CPE Lyon - 3IRC Année 2021/2022

Structures de données et algorithmes avancés

Séance 1 - Structures de données

Exercice 1. Tables de Hachage (TD)

Dans une table de hachage de taille `11`, les clés `9, 26, 50, 15, 2, 21, 36, 22, et 31` sont insérées dans cet ordre à l'aide de la fonction de hachage `modulo`. Pour chacune des deux méthodes ci-dessous, dessinez la table de hachage finale :

- chaînage
- sondage linéaire

Exercice 2. Listes chaînées

1. Implémentez une classe `ListeChainee` **sans utiliser les listes de Python** : vous devez donc d'abord écrire une classe `Noeud`, puis utiliser cette classe pour votre `ListeChainee`.
2. Ajoutez les méthodes `inserer(valeur, k)`, `supprimer(k)`, `rechercher(valeur)`, `taille()` et `estVide()`.

Exercice 3. Piles

Réutilisez le code de la question précédente pour implémenter une classe `Pile` et toutes ses méthodes. Utilisez-la pour écrire un validateur de parenthésage, c'est-à-dire un programme qui prend en paramètre un fichier texte, et qui valide que chaque caractère *ouvrant* `{`, `<`, `(` ou `[` est **correectement** associé au caractère *fermant* correspondant (tous les éventuels autres caractères du fichier ne sont pas pris en compte). Le programme doit **écrire** `"Aucune erreur de syntaxe"` si le fichier est valide, ou "Erreur de syntaxe à la ligne k"

Exercice 4. Tables de hachage

Implémentez une classe `HashTable`. Utilisez-la pour écrire un programme qui compte le nombre d'occurrences de chaque mot d'un texte. Complétez ensuite ce programme pour qu'il affiche toutes les occurrences d'un mot recherché, et les affiche sous la forme :

```
Le mot 'algorithme' apparaît :
- à la ligne 3, mot 4
- à la ligne 7, mot 8
- à la ligne 17, mot 14
```

Exercice 5. Problème de Josephus (Examen 2021)

Le problème de Joséphus est un célèbre problème d'algorithmique inspiré d'un fait de l'Antiquité : en l'an 67, l'historien Flavius Josèphe, dit Joséphus, et 40 de ses soldats furent piégés dans une grotte par les Romains. Refusant de se rendre, ils décidèrent de s'exécuter mutuellement en appliquant la règle suivante : ils formèrent un cercle, et tuèrent un soldat sur 3 (à partir de l'un d'entre eux désigné comme n°1) tant qu'il y resta des survivants, le dernier promettant de se donner lui-même la mort (cf. Figures 1 et 2). Peu enclin à subir un tel sort, Joséphus trouva où se positionner dans le cercle pour être le dernier désigné, et choisit de se rendre aux Romains... qui finalement l'épargnèrent.

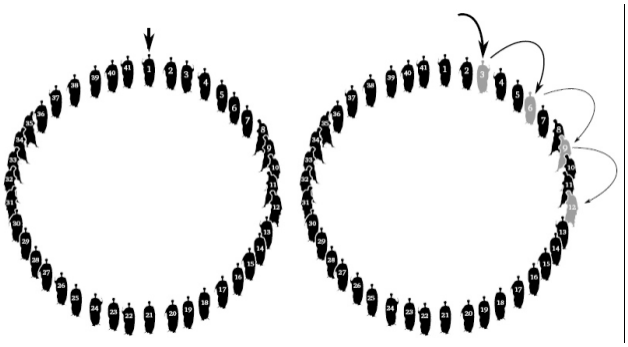

FIGURE 1 – Un soldat est désigné comme étant le n°1.
Les premiers éliminés portent les numéros 3, 6, 9 et 12.

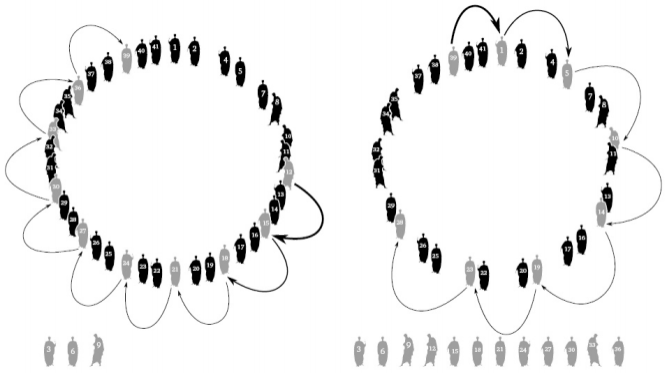

FIGURE 2 – Les éliminations continuent, jusqu'à faire un tour complet.
Aux tours suivants, on élimine toujours une personne sur trois avec
les soldats restants.

Le problème de Joséphus consiste à calculer l'ordre d'élimination de n soldats en cercle où un soldat sur k est éliminé, aussi appelé (n, k)**-permutation de Joséphus** (et en particulier son dernier élément, puisqu'il s'agit de la position permettant de rester sain et sauf) ; par exemple, la $(41, 3)$-permutation de l'exemple ci-dessus est `[3, 6, 9, 12, 15, ..., 31]`.

1. Quelle est la $(15\,;\,4)$-permutation de Joséphus ?
2. Quelle structure de données vous semble la plus appropriée pour modéliser le problème de Joséphus **(soyez précis et justifiez)** ?
3. Codez la fonction `Josephus(n, k)` qui calcule et retourne la $(n; k)$-permutation de Joséphus