Corrigé E.D. Algorithmes et Structures de Données n° 3

Thème : Arbres binaires et Tas

Exercice III.1 Expressions : Parcours d'arbre

Question 1

Question 2 Ecrivez une méthode qui teste si un arbre binaire est une feuille (c'est-à-dire un arbre dont les deux sous-arbres sont vides).

Algorithme :

Si sous-arbre droit est vide et sous-arbre gauche est vide alors l'arbre est une feuille

FinSi

```
booléen est_feuille(){

retourner ((this.filsgauche==null) && (this.filsgauche==null))
}
```

Question 3 Ecrivez une méthode qui affiche les arbres binaires d'opérateurs sous forme agréable (" 2*(3+1) ").

Nous allons d'abord écrire une méthode qui, pour un opérateur donné, affiche le symbole correspondant :

```
void printOp(Op O ) {

          switch (O) {
               case fois : print(" * ") ;
               case plus  :   print(" + ") ;
               case moins  :   print(" – ") ;
               case sur  :   print(" / ") ;
               case zero  :   print(0) ;
               case un  :   print(1) ;
               case deux  :   print(2) ;
               case trois  :   print(3) ;
}
```

Nous allons maintenant décrire la méthode qui va afficher l'arbre binaire.
On remarquera que si la racine n'est pas une feuille, alors c'est un opérateur binaire.
L'algorithme est le suivant :
Si l'arbre est une feuille, on affiche le contenu de sa racine
Sinon on affiche une parenthèse ouvrante, le sous-arbre gauche, le contenu de la racine, le sous-arbre droit et une parenthèse fermante.
Finsi

```
void affiche() {

          if (this.est_feuille()
               printOp(this.contenu());
          else {
```

```
                print("(");
                this.filsgauche.affiche();
                printOp(this.contenu());
                this.filsdroit.affiche();
                print(")");
            }
        }
```

Question 4 Ecrire une méthode qui calcule la valeur associée à l'arbre binaire (résultat de type entier).

Suivant la racine de l'arbre
    Si c'est une valeur alors renvoyer ce résultat Finsi
    Si c'est un opérateur
        Calculer la valeur associée au sous-arbre gauche
        Calculer la valeur associée au sous-arbre droit
        Appliquer l'opérateur à ces deux valeurs
        Renvoyer le résultat de l'opération
    Sinon
        Gestion d'une erreur
    FinSi
FinSuivant
FinSi

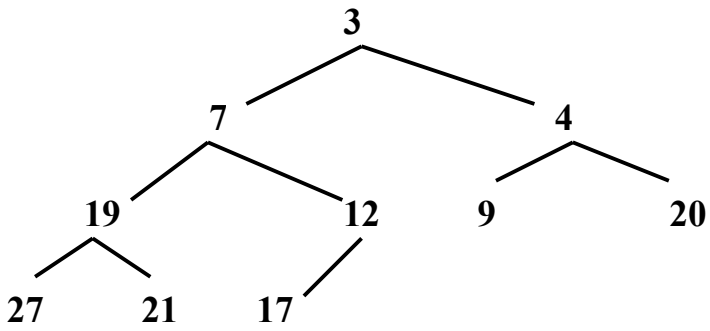
```
entier eval(){
entier x, y, resultat;

     switch (this.contenu){
          case zero : resultat = 0;
          case un : resultat = 1;
          case deux : resultat = 2;
          case trois : resultat = 3;
          case default :
          x = this.filsgauche.eval() ;
          y = this.filsdroit.eval();
          switch (this.contenu()) {
               case plus : resultat = x + y;
               case moins : resultat = x – y;
               case fois : resultat = x * y;
               case sur : resultat = x / y;
               default : erreur;
          }
     }
     retourner resultat;
}
```

Construction de l'arbre binaire correspondant à 2*(3+1), affichage de la formule et de la valeur :

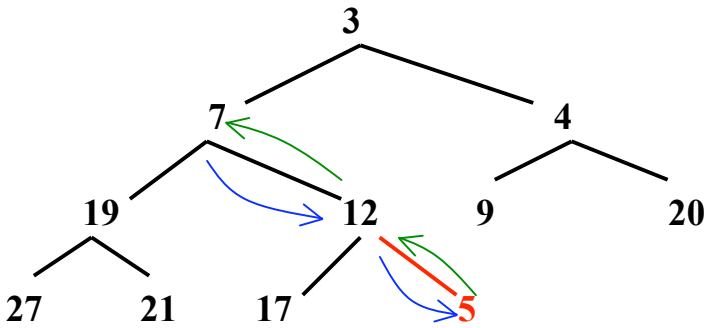
```
ArbreBinaireOp a ;
ArbreBinaireOp  ad= new ArbreBinaireOp(deux,null,null);
ArbreBinaireOp ag1= new ArbreBinaireOp(trois,null,null);
ArbreBinaireOp ag2= new ArbreBinaireOp(un,null,null);
ag1 = new ArbreBinaireOp(plus,ag1,ag2);
a= new ArbreBinaireOp(fois,ad,ag1);
a.affiche();
print(a.eval());
```

Exercice III.3 Soient des nombres rangés dans la structure suivante:

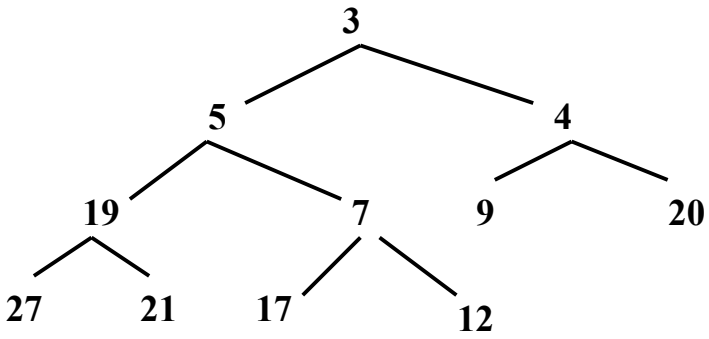


Question 1 Il s'agit d'un tas. C'est un arbre binaire parfait, (c'est à dire que toutes les feuilles ont presque la même distance par rapport à la racine) tel que le contenu de chaque sommet est inférieur ou égal à celui de ses fils.

Question 2 Insertion de 5

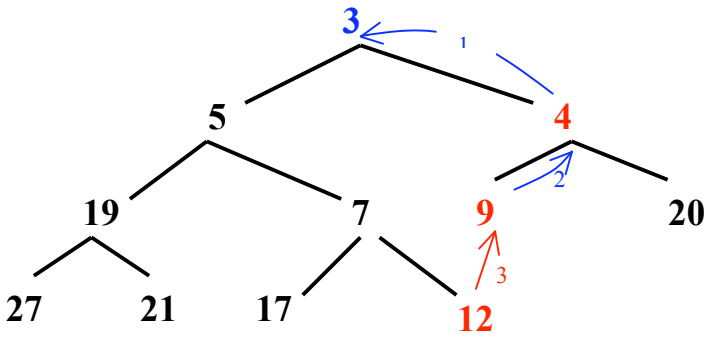


Résultat :

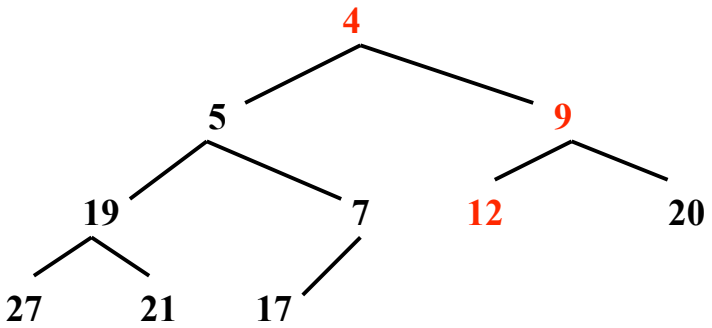


La structure obtenue est de nouveau un tas.

Question 3 Même question si ensuite on supprime 3 (dans l'arbre obtenu en 2).
Rappel : la dernière feuille de l'arbre va disparaître : en partant du haut, on remonte le plus petit des fils tant qu'il est inférieur à la dernière feuille ; ensuite, on place le contenu de la dernière feuille dans la place libre.



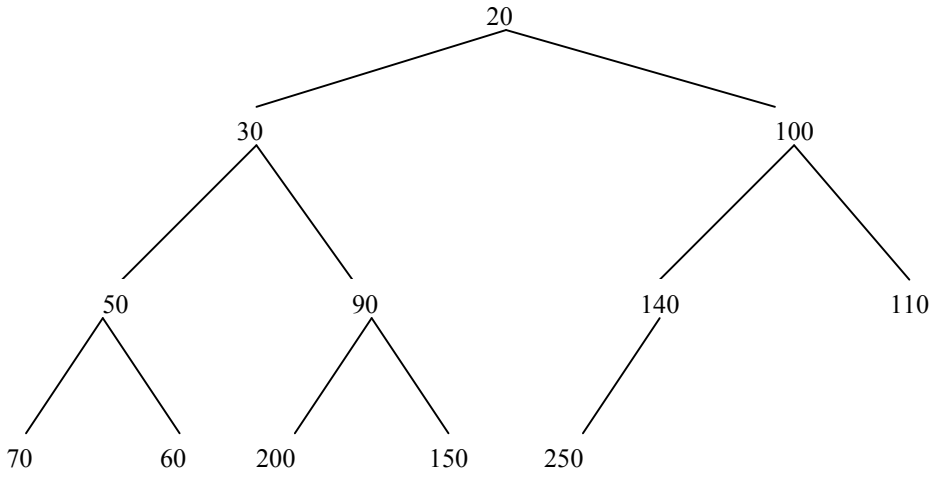
Résultat :



De nouveau, on obtient un tas.

Exercice III.3

Soit le tas suivant (on suppose que les nombres correspondent à des clés)



Question 1 On utilise un tableau T_ARB pour représenter cet arbre. Donner ce tableau.
Rappel du cours : Si un sommet est à la place (I) du tableau alors son fils gauche est en (2*I) et son fils droit en (2*I +1).

Cela donne, pour l'arbre ci-dessus, la tableau suivant :

1	2	3	4	5	6	7	8	9	10	11	12
20	30	100	50	90	140	110	70	60	200	150	250

Question 2 Soit la méthode récursive suivante applicable à un tas non vide

```
void lecture () {
      si (gauche /= null) alors
      gauche.lecture() ;
      finsi ;
      ecrire (this.contenu() );// écrit la valeur de la racine
      si (droit /= null) alors
      droit.lecture ();
      finsi;
      }
```

La méthode lecture traite d'abord la totalité du sous-arbre gauche, écrit la valeur de la racine, puis traite la totalité du sous-arbre droit.
La première valeur qui s'affiche est donc la feuille la plus à gauche, puis le père de cette feuille, le sous-arbre droit de ce père, ... Il s'affiche donc, dans l'ordre :

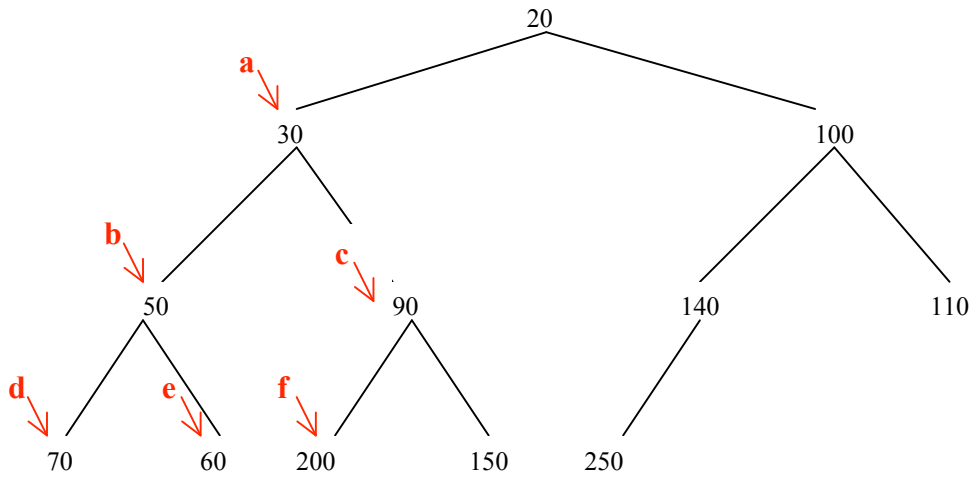
70, 50, 60, 30, 200, 90, 150, 20, 250, 140, 100, 110

Question 3

On suppose ici que `a, b, c, d, e` et `f` désignent des sous-arbres de l'arbre initial et `a, b, c, d, e` et `f` sont tels que :

```
a.contenu=30;      b.contenu=50;      c.contenu=90;
d.contenu=70;      e.contenu=60;      f.contenu=200;
```

On peut schématiser ces pointeurs de la façon suivante :



On a ainsi :

```
b.gauche=d;                a.droit.contenu=90;
f.droit=null;              a.gauche.droit.contenu=60
```

Noter que `b.gauche, f.droit` sont de type `ArbreBinaire` alors que `a.droit.contenu` et `a.gauche.droit.contenu` sont de type C_cle.