Sujet G

❑ Exercice : *type A*

Dans cet exercice, on autorise les doublons dans un arbre binaire de recherche et pour le cas d'égalité on choisira le sous-arbre gauche. On ne cherchera pas à équilibrer les arbres.

1. Rappeler la définition d'un arbre binaire de recherche.
2. Insérer successivement et une à une dans un arbre binaire de recherche initialement vide toutes les lettres du mot `bacddabdbae`, en utilisant l'ordre alphabétique sur les lettres. Quelle est la hauteur de l'arbre ainsi obtenu ?
3. Montrer que le parcours en profondeur infixe d'un arbre binaire de recherche de lettres est un mot dont les lettres sont rangées dans l'ordre croissant. On pourra procéder par induction structurelle.
4. Proposer un algorithme qui permet de compter le nombre d'occurrences d'une lettre dans un arbre binaire de recherche de lettres. Quelle est sa complexité ?

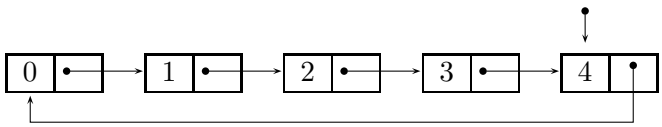
❑ Exercice : *type B*

Le langage utilisé dans cet exercice est le langage C.

Certaines comptines enfantines ont pour objectif de désigner une personne « au hasard », un exemple bien connu est « *Am, stram, gram, pic et pic et colégram* ». On suppose que N enfants numérotés de 0 à $N-1$ sont assis en cercle et que l'un d'entre eux (le numéro k) récite une comptine contenant S syllabes. A la première syllabe il désigne son suivant immédiat dans le cercle puis il avance d'un enfant à chaque syllabe jusqu'à la fin de la comptine. L'enfant désigné à la fin de la comptine doit quitter le cercle et le processus recommence à partir de son suivant immédiat jusqu'à ce qu'un seul enfant reste.

1. Donner une illustration de ce processus avec $N = 5$ et $S = 7$, en supposant que l'enfant 0 commence et indiquer le numéro du dernier enfant restant.

On veut implémenter une structure de données de liste chainée circulaire permettant de représenter les enfants assis en cercle. On rappelle qu'une liste chainée circulaire est une liste dont le dernier maillon pointe vers le premier maillon. L'accès au cercle se fait via un pointeur qui désigne l'enfant qui *précède celui qui doit dire la comptine*. Par exemple un cercle initial de cinq enfants où le premier à dire la comptine est le numéro 0 est représenté par la structure de données ci-dessous :



Afin d'implémenter cette structure de données, on propose d'utiliser les types suivants

```c
struct maillon_s
{
    int valeur;
    struct maillon_s *suivant;
};
typedef struct maillon_s maillon;
typedef maillon *liste_circulaire;
```

Ce type est défini dans le fichier compagnon de cet exercice téléchargeable à l'adresse :
`https://fabricenativel.github.io/cpge-info/oraux/`.
Ce fichier contient également une fonction de prototype `void affiche(liste_circulaire lc)` qui affiche le contenu de la liste circulaire à partir du maillon *qui suit* celui désigné par `lc`.

2. Ecrire une fonction de signature `void ajouter(liste_circulaire *lc, int v)` qui prend en paramètre un pointeur vers une liste circulaire et ajoute un maillon de valeur `v` après le maillon pointé par `lc` et met à jour le pointeur `lc` pour qu'il pointe vers le nouveau maillon.
3. En déduite une fonction de signature `liste_circulaire creer_cercle(int n)` qui prend en paramètre un entier `n` et qui crée un cercle de `n` enfants numérotés de 0 à `n-1`.
4. Ecrire une fonction de signature `void avance(liste_circulaire *lc, int s)` qui modifie la liste circulaire `lc` en avançant de `s` maillons.

5. Ecrire une fonction de signature `void enleve(liste_circulaire *lc)` qui modifie la liste circulaire `lc` en enlevant le maillon *qui suit* celui pointé par `lc`.
6. En déduire une fonction de signature `int dernier_enfant(int n, int s)` qui renvoie le numéro du dernier enfant restant dans le cercle de `n` enfants après avoir récité une comptine de `s` syllabes en supposant que l'enfant 0 commence.