Programmation & Algorithmique 2

TP - Séance 3

8 mars 2023

Matière visée : Utilisation de tableaux, paramètres en ligne de commande et conversion de String.

1 Tableaux

Veuillez créer une classe `ExoArray` qui contiendra une série de méthodes de classe.

1. Dans la méthode `main`, veuillez effectuer les choses suivantes :
   — créez un tableau pouvant contenir 20 entiers et remplissez-le avec les 20 premières puissances de 2 ;
   — créez un tableau pouvant contenir 20 entiers et remplissez-le avec les 20 premiers nombres impairs (par odre décroissant) ; et
   — créez un tableau pouvant contenir 20 entiers et remplissez-le de telle manière à ce que l'élément en position i contienne la valeur $(-1)^i \times i$;
2. créez une méthode `display` qui prend un tableau d'entiers en paramètre. Cette méthode doit afficher à l'écran les éléments du tableau (séparés par un espace) ;
3. créez une méthode `sum` qui prend un tableau d'entiers en paramètre et retourne la somme des éléments présents dans le tableau ;
4. créez une méthode `average` qui prend un tableau d'entiers en paramètre et retourne la moyenne des éléments présents dans le tableau ;

   Attention : Veuillez réaliser l'exercice 2 avant de poursuivre avec les points suivants.

5. créez une méthode `zipSum` qui prend deux tableaux d'entiers en paramètres et qui retourne un nouveau tableau dont l'élément d'indice `i` vaut la somme des éléments en indice `i` des deux tableaux passés en paramètres ;
6. créez une méthode `zipAverage` qui prend deux tableaux d'entiers en paramètres et qui retourne un nouveau tableau dont l'élément d'indice `i` vaut la moyenne des éléments en indice `i` des deux tableaux passés en paramètres ;
7. créez une méthode `ordered` qui prend un tableau d'entiers en paramètre et qui retourne `true` si le tableau est trié par ordre croissant ou par ordre décroissant, `false` sinon ;
8. créez une méthode `max` qui prend un tableau d'entiers en paramètre et retourne le maximum du tableau ; et
9. complétez la méthode `main` afin de tester vos méthodes avec les tableaux créés précédemment.

2 Paramètres en ligne de commande et conversion de String

Veuillez créer une classe `ExoCmd`. Voici le comportement que doit avoir votre programme (méthode `main`).

Si l'utilisateur n'a pas entré au moins 2 paramètres (en ligne de commande), le programme doit s'arrêter et afficher un message indiquant qu'il faut au moins 2 paramètres. Dans le cas contraire, on suppose que l'utilisateur a entré uniquement des paramètres qui sont des nombres entiers (précondition). Le programme doit afficher à l'écran la somme et la moyenne des nombres qui ont été passés en paramètres. Vous pouvez vous servir de la classe `ExoArray` si nécessaire.

3 Mémoïsation

La mémoïsation est une technique utilisée dans les algorithmes récursifs lorsqu'un algorithme effectue les mêmes appels récursifs de nombreuses fois. Par exemple, lorsqu'on calcule le 4^e élément de la suite de Fibonacci, si on regarde les appels récursifs effectués (cfr. Figure 1), on se rend compte qu'on a besoin de calculer le 2^e élément deux fois ainsi que ses appels récursifs.

Pour améliorer les performances, on va stocker les résultats des appels récursifs dans un tableau. Ainsi, avant chaque appel récursif, on va d'abord vérifier si la valeur est déjà disponible et on effectue un appel récursif uniquement si la valeur n'est pas disponible.

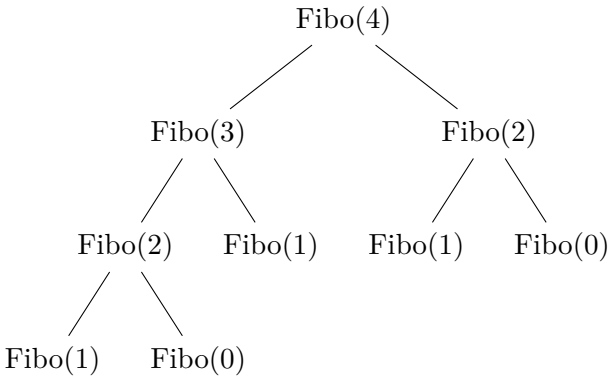

FIGURE 1 – Appels récursifs effectués pour calculer Fibo(4)

Il vous est demandé d'implémenter des fonctions récursives utilisant cette techique pour le problème de la suite de Fibonacci et pour le triangle de Pascal. Inspirez-vous des solutions récursives à ces problèmes que vous aviez implémentées lors de la séance précédente.

4 Classement d'un championnat de football

Reprenez la classe **`Point`** que vous avez créée dans le TP précédent et transformez la pour qu'elle représente des couples d'entiers au lieu de doubles. Ensuite, créez une classe **`FootballScores`** qui instancie un tableau reprenant les résultats des matchs de football d'un championnat. Le constructeur de cette classe prend en paramètre un tableau `teams` de `String` représentant le nom des équipes du championnat. Il crée un tableau à deux dimensions de points (couples) tel qu'en position (i,j) se trouve le résultat du match de l'équipe en position i contre l'équipe en position j dans le tableau `teams` ou `NULL` si le match n'a pas encore été joué. Ajoutez maintenant les méthodes suivantes :

1. `getIndex( )`
   Entrée: une chaine de caractères e.
   Sortie: l'index de l'équipe e dans le tableau `teams`.
2. `setScore( )`
   Entrée: deux chaines de caractères $e1$ et $e2$ et un Point p.

Sortie: /.
Insère dans le tableau des scores, le résultat p du match entre les équipes $e1$ et $e2$.

3. `simulateMatch( )`
Entrée: deux chaines de caractères $e1$ et $e2$.
Sortie: /.
Crée un score aléatoire (maximum 5 buts) pour le match entre les équipes $e1$ et $e2$ et l'insère dans le tableau des scores.
4. `simulateChampionship()`
Entrée: /.
Sortie: /.
Simule le score de tous les matchs du championnat qui n'ont pas encore été joués.

5 Statistiques

Créez une classe **`StatChampionship`** qui va permettre de faire des statistiques à partir d'un tableau de scores des matchs de football d'un championnat. Un constructeur de cette classe prendra donc en paramètre un objet du type `FootballScores`. Ajoutez ensuite les méthodes suivantes :

1. `getMatchsPlayed( )`
Entrée: une chaine de caractères e.
Sortie: le nombre de matchs joués par l'équipe e.
2. `getWins( )`
Entrée: une chaine de caractères e.
Sortie: le nombre de matchs gagnés par l'équipe e.
3. `getDraws( )`
Entrée: une chaine de caractères e.
Sortie: le nombre de matchs nuls de l'équipe e.
4. `getLosses( )`
Entrée: une chaine de caractères e.
Sortie: le nombre de matchs perdus par l'équipe e.
5. `getGoals( )`
Entrée: une chaine de caractères e.
Sortie: le nombre de buts marqués au cours du championnat par l'équipe e.
6. `getConceded( )`
Entrée: une chaine de caractères e.
Sortie: le nombre de buts encaissés au cours du championnat par l'équipe e.
7. `getDifference( )`
Entrée: une chaine de caractères e.
Sortie: la différence entre le nombre de buts marqués et le nombre de buts encaissés au cours du championnat par l'équipe e.
8. `getPoints( )`
Entrée: une chaine de caractères e.
Sortie: le nombre de points obtenus par l'équipe e sachant qu'une victoire vaut 3 points, un match nul 1 point et une défaite 0 point.
9. `printStat()`
Entrée: /.
Sortie: /.
Imprime à l'écran le classement du championnat. Plus précisément, pour chaque équipe (par ordre décroissant des points obtenus), imprimer une ligne contenant : le nom, le nombre de matchs joués, le nombre de victoires, de matchs nuls et de défaites, le nombre de but marqués et encaissés, la différence de buts et les points de l'équipe considérée.