès(ref L:liste d'objet):vide;
fonction supprimerEnTete(ref L:liste d'objet):vide;
fonction detruireListe(ref L:lisetd'objet):vide;</pre>

Liste Doublement Chainées (ajout de)

fonction finListe(ref L:liste d'objet):vide;
fonction précédent(ref L::liste d'objet): vide;

Fonctions sur les listes

fonction estFinListe(val L:liste d'objet):booléen;
fonction appartient(ref L:liste d'objet; ref x:objet): booléen ;

Piles

fonction valeur(ref P:pile de objet):objet;
fonction fileVide(ref P:pile de objet):booléen;
fonction créerPile(P:pile de objet);
fonction empiler(ref P:pile de objet;val x:objet):vide;
fonction dépiler(ref P:pile de objet):vide;
fonction detruirePile(ref P:pile de objet):vide;

Files

fonction valeur(ref F:file de objet):objet;
fonction fileVide(ref F:file de objet):booléen;
fonction créerFile(F:file de objet);vide;
fonction enfiler(ref F:file de objet;val x:objet):vide;
fonction défiler (ref F:file de objet):vide;
fonction detruireFile(ref F:file de objet):vide;

Arbres binaires

fonction getValeur(val S:sommet):objet;
fonction filsGauche(val S:sommet):sommet;
fonction filsDroit(val S:sommet):sommet;
fonction pere(val S:sommet):sommet;
fonction setValeur(ref S:sommet;val x:objet):vide;
fonction ajouterFilsGauche(ref S:sommet,val x:objet):vide;
fonction ajouterFilsDroit(ref S:sommet,x:objet):vide;
fonction supprimerFilsGauche(ref S:sommet):vide;
fonction supprimerFilsDroit(ref S:sommet):vide;
fonction detruireSommet(ref S:sommet):vide;
fonction créerArbreBinaire(val Racine:objet):sommet;

Arbres planaires

fonction valeur(val S:sommetArbrePlanaire):objet;
fonction premierFils(val S:sommetArbrePlanaire):sommetArbrePlanaire;
fonction frere(val S:sommetArbrePlanaire):sommetArbrePlanaire;
fonction pere(val S:sommetArbrePlanaire):sommetArbrePlanaire;
fonction créerArborescence(val racine:objet):sommetArbrePlanaire;
fonction ajouterFils(ref S:sommetArbrePlanaire,val x:objet):vide;
fonction supprimerSommet(ref S:sommetArbrePlanaire):vide;
fonction créerArbreBPlaniare(val Racine:objet):sommet;