Algorithmique 1
TD 6 Arbres planaires

Soit la définition du type abstrait `sommetArbrePlanaire` vue en cours.
Pour rappel voir annexe A.

Exercice 6.1 *Exemples*

Soit l'arbre planaire illustré sur la figure 1.

1. Donner les suites de sommets correspondant respectivement aux parcours préfixe, postfixe et hiérarchique.
2. Tout noeud d'un arbre est la racine du sous-arbre constitué par sa descendance et lui-même. Dénombrer l'ensemble de sous-arbres pour l'arbre planaire de la figure 1 .

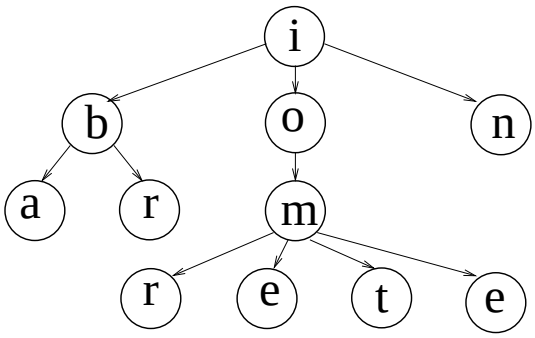

FIGURE 1 – Arbre planaire

Exercice 6.2 *Construction*

En utilisant les primitives du type `sommetArbrePlanaire` écrire une fonction qui construit l'arbre planaire de la figure 2(a).

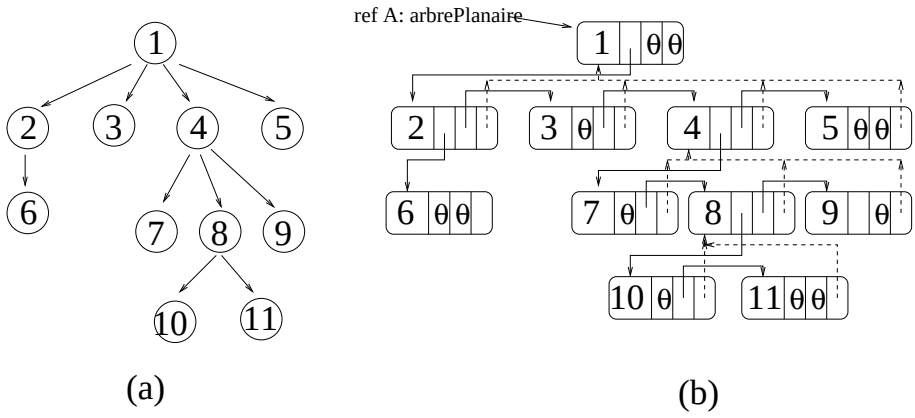

FIGURE 2 – Représentation d'un arbre planaire

Exercice 6.3 *Implémentation d'un arbre planaire par allocation dynamique "premierFils&Frère"*
Soit l'implémentation de `sommetArbrePlanaire` vue en cours et illustrée sur la figure 2(a).
Pour rappel voir annexe B.
Ecrire les primitives `ajouterFils` et `supprimerSommet`

Exercice 6.4 *Implémentation d'un arbre planaire dans le type arbreBinaire*
Tout arbre planaire peut être représenté par un arbre binaire en prenant comme lien gauche de tout noeud le lien vers son premier fils, et comme lien droit le lien vers son frère de droite.

1. Soit l'arbre planaire de la figure 2(a). Dessiner sa représentation binaire.
2. Dessiner l'arbre plainaire qui correspond à la représentation binaire illustrée sur la figure 3.
3. Donner l'implémentation des primitives du type `sommetArbrePlanaire` dans le type `arbreBinaire`.

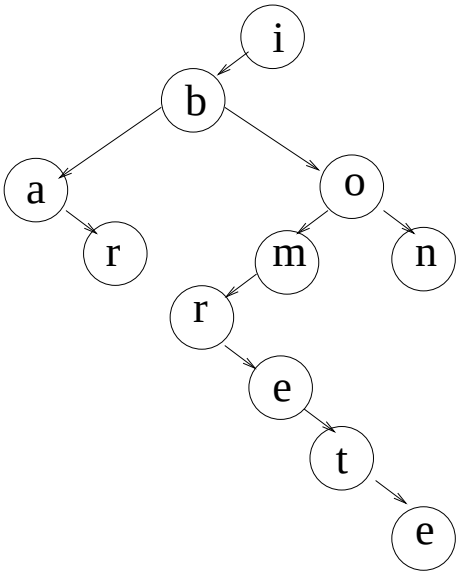

FIGURE 3 – Représentation binaire d'un arbre planaire

Exercice 6.5 *Parcours*

1. Ecrire une fonction de parcours préfixe itératif d'un arbre planaire.
2. Ecrire une fonction de parcours préfixe itératif d'un arbre planaire en utilisant une pile.
3. Ecrire une fonction de parcours par niveaux (parcours hiérarchique) d'un arbre planaire en utilisant une file.

Problème récurrent (notre fil d'ariane)
Exercice 6.6 Gestion d'une piste d'atterrissage des avions

Un avion est un enregistrement contenant :
– l'indicatif (6 caractères)
– la destination (30 caractères)
– l'autonomie résiduelle de carburant comptée en heures de vol (entier)
– deux booléens indiquant s'il y a un pirate à bord et s'il y a le feu.

1. Définir les structures de données nécessaires.
2. Ecrire la fonction `Priorité` ainsi que la gestion complète de la piste.

3. Envisager le cas de suppression d'un élément quelconque de la file lorsque le pirate a mis sa menace de détournement à exécution.

Quelles sont les notions que vous venez de voir qui peuvent permettre d'amorcer le fil d'ariane ?

Annexe A **Type abstrait** *sommetArbrePlanaire*

```
sommetArbrePlanaire= curseur;
```

- **Création**
  ```
  fonction creerArbrePlanaire(val Racine:objet):sommetArbrePlanaire;
  ```
- **Accès**
  ```
  fonction getValeur(val S:sommetArbrePlanaire):objet;
  fonction premierFils(val S:sommetArbrePlanaire):sommetArbrePlanaire;
  fonction frere(val S:sommetArbrePlanaire):sommetArbrePlanaire;
  fonction pere(val S:sommetArbrePlanaire):sommetArbrePlanaire;
  ```
- **Modification**
  ```
  fonction setValeur(ref S:sommetArbrePlanaire, val x:objet):vide;
  fonction ajouterFils(ref S:sommetArbrePlanaire, val x:objet):vide;
  fonction supprimerSommet(ref S:sommetArbrePlanaire):vide;
  fonction detruireArbrePlanaire(ref S:sommetArbrePlanaire):vide;
  ```

Annexe B **Implémentation du type abstrait** *sommetArbrePlanaire*

```
cellule= structure
          info: objet;
          premierFils: sommet
          frere: sommet;
          pere: sommet
         finstructure
sommet= ^cellule;
sommetArbrePlanaire= sommet;
```