TD 3

Chaines de caractères & Tableaux

Exercice 1 : Nettoyer un formulaire utilisateur

Un utilisateur saisit son nomet prenom avec des espaces ou des caractères inutiles, par exemple « En-Nassiri ahmed ». Pour uniformiser les noms, écrire un programme qui :

- nettoie la chaîne des espaces au début et fin.
- met le nom en majuscule
- met la première lettre du prénom en majuscule et le reste en minuscule
- remplace les – par des espaces

Exercice 2 : Compter les occurrences d'un mot dans un texte

Écrire un programme qui compte combien de fois le mot "java" apparaît dans une phrase.
Ne pas tenir compte de la casse (min/maj).

Exercice 3 : Extraction d'email

On dispose d'une phrase contenant éventuellement une adresse e-mail respectant la forme :

(espace) + (mot alphanumérique) + @ + (mot alphanumérique) + (espace)

Exemples :
"Bonjour ali123@gmail merci"
" contactez user01@test maintenant"

Écrire une méthode extraireEmail(String texte) qui retrouve l'adresse e-mail dans la phrase en utilisant les méthodes vues en cours.

➔ **Indication** : utiliser la methode suivante :

```
public static boolean estAlphaNum(char c) {
        return (c >= 'A' && c <= 'Z') ||
               (c >= 'a' && c <= 'z') ||
               (c >= '0' && c <= '9');
    }
```

Exercice 4 : Classe AnalyseTableau

Ajoutez et implémentez les méthodes suivantes :

1. **frequenceValeur(int[] t, int v)** : retourne le nombre de fois que la valeur v apparaît dans le tableau t.

2. **estSymetrique(int[] t)** : retourne true si le tableau est symétrique (lecture identique de gauche à droite et de droite à gauche).

3. **extraireSousTableau(int[] t, int debut, int fin)** : retourne un sous-tableau contenant les éléments de t entre les indices debut et fin. Affiche une erreur si indices invalides.

4. **sommePairsImpairs(int[] t)** : retourne un tableau de taille 2 contenant la somme des éléments pairs et la somme des éléments impairs du tableau.

5. **fusionEtTri(int[] t1, int[] t2)** : retourne un tableau contenant tous les éléments triés (avec doublons supprimés) en utilisant triSelection.

6. **estPermutation(int[] t1, int[] t2)** : retourne true si les deux tableaux contiennent les mêmes éléments (pas forcément dans le même ordre).

Exercice 5: Classe AnalyseTexte

Ajoutez et implémentez les méthodes suivantes :

1. **nettoyerTexte(String texte)** : retourne une chaîne où tous les caractères spéciaux sont retirés (seuls lettres et chiffres sont conservés).

2. **motLePlusLong(String phrase)** : retourne le mot le plus long dans une phrase.

3. **compterMots(String phrase)** : retourne le nombre de mots dans la phrase (mots séparés par un ou plusieurs espaces).

4. **remplacerVoyelles(String phrase, char remplaçant)** : remplace toutes les voyelles par le caractère donné.

5. **inverserPhrase(String phrase)** : inverse les mots dans la phrase (ex : "je suis étudiant" → "étudiant suis je").

6. **estAnagramme(String mot1, String mot2)** : retourne true si les deux mots sont des anagrammes (mêmes lettres dans un ordre différent).

Exercice 6 : Classe Etudiant

- Attributs :
  - String nom
  - String prenom
  - double[] notes
- Méthodes à implémenter :

1. **moyenne**() : retourne la moyenne des notes.
2. **noteMax**() : retourne la note maximale obtenue.
3. **estAdmis**(double seuil) : retourne true si la moyenne est supérieure ou égale au seuil.
4. **ajouterNote**(double note) : ajoute une note à la première case vide du tableau. Affiche un message si le tableau est plein.
5. **afficherInfos**() : affiche nom, prénom, moyenne, admission (oui/non).
6. **afficherNotesTriees**() : affiche les notes dans l'ordre croissant.

Écrire un petit **menu** console qui permet à l'utilisateur de :

- Créer un étudiant
- Saisir ses notes
- Afficher ses résultats
- Vérifier son admission