Compte-Rendu de la séance n°2

Nom de la formatrice.	MARZAK Bouchra
Nom de l'apprenant.	JAMAL EDDINE ELIDRISSI Yassir.
Date.	19/11/2021

Aujourd'hui, on a reçu le premier brief visant deux objectifs, d'une part développer la capacité de problème solving et d'une autre découvrir Java, en l'occurrence les base de ce dernier (Les variables, les Conditions, les Boucles, les Collections, les Fonctions).

Pour pouvoir programmer une solution à l'algorithme donné il me semble nécessaire de faire le tour concernant les outils à mettre en place pour mettre en place le programme, dans les lignes qui suivent je relate les différents concepts que j'ai survoler lors de la journée :

En me documentant j'ai suivi une chronologie logique, en commençant avec ArrayList puis Hashmap pour en finir avec Hashset.

1) ArrayList, il s'agit d'une façon de stocker des entrés en tableau et ce en implémentant l'interface liste (concrètement en important java.util.List). En effet, l'interface list modélise une liste indexée par des entiers, elle étend Collection, et lui ajoute des méthodes de manipulation et des méthodes de parcours, pour cela Arraylist en plus des méthodes de list elle implémente ses propres méthodes (à savoir la manipulation de la taille d'un tableau par exemple)
2) Pamis les méthodes d'arraylist que j'ai survolé on trouve ensureCapacity servant à augmenté la capacité d'arrayList (on peut même travailler avec la syntaxe suivante : ArrayList<>(le nombre initiale à allouer)).
3) ArrayList est multithread.
4) Nb : il existe vector assurant la même fonctionnalité qu'ArrayList mais c'est désormais LegacyClass.
5) Une autre composante de collection c'est Hashmap, mais la manière dont ils stock et traite les données est différente, la majeure différence c'est que ça stock en étant deux objets clé et sa valeur (sachant que arraylist stock un seul objet en étant la valeur). Hashmap implémente l'interface map pour cela une importation java.util.Map est nécessaire, parmis les caractéristiques de Hashmap c'est qu'elle permet l'utilisation de la valeur null comme clé et valeur. Toutefois, parmis les méthodes qu'on peut effectuer sur Hashmap on trouve put (similaire à la méthode add dans arraylist) ou bien la méthode .get() utilisée principalement pour récupérer la valeur mappée par son clé, cette méthode renvoie NULL si la Hashmap ne

contient pas de clé (quant à la même méthode (.get()) dans ArrayList elle retourne l'objet à une position donnée (paramètre 6 renvoie l'objet dans la 6e position).

6) Pour Hashset, elle implemente l'interface set et elle utilise une HashMap dont la clé est l'élément et dont la valeur est une instance d'Object identique pour tous les éléments. Elle ne propose aucune garantie sur l'ordre de parcours lors de l'itération sur les éléments qu'elle contient et si une valeur est ajoutée deux fois, elle n'apparaît qu'une fois, car chaque élément d'un ensemble doit être unique, mais elle permet l'ajout d'un élément null, de même la classe HashSet possède de nombreuses méthodes utiles. Par exemple, pour ajouter des éléments, on utilise la méthode add(), dans Hashset on a la possibilite de parcourir avec un objet dit Iterator (import java.util.Iterator).
7) NB : l'ordre des éléments insérés n'est naturellement pas conservé (bonus : pour les faire ordonner -> LinkedHashSet).

À suivre …