# Algorithmique Contrôle nº 2

Info-Sup - Epita

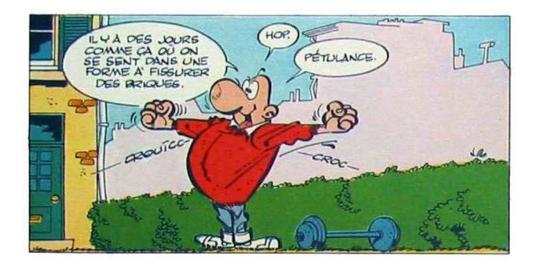
D.S. 311236.11 BW (27 mar 2012 - 10:00)

# Remarques (à lire!):

- □ Vous devez répondre sur les feuilles de réponses prévues à cet effet.
  - Aucune autre feuille ne sera ramassée (gardez vos brouillons pour vous).
  - Répondez dans les espaces prévus, les réponses en dehors ne seront pas corrigées : utilisez des brouillons!
  - Ne séparez pas les feuilles à moins de pouvoir les ré-agrafer pour les rendre.
  - Aucune réponse au crayon de papier ne sera corrigée.
- □ La présentation est notée en moins, c'est à dire que vous êtes noté sur 20 et que les points de présentation (2 au maximum) sont retirés de cette note.

#### $\square$ Les algorithmes :

- Tout algorithme doit être écrit dans le langage Algo (pas de C#, Caml ou autre).
- Tout code Algo non indenté ne sera pas corrigé.
- En dehors d'indication dans les énoncés, vous ne pouvez utiliser aucune routine (fonction ou procédure) supplémentaire.
- Tout ce dont vous avez besoin (opérations de types abtraits, types) est donné en annexe.
- Rappel : pour chaque algorithme, lorsque demandé sur les feuilles de réponses, vous devez donner :
  - Les spécifications c'est à dire ce qu'il fait, les paramètres (types et significations) et les éventuelles conditions d'utilisation.
  - Le principe algorithmique c'est à dire en clair la méthode retenue pour résoudre le problème. Attention le principe et les spécifications sont notés, ne pas les envisager revient à sacrifier directement des points.
- $\square$  Durée : 2h.



## Exercice 1 (ABR: chemin de recherche - 2 points)

Soit un arbre binaire de recherche contenant des valeurs entières. On désire chercher la valeur 42. Quelle(s) séquence(s) parmi les suivantes, **ne pourrai(en)t pas** être la suite des noeuds parcourus? Entourer sur les feuilles de réponses le(s) numéro(s) de séquence impossible.

- (T) 50 15 48 22 46 42
- (2) 48 15 45 22 47 42
- (3) 15 22 45 43 35 42
- (4) 22 45 43 15 35 42

## Exercice 2 (Trichotomie - 7 points)

La recherche dichotomique (principe bien connu) consiste à séparer en deux parties égales (l'égalité dans ce cas est une quasi-égalité dans la mesure où il peut y avoir un élément d'écart entre les deux parties) l'ensemble de données sur lequel se fait la recherche d'un élément. La trichotomie consiste elle en une découpe de l'ensemble en trois parties quasi-égales (même remarque).

- 1. Donnez le principe de recherche trichotomique d'un élément x dans une liste 1 de n éléments. Votre principe doit décrire une fonction entière qui retourne le rang de x dans 1 s'il existe et 0 sinon.
- 2. Selon ce principe, en utilisant les opérations définies par le type algébrique abstrait liste itérative rappelé en annexe, écrivez l'algorithme abstrait de la fonction récursive trichotomie (x,1,g,d) où x est l'élément recherché, 1 la liste dans laquelle s'effectue la recherche et g et d les bornes gauche et droite de recherche dans la liste.

Par exemple pour la recherche d'un élément x dans une liste 1 entre le premier et le dernier élément de la liste, l'appel serait : trichotomie(x, l, 1, lonqueur(l))

## Exercice 3 (Listes chaînées : occurrences - 5 points)

Après avoir donné son principe, écrire une fonction **itérative** qui donne le nombre d'occurrences d'une valeur x dans une liste chaînée L (de type  $t_pListe$ , donné en annexe).

### Exercice 4 (Listes chaînées: insertion - 6 points)

Écrire un algorithme **itératif** qui insère une valeur donnée dans une liste chaînée (de type t\_pListe) triée en ordre croissant (au sens large, la valeur est insérée même si elle est déjà présente).

# Annexes

### Liste itérative

```
SORTE
     Liste, Place
UTILISE
      Entier, Element
OPERATIONS
     \texttt{Liste-vide} \,:\, \to\, \texttt{Liste}
     accès
                    : Liste 	imes Entier 	o Place
      contenu
                     : Place 
ightarrow Element
      ième
                     : Liste 	imes Entier 	o Element
      longueur
                     : Liste 
ightarrow Entier
      \mathtt{supprimer} \quad : \; \mathtt{Liste} \; \times \; \mathtt{Entier} \; \to \; \mathtt{Liste}
                     : Liste \times Entier \times Element \to Liste
      insérer
      succ
                     : Place 
ightarrow Place
```

# Implémentation dynamique des listes

Les listes chaînées sont les mêmes que celles utilisées en TD, et sont représentées par le type suivant :