IFT1025 – Programmation 2
TP1 : Mini correcteur d'orthographe
Remise : le 1 décembre avant 23:55
À faire en groupe allant jusqu'à 3 personnes
Ce TP correspond à 10% dans la note globale.

But
Ce TP vise à utiliser les concepts suivants : interface graphique, classe, interface, des structures de données prédéfinie, et manipulation des fichiers.

Problème
Ce TP traite le problème de correction d'orthographe. Pour le but de ce TP, on considère une correction très restreinte : Seuls les mots stockés dans un dictionnaire (qu'on peut lire dans un fichier) sont considérés corrects et on ne considère pas la grammaire. Dans un texte entré par un utilisateur ou lu d'un fichier, il peut y avoir des mots qui ne sont pas dans le dictionnaire. Ces mots sont « inconnus ». Le rôle du correcteur est (1) d'identifier ces mots « inconnus », (2) de proposer des mots du dictionnaire les plus proches à l'utilisateur pour le remplacer, (3) de remplacer le mot si une proposition est acceptée. Après la correction, l'utilisateur peut choisir à stocker le résultat dans un fichier.

Tâches
Pour ce TP, vous devez concevoir une interface graphique qui permet à l'utilisateur d'exécuter les tâches suivantes :

1. Lire un dictionnaire (une liste de mots) à partir d'un fichier : ceci va créer une structure de données interne contenant tous les mots.
2. Entrer un texte ou lire un texte d'un fichier;
3. Lancer la vérification d'orthographe qui vérifie si chaque mot est un mot connu. Si non, le mot sera signalé à l'utilisateur (par exemple, surligné en rouge).
4. Si l'utilisateur clique sur un mot en rouge, on doit voir apparaître les mots les plus proches dans un petit menu à côté du mot. L'utilisateur peut alors choisir un des mots proposés pour le remplacer.
5. Finalement, une fois la correction est faite, l'utilisateur peut stocker le texte corrigé dans un fichier.

Interface graphique
Menu :
L'interface graphique que vous programmez doit avoir un menu, dans lequel il y a « Fichier », « Dictionnaire » et « Vérifier ».

- Quand on clique sur le menu « Fichier », on verra apparaître les options « Ouvrir » et « Enregistrer » pour respectivement lire un texte d'un fichier et sauvegarder le texte dans un fichier. Le texte lu sera affiché dans une zone de texte (un TextArea - voir plus bas).
- Quand on clique sur le menu « Dictionnaire », on veut charger un dictionnaire à partir d'un fichier. Ce dictionnaire sera alors stocké dans une structure de données interne (e.g. un tableau, un ArrayList, ou une autre structure de votre choix comme HashSet (voir les exemples à https://www.cs.cmu.edu/~adamchik/15-121/lectures/Hashing/hashing.html). Votre choix de la structure de données doit tenir compte des manipulations que vous devez faire.

  Pour ces deux opérations liées au fichier, vous devez utiliser la classe FileChooser (ou JFileChooser) pour demander le nom du fichier. Voir le tutoriel https://docs.oracle.com/javase/tutorial/uiswing/components/filechooser.html

- Cliquer sur « Vérifier » va lancer la vérification du texte, qui change tous les mots inconnus en rouge.

Zone d'affichage de texte à corriger :
Votre interface doit contenir une zone d'affichage de texte (TextArea ou JTextArea), qui affiche le texte (lu d'un fichier, ou entré directement par l'utilisateur). L'utilisateur peut voir apparaître les mots en rouges après avoir cliqué sur « Vérifier ». Il peut alors cliquer sur un mot en rouge, et votre programme doit lui proposer une liste de 5 mots les plus proches selon la distance d'édition dans un menu à côté (voir l'explication de la distance d'édition plus bas). Si l'utilisateur choisit un de ces mots, le mot inconnu sera remplacé par le mot choisi.

Les opérations sur le texte à corriger :
Pour surligner les mots inconnus, vous aurez besoin d'identifier chaque mot dans le texte, et s'il est inconnu (non existant dans le dictionnaire), de le surligner en rouge (ou utiliser une autre façon appropriée pour le signaler). Un exemple TextAreaHighlight.java, qui surligne un mot (le mot « public ») dans un JTextArea, est fourni. Cet exemple (1) identifie l'index (la position) du mot à surligner, (2) surligne les caractères couvrant la longueur du mot. Vous pouvez vous inspirer de cet exemple pour surligner les mots. Cet exemple utilise l'interface Highlighter (et son implantation par défaut).

Quand l'utilisateur clique sur un mot dans le texte, vous aurez besoin d'identifier le mot en question. Vous pouvez vous inspirer de l'exemple TextAreaTest.java, qui est également fourni. Cet exemple affiche un petit texte, et quand on clique sur un mot, il afficher sur l'écran les coordonnées de la position cliquée. C'est la méthode `public void mouseClicked(MouseEvent e)` qui détecte le mot sur lequel l'utilisateur a cliqué. Pour ce faire, l'exemple (1) détermine la position du click, (2) détermine l'offset de la position dans le texte, et (3) détermine le mot correspondant. Vous pouvez vous inspirer de cet exemple pour identifier le mot cliqué.

Pour modifier un mot inconnu, vous aurez besoin de la méthode `void replaceRange (String, int, int)` de la classe TextArea (voir un tutoriel http://java.sun.com/docs/books/tutorial/uiswing/components/textarea.html). Remarquer qu'il est possible de remplacer un mot par un autre mot de longueur différente (même vide).

Les mots les plus proches
Pour déterminer les mots du dictionnaire les plus proches à un mot inconnu, on utilise la distance d'édition (edit distance en anglais), ou la distance Levenshtein (https://fr.wikipedia.org/wiki/Distance_de_Levenshtein). Cette distance correspond au nombre d'insertion, de remplacement et d'enlèvement de caractère pour transformer un mot en un autre. Par exemple, la distance d'édition entre « extraction » et « axtrations » est 3 : 1 remplacement (e par a), 1 enlèvement (c) et 1 insertion (s). Les mots les plus proches sont les mots du dictionnaire ayant la distance minimale avec le mot inconnu. Dans ce TP, vous faîtes afficher 5 mots les plus proches. Il peut y avoir plusieurs mots de même distance. Dans ce cas, vous pouvez choisir à afficher n'importe quels candidats de distance équivalente.

Vous pouvez trouver une implantation de cette distance : https://gist.github.com/gabhi/11243437
Vous pouvez utiliser cette implantation directement dans votre TP.

Conception et organisation
Vous êtes demandé à concevoir vos programmes vous-mêmes, en exploitant le plus possible les outils existants (les classes de Java et les structures de données prédéfinies). Vous pouvez utiliser

des outils ou des classes qui ne sont pas présentés dans la classe. Vous devez chercher la documentation sur le Web.
La configuration de l'interface graphique présentée plus haut est juste une suggestion. Si vous jugez qu'il y a une autre configuration ou une autre manière de faire la correction plus appropriée, vous pouvez l'utiliser. Cependant, toutes les fonctions décrites doivent être réalisées.
La structure MVC pour l'interface graphique n'est pas exigée, mais conseillée. Vous devez penser à bien structurer vos programmes en des classes. Cet aspect est pris en compte dans la correction.

Classes
Vous aurez besoin de déclarer des classes pour gérer différents éléments.

- Notamment, vous devez déclarer une classe Dictionnaire. Cette classe doit contenir une structure de données, contenant l'ensemble de mots connus. La classe doit aussi avoir une méthode pour permettre de trouver les 5 mots les plus proche à un mot en entrée. La lecture du dictionnaire à partir d'un fichier est aussi faite par une méthode dans cette classe. Vous pouvez ajouter d'autres méthodes ou attributs que vous jugez utiles.
- Déclarer toute autre classe que vous jugez nécessaires.

À remettre

- Vous devez rendre un fichier zip, créé par l'export de votre projet depuis Eclipse.
- Vous devez aussi inclure un fichier d'explication « readme » qui donne une brève description de votre interface graphique (comment l'utiliser), votre organisation des classes, et tout d'autres informations que vous pensez utiles. Rappelez-vous que votre programme sera corrigé par une autre personne qui doit comprendre ce que vous avez fait. C'est comme si votre programme doit être utilisé par un autre programmeur. La difficulté pour comprendre votre programme va vous pénaliser.

Dans vos programmes, pensez aussi à ajouter des commentaires, sans toutefois surcharger et en mettre trop.

Évaluation
Ce TP compte pour 10% dans la note finale. La correction se base sur les éléments suivants :

- Conception et affichage de l'interface graphique, activation de menus et interactions de l'utilisateur sur le texte à corriger : 3 points
- Lecture du dictionnaire et le stockage dans une structure interne : 2 points
- Lecture d'un fichier texte à corriger et identification et surlignement des mots inconnus dans le texte : 2 points
- Correction d'un mot inconnu (afficher les mots proches et remplacement du mot inconnu) : 2 points
- Organisation globale de votre programme (des classes) et les commentaires nécessaires : 1 point

La date limite de remise est avant 23:55 le 1 décembre. Mais vous pouvez remettre avant cette date, bien entendu. La remise après la date limite est tolérée, mais avec 1 point (sur 10) de pénalité par jour de retard.
Pour encourager la collaboration, vous êtes demandés à travailler en groupe de 2-3 personnes.