TD12 – Arbres binaires et ABR

1. Taille d'un arbre binaire (nb noeuds)

```
Fonction taille (a : arbrebinaire) : entier
Var res : entier
Début
Si arbre-vide(a)
  Alors res ← 0
  Sinon res ← 1 + taille(fils-gauche(a)) + taille(fils-droit(a))
FinSi
Retourner(res)
Fin
```

2. Hauteur d'un arbre binaire (lg de la plus longue branche)

```
Fonction hauteur (a : arbrebinaire) : entier
Var res : entier
Début
Si arbre-vide(a)
      ou (arbre-vide(fils-gauche(a)) et arbre-vide(fils-droit(a))
  Alors res ← 0
  Sinon res ← 1 + max(hauteur(fils-gauche(a)),hauteur(fils-droit(a)))
FinSi
Retourner(res)
Fin
```

3. Arbre binaire entier ou parfait

- Un **arbre binaire entier** est un arbre dont tous les nœuds possèdent zéro ou deux fils.
- Un **arbre binaire parfait** est un arbre binaire entier dans lequel toutes les feuilles sont à la même profondeur.

```
Fonction est-entier (a : arbrebinaire) : booléen
Var res : booléen
Début
Si arbre-vide(a)
   Alors res←vrai
   Sinon res←not(arbre-vide(fils-gauche(a)) XOR arbre-vide(fils-droit(a)))
             et est-entier(fils-gauche(a)) et est-entier(fils-droit(a)))
FinSi
Retourner(res)
Fin

Fonction est-parfait (a : arbrebinaire) : booléen
Var res : booléen
Début
Si arbre-vide(a)
  Alors res←vrai
  Sinon res← not(arbre-vide(fils-gauche(a)) XOR arbre-vide(fils-droit(a)))
             et hauteur(fils-gauche(a))=hauteur(fils-droit(a))
             et est-parfait(fils-gauche(a)) et est-parfait(fils-droit(a)))
FinSi
Retourner(res)
Fin
```

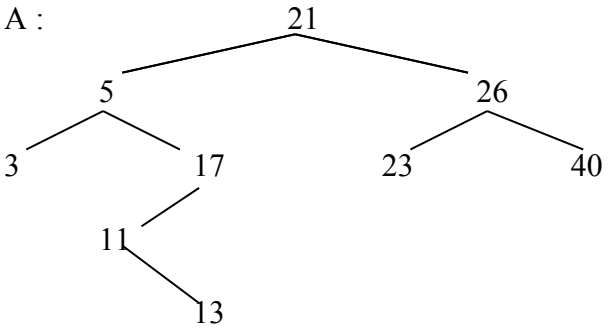
2. Coupure d'un ABR

L'opération de coupure d'un ABR en l'élément x consiste à scinder l'arbre A en deux ABR G et D où G contient les éléments de A inférieurs ou égaux à x et D contient les éléments de A strictement supérieurs à x. On dit que G (resp. D) est la coupure gauche (resp. droite) de A.

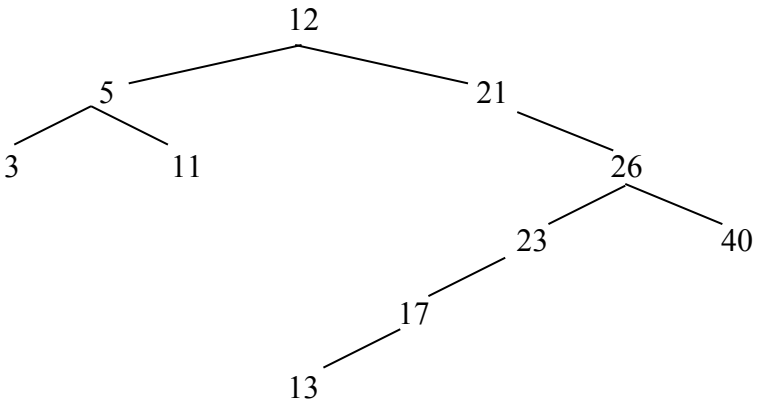
Coupure d'un élément X dans un ARBIN A en 2 ARBIN G et D
- il n'est pas nécessaire de parcourir TOUS les noeuds de A, seulement les nœuds N situés sur le chemin de recherche de X dans A
- si nœud N<X : G ← N + Fils-Gauche(N) sur le bord droit de G
- si nœud N X : D ← N + Fils-Droit(N) sur le bord gauche de D

2.1. Donner la coupure de A en 12, en 30 et en 5.

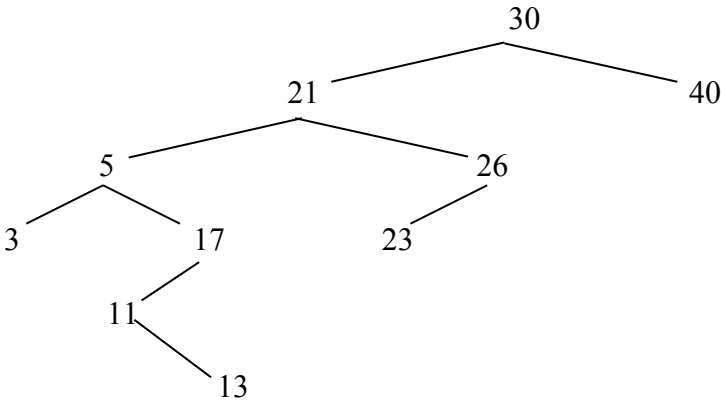
A :



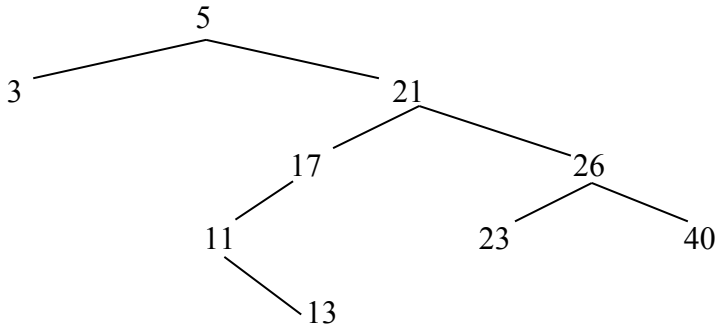
En 12 :



En 30 :



En 5 :



2.2. Ecrire l'algorithme de coupure sur un ABR en fonction des opérations définies sur le type abstrait `arbrebinaire`.

Le principe de l'algorithme de coupure est le suivant, quatre cas sont possibles :

- A est vide : G et D sont vides.
- la valeur de la racine de A est supérieure ou égale à x : D est égal à A à la différence que le fils gauche de D est la coupure droite du fils gauche de A ; G est la coupure gauche du fils gauche de A.
- la valeur de la racine de A est inférieure à x : G est égal à A à la différence que le fils droit de G est la coupure gauche du fils droit de A ; D est la coupure droite du fils droit de A.

```
Procédure Coupure (E x : élément, E A : arbrebin, S G,D : arbrebin)
Début
Si arbre-vide(A)
   Alors
     G←créer-arbre-vide()
     D←créer-arbre-vide()
   Sinon
     Si x ≤ val(A)
        Alors
          D←A
          Coupure(x,fils-gauche(A),G,fils-gauche(D))
        Sinon
          G←A
          Coupure(x,fils-droit(A),fils-droit(G),D)
     FinSi
FinSi
Fin
```