ACADÉMIE
DE REIMS
Liberté
Égalité
Fraternité

NUMERIQUE
et
SCIENCES
INFORMATIQUES

Épreuve de l'enseignement de spécialité

Sujet d'entraînement

Partie Pratique

Classe Terminale de la voie générale

Le candidat doit traiter les 2 exercices

Ce sujet comporte 4 pages

Exercice 1 (4 points)

Un palindrome est un mot qui se lit lettre par lettre, de gauche à droite, exactement comme de droite à gauche.

Exemples :
“elle”, “kayak”, “ressasser” ou “yopkpoy” sont des palindomes.

On peut construire un palindrome à partir d’une chaîne de caractères en concaténant la châine à sa chaîne renversée.

Écrire une fonction `creer_palindrome` qui prend en paramètre une chaine de caractères `mot` non vide et qui renvoie le palindrome créé en concaténant `mot` suivi par le renversement de `mot`.
Rajouter une documentation claire ainsi que deux préconditions.

Exemples :

```
>>> creer_palindrome("OK")
    "OKKO"
>>> creer_palindrome("INGIRUMIMUSNOCTE")
    "INGIRUMIMUSNOCTEETCONSUMIMURIGNI"
```

Attention, la fonction `reversed` tout comme le slicing sur des chaînes de caractères (`:` ou tranche en français) sont interdits dans cet exercice.

Exercice 2 (4 points)

Dans cet exercice, l'utilisation de la méthode `sort` et des fonctions `sorted` et `mean` sont interdites.

Vous travaillez pour une organisation féministe qui cherche à évaluer les inégalités salariales liées au genre dans différentes entreprises salariant au moins un homme et au moins une femme.
Une personne travaillant dans cette même organisation a déjà réussi à implémenter les informations pour chaque entreprise sous forme d'une liste de tuples `("nom",salaire,genre)` tuples qui précise :

- le nom `"nom"` de chaque employé.e,
- le nombre entier `salaire` qui correspond au salaire mensuel moyen arrondi à l'euro près de cet.te employé.e,
- la chaîne de caractères `genre` donnant le genre, `"féminin"`ou `"masculin"`, de la personne.

Voici, par exemple, les listes associées respectivement aux entreprises 'Pere en tris', 'Repentires', 'Serpentier' et 'Presentire' :

```
>>>pere_en_tris = [('Alhimi', 2343, 'masculin'), ('Allamal', 2009, 'masculin'),
                   ('Bovoire', 2615, 'masculin'), ('De pisan', 2405, 'féminin'),
                   ('Diallo', 2194, 'féminin'), ('Dévice', 2423, 'masculin'),
                   ('Gouges', 1793, 'féminin'), ('Parques', 1955, 'masculin'),
                   ('Punkharst', 1903, 'masculin'), ('Stanton', 2345, 'féminin'),
                   ('Veille', 2501, 'masculin'), ('Zetkin', 1910, 'féminin')]
>>>repentires = [("Ohm",1789,"masculin"),("Fame",1980,"féminin"),("Rezist-Hans",1940,"féminin")]
>>>serpentier=[("celaume",1687,"masculin")]
>>>presentire=[("Phame",1975,"féminin"),("Fémaine",2008,"féminin")]
```

Une collaboratrice de l'organisation a commencé à écrire une fonction `ecart_salarial` qui prend en paramètres une liste de tuples comme précédemment et qui renvoie l'écart entre la moyenne des salaires masculins de l'entreprise avec la moyenne de ceux féminins.
Vous remarquez que le fonction `ecart_salarial` ébauchée suppose que la liste saisie comme argument doit déjà être triée par le genre : toutes les femmes salariées doivent apparaître avant tous les hommes.
De plus, le code sous-entend que la liste doit comporter au moins une femme et au moins un homme.

Il faut donc créer aussi une procédure `discriminer` qui prend en paramètre une liste de tuples comme ci-dessus, qui sépare les femmes des hommes en plaçant les femmes au début de liste et les hommes en fin de liste.

Pour cette procédure `discriminer`, on vous propose de suivre l'algorithme suivant :

- à chaque étape du tri, le tableau est constitué de trois zones consécutives :
  - la première ne contient que des tuples liés aux femmes, c'est-à-dire dont le dernier élément est `"féminin"`,
  - la seconde n'est pas encre triée,
  - la dernière ne contient que des tuples liés aux hommes, c'est-à-dire dont le dernier élément est `"masculin"`.

Ci-dessous un schéma de la situation pendant le processus de séparation des femmes et des hommes :

```
  début de la zone non triée              fin de la zone non triée
                             |            |
                             v            v
 ------------------------------------------------------------------------
| "féminin" ... "féminin" | zone non triée | "masculin" ... "masculin" |
 ------------------------------------------------------------------------
```

- Tant que la zone non triée n'est pas réduite à un seul tuple, on regarde son premier tuple :
  - si ce tuple a pour dernier élément `"féminin"`, on considère qu'il appartient désormais à la première zone ne contenant que des femmes,
  - si ce tuple a pour dernier élément `"masculin"`, il est échangé avec le dernier tuple de la zone non triée et on considère alors qu'il appartient à la dernière zone ne contenant que des hommes. Dans tous les cas, la longueur de la zone non triée diminue de 1.

Compléter le code ci-dessous commencé par votre ami puis lancer une partie proposant dix essais :

```
from random import choice

def discriminer(entreprise:list):
    """entreprise est supposé être une liste de tuple de type ("nom",salaire,"genre").
    Cette fonction place tous les éléments de genre "féminin" de entreprise à gauche
    et tous les éléments de genre "masculin" à droite"""
    debut = ...  # indice de début
    fin = ...    # indice de fin
    while debut ...:
        if ... == "féminin":
            debut = ...
        else:
            # permutation :
            temp = ...
            ...
            ... = temp
            fin = ...

def ecart_salarial(entreprise_separee:list)->int:
    """entreprise_separee est supposée être une liste de tuple du type ("nom",salaire,"genre")
    telle que tous les éléments de genre "féminin" sont placés avant ceux de genre "masculin"
    Fonction qui renvoie l'écart entre la moyenne des salaires masculins et la moyenne de ceux féminins.
    """
    assert entreprise_separee[0][2] == "féminin", "l'entreprise considérée doit employer au moins une femme"
    assert ..., "l'entreprise considérée doit employer au moins une homme"
    somme_sal_femme = 0
    somme_sal_homme = 0
    i = 0
    while ...:
        somme_sal_femme += ...
        i += 1
    nb_femmes = ...
    while i<len(entreprise_separee):
        somme_sal_homme += ...
        i += 1
    return somme_sal_homme/(len(entreprise_separee)-nb_femmes)-somme_sal_femme/nb_femmes
```

Exemples :

```
>>> discriminer(pere_en_tris)
>>> pere_en_tris
    [('Zetkin', 1910, 'féminin'), ('Stanton', 2345, 'féminin'), ('Gouges', 1793, 'féminin'),
    ('De pisan', 2405, 'féminin'), ('Diallo', 2194, 'féminin'), ('Dévice', 2423, 'masculin'),
    ('Parques', 1955, 'masculin'), ('Punkharst', 1903, 'masculin'), ('Bovoire', 2615, 'masculin'),
    ('Veille', 2501, 'masculin'), ('Allamal', 2009, 'masculin'), ('Alhimi', 2343, 'masculin')]
>>> ecart_salarial(pere_en_tris)
    120.45714285714257
>>> discriminer(repentires)
>>> repentires
    [('Fame', 1980, 'féminin'), ('Ohm', 1789, 'masculin'),,("Rezist-Hans",1940,"féminin")]
>>> ecart_salarial(repentires)
    -171.0
```