ICC: Programmation — Exercices supplémentaires sur les collections

1. Créez une **`dataclass`** **`Contact`** qui stocke le prénom, le nom de famille, le genre, l'âge et la taille d'une personne.

2. Créez ensuite une liste de contacts. Vous pouvez utiliser ces données ci-dessous:

```
Contact("Josh",  "Lyman",   False, 42, 1.86)
Contact("Toby",  "Ziegler", False, 51, 1.78)
Contact("Donna", "Moss",    True,  25, 1.72)
Contact("Jed",   "Bartlet", False, 56, 1.75)
Contact("Zoey",  "Bartlet", True,  19, 1.69)
Contact("Leo",   "McGarry", False, 62, 1.77)
Contact("C. J.", "Cregg",   True,  43, 1.82)
Contact("Sam",   "Seaborn", False, 39, 1.71)
```

3. Affichez tous les contacts dans l'ordre de création ci-dessus.

4. Écrivez une fonction **`reverse()`** qui accepte une liste de contacts et renvoie une nouvelle liste dans laquelle on retrouve tous les contacts de la liste donnée, mais dans l'ordre inverse. Utilisez-la pour afficher de nouveau tous les contacts, mais dans l'ordre inverse.

5. Écrivez une fonction **`filter_by_gender()`** qui accepte une liste de contacts et un `bool` indiquant le genre à filter. Si le boolean est **`True`**, cette fonction renvoie une nouvelle liste avec uniquement les femmes de la liste passée en paramètre. Sinon, une nouvelle liste avec uniquement les hommes. Utilisez cette fonction pour obtenir une liste avec seulement les hommes et une autre liste avec seulement les femmes.

6. Écrivez une fonction **`calculate_average_age()`** qui renvoie l'âge moyen de tous les contacts qu'on lui donne en paramètre dans une liste. Utilisez-la pour afficher l'âge moyen de tous vos contacts, puis de toutes les femmes de votre répertoire (comme retourné par la fonction **`filter_by_gender()`**) et l'âge moyen de tous les hommes (même chose).

7. Écrivez une fonction **`calculate_maximum_height()`** qui renvoie la taille maximale des contacts passés en paramètre dans une liste. Affichez la taille maximale de tous les contacts, puis des femmes seulement, puis des hommes seulement.

8. Écrivez une fonction **`count_first_names_of_length()`** qui accepte un **`int`** et une liste de contact, et renvoie un **`int`** qui est le nombre de contacts qui possèdent un prénom de la longueur donnée. Dans une boucle, appelez cette fonction pour connaître le nombre de vos contacts qui ont 3, 4 ou 5 lettres dans leur prénom.

9. Écrivez une fonction **`gather_first_names_alphabetically()`** qui renvoie une liste triée de tous les prénoms des contacts passés en paramètre dans une liste. Utilisez-la pour afficher tous les prénoms dans l'ordre alphabétique.

10. Écrivez une fonction **`build_letter_frequency_dict()`** qui accepte une liste de contacts et qui renvoie un **`Dict[str, int]`** qui relie chacun des caractères au nombre de fois qu'il apparaît dans tous les prénoms et noms de famille des contacts. On ne veut pas faire pas la différence entre un caractère minuscule et un caractère majuscule.

11. Écrivez une fonction **`find_most_often_used_character()`** qui accepte un **`Dict[str, int]`** telle que retournée par la fonction **`build_letter_frequency_dict()`** et renvoie le caractère du dict qui est lié au **`int`** le plus grand, c'est-à-dire au caractère qui arrive le plus souvent. Appelez cette fonction en utilisant le résultat du point précédent et affichez le résultat.

12. Écrivez une fonction **`group_by_last_name()`** qui accepte une liste de contact et qui renvoie un **`Dict[str, Set[str]]`** où chaque entrée dans le dict relie un nom de famille à l'ensemble de tous les prénoms qui ont ce nom de famille. Affichez le résultat.