

2. Une liste doublement chaînée avec sentinelle est une liste dont le premier élément est le suivant de la sentinelle et le dernier élément de la liste est le précédent de la sentinelle. On a donc désormais la primitive **sentinelle**. Soit la liste composée des éléments [1, 4, 2, 5]. Dessinez cette liste doublement chaînée avec sentinelle.

3. Écrivez la fonction **premier**(L) qui renvoie le premier élément d'une liste doublement chaînée avec sentinelle L.

4. Écrivez la fonction **dernier**(L) qui renvoie le dernier élément d'une liste doublement chaînée avec sentinelle L.

-
5. Écrivez la fonction `ajouteEnTête(x, L)` qui ajoute `x` au début d'une liste doublement chaînée avec sentinelle `L`.

6. Écrivez la fonction `ajouteAprès(x, y, L)` qui ajoute `x` après l'élément `y` d'une liste doublement chaînée avec sentinelle `L`.

2 Nouveau système de notation (6 points)

Une maîtresse d'école décide d'instaurer un système de points correspondant à la participation des élèves. À chaque élève, est associée une pile de type LIFO qui mémorise les différentes réponses qu'il a faites. Si un élève répond bien à une question, alors un feu vert est empilé sur sa pile. Si un élève répond de façon approximative à une question, alors un feu jaune est empilé sur sa pile. Enfin, si un élève répond mal à une question, alors un feu rouge est empilé sur sa pile.

Un élève a 0 point initialement, puis les points sont comptés de la façon suivante : si la pile contient 3 fois de suite la même couleur de feu, alors le nombre de points de l'élève est incrémenté de 10 points si les feux sont verts, 5 points s'ils sont jaunes et décrétementé de 5 points s'ils sont rouges.

-
-
2. Écrivez une fonction qui prend en entrée la pile d'un élève munie des primitives vues en cours, et un tableau de feux et qui modifie la pile afin de calculer la valeur associée à l'élève et qui renvoie cette valeur. On pourra considérer que la pile est vide lors de l'appel de la fonction.

3 Structure de Tas (6 points)

Rappel de cours :

Un tas ascendant est un arbre binaire vérifiant les propriétés suivantes :

- la différence maximale de profondeur entre deux feuilles est de 1 (*i.e.* toutes les feuilles se trouvent sur la dernière ou sur l'avant-dernière ligne) ;
- les feuilles de profondeur maximale sont "tassées" sur la gauche.
- chaque nœud est de valeur supérieure à celle de ces deux fils.

Un tas ou un arbre binaire presque complet peut être stocké dans un tableau, en posant que les deux descendants de l'élément d'indice n sont les éléments d'indices $2n$ et $2n + 1$ (pour un tableau indicé à partir de 1). En d'autres termes, les nœuds de l'arbre sont placés dans le tableau ligne par ligne, chaque ligne étant décrite de gauche à droite.

L'insertion d'un élément dans un tas se fait de la façon suivante : on place l'élément sur la première case libre et on échange l'élément et son père quand ce dernier est inférieur et qu'il existe.

L'opération de **tamisage** consiste à échanger la racine avec le plus grand de ses fils, et ainsi de suite récursivement jusqu'à ce qu'elle soit à sa place.

1. Dessinez un tas correspondant au tableau $T : [18, 10, 17, 5, 8, 15, 4, 2]$

2. Insérez la valeur 12 dans le tas et donnez le tas final.

-
-
3. Écrivez la fonction `insère(x, T, n)` permettant d'insérer une valeur x dans un tas ascendant représenté par le tableau T dont la taille est n .

-
-
-
-
-
-
-
-
-
-
-
-
-
-
-
-
-
-
-
-
-
-
-
-
4. Remplacez la racine du tas par la valeur 3 et détaillez le tamisage.