Contrôle Continu

Durée: 1h30

Avant de commencer...

Informations générales:

- Barème donné à titre indicatif
- Le respect des conventions de nommage et des principes de POO vus en cours sera pris en compte dans la notation
- Délai avec tolérance de 10 minutes avant et 10 minutes après. Tout retard sera sanctionné !

Consignes :

- Dans Eclipse (ou équivalent), créer un projet **CC_2020_Lundi** dans votre workspace.
- Le sujet fournit des structures et méthodes à compléter que vous devez récupérer depuis le sujet. Tout ajout de méthodes supplémentaires est possible mais devra faire l'objet d'une explication en commentaire dans votre code.
- **Le projet est à rendre sous format zip** depuis Eclipse ou équivalent (`Export` → `General` → `Archive File`) sur Moodle dans l'espace de rendu prévu à cet effet.
- Votre zip doit contenir votre NOM, PRENOM, afin de faciliter l'identification. Merci de l'enregistrer selon le schéma suivant : **NOM_PRENOM_CC_2020_Lundi.zip.**

Exercice 1 (8 points)

Information : Creer un module/dossier exo1, et une classe `UtilString` dans celui-ci. Cette classe contiendra toutes les méthodes de cet exercice (excepté les tests).

1. Ajouter et compléter la méthode suivante :

```java
/**
* Détermine si une chaîne de caractères appartient
* a un tableau de chaînes de caractères
*
* @param str la chaîne
* @param tab le tableau
* @return true si et seulement si str appartient a tab
*/
public static boolean appartientTableau(String str, String[] tab) {
  //votre code ici
}
```

2. Implémenter maintenant la variante suivante:

```java
/**
* Méthode qui détermine si une chaîne de caractères apparaît dans l'un des éléments
d'un tableau de chaînes de caractères
*
* @param str la chaîne
* @param tab le tableau
* @return String contenant l'élément du tableau dans lequel str apparaît dans tab,
reste null sinon.
*/
public static String apparaitDansTableau(String str, String[] tab) {
   //votre code ici
}
```

3. Écrire une méthode statique `estPalindrome(...)` (avec la javadoc) qui détermine si une chaîne d'**au moins deux caractères** est un palindrome, c'est-à-dire est symétrique par rapport à son milieu (**"00100100"** par exemple, car **"0010"** est le *miroir* de **"0100"**) . **Toute chaîne de caractères contenant moins de deux caractères n'est de facto pas un palindrome**.
   - Entrée : Chaine de caractère (`String`)
   - Sortie : Booléen (`true` si palindrome, `false` sinon)

4. Écrire une méthode statique `estPalindromeParMorceaux(...)` (avec la javadoc) qui reconnaît si une chaîne est un palindrome *par morceaux*, c'est-à-dire si elle est la concaténation de plusieurs palindromes (“01010010” par exemple). Même chose, **toute chaîne de caractères contenant moins de deux caractères n’est *de facto* pas un palindrome**.
   -Entrée : Chaine de caractère (`String`)
   -Sortie : Tableau de `String` contenant chacun des palindromes que vous avez trouvé.

5. Écrire une méthode statique `listerPalindrome(...)` qui va regarder si un tableau de chaînes de caractères contient des palindromes parmi ses éléments ou au sein de ses éléments.
   -Entrée : Tableau de `String` à analyser
   -Sortie : Tableau de `String` contenant chacun des palindromes que vous avez trouvé.

6. Dans un fichier **TestExo1.java**, contenant un main comme ci-dessous, écrire un test par méthode que vous avez eu à implémenter dans cet exercice. Par test, il faut entendre au moins deux cas (par exemple, une liste contenant une chaîne de caractère, et une autre n’en contenant pas), de sorte à couvrir l’ensemble des tests de votre méthode.

```
public static void main(String[] args) {
   //Question 1
  System.out.println("Exo 1 - Test NOM_METHODE : " + "...");
   //...
   //Question 2
}
```

Exercice 2: Tour opérateur (10 points)

Information : Créer un module/dossier exo2 qui contiendra toutes les classes de cet exercice. Les dates peuvent être gérées sous forme de chaînes de caractères ou grâce à des structures (Date de java.util par exemple). Dans cet exercice, vous êtes libre de fournir une solution propre et fonctionnelle, dans le respect des consignes, en respectant le **nom exact des classes et attributs** (quand ils sont donnés) et en choisissant judicieusement le type associé.

1. Créer une classe **`Voyage`** qui représente une prestation de voyage. Cette classe contiendra les informations suivantes :
    - **`titre`** du voyage (“Bain de soleil et noix de coco”)
    - **`dateDebut`** (“2021-01-01”)
    - **`dateFin`** (“2021-05-01”)
    - **`prix`** en euros (10000) que doit payer le tour pour organiser une prestation
    - **`prixClient`** en euros (1000) que doit payer le client
    - **`typeHebergement`** ("HÔTEL","AUBERGE",...)
    - **`typeTransport`** (“AVION”,”BUS”,...)
    - **`formule`** (“DEMI-PENSION”,...)
    - **`nombreMaximum`** (15 touristes au maximum)

2. Créer une classe **`Touriste`** qui contiendra des attributs personnels (**`nom, prenom, numTelephone, mail, age`**).

3. Créer une classe **`Tour`** qui contient les informations du Tour opérateur (site en ligne qui possède une adresse , une date de création ainsi qu’une liste de clients référencés et une liste de prestation de voyages. Le tour opérateur décide aussi de stocker les voyages de ses clients. La classe Tour doit contenir une structure permettant de stocker pour chaque client, une liste de voyages.

4. La classe `Tour` doit contenir des méthodes permettant:
    a. D'obtenir le client qui a voyagé le plus par rapport à une année donnée

       `obtenirGrandVoyageur(...)`

    b. D'obtenir le nombre de clients sur un voyage donné.

       `obtenirNombreClientParVoyage(...)`

    c. D'obtenir le client ayant payé le plus sur une année sélectionnée.

       `obtenirMeilleurClient(...)`

    d. D'obtenir une liste de mails de clients qui auraient moins voyagé l'année N par rapport à l'année N-1 (en terme de fréquence de voyage). N et N-1 étant des années à renseigner.

       `obtenirListeMailARelancerFrequence(...)`

    e. D'obtenir une liste de mails de clients qui auraient moins voyagé l'année N par rapport à l'année N-1 (en terme de prix total des prestations consommées). N et N-1 étant des années à renseigner.

       `obtenirListeMailARelancerPrix(...)`

    f. D'obtenir un voyage pour un mois donné, pour lequel le nombre maximum de clients n'est pas atteint.

       `obtenirVoyageAVendreParPeriode(...)`

5. Écrire une classe de Tests, nommée `TestExo2.java`, et tester au minimum 3 méthodes avec différents cas, cinq voyages et trois clients.

Exercice 3: Pour la fin (2 points)

1. Créer un module/dossier exo3 contenant une classe Catalogue. Cette classe contiendra dans un attribut la chaîne de caractères suivante :

```
String catalogue_texte = "Avec un bain de soleil digne de la Nouvelle Zélande, on peut capturer un girafarig au Zoo Safari sans prendre de Xanax"
```

2. Utiliser en les important, une méthode de l'exercice 1 pour trouver les palindromes contenus dans cette `String` .

   Vous pourrez faire cela directement dans un `Main` au sein de cette classe.

3. Importer de l'exercice 2 la classe **Voyage**, en ajoutant au catalogue un attribut contenant une liste de voyages.

■ ■ ■

Devoir à remettre sur Moodle dans l'espace de Rendu :

CC_2020_Lundi_Espace_Rendu

Merci à tous !