

## EXAMEN 2019-2020

Durée : 3h

*Tous les documents sont autorisés, mais l'accès à internet est strictement interdit (sauf peut-être sur la page web du professeur). A l'issue de l'examen, 11h45 au plus tard, le fichier source .py, commenté si possible, devra être envoyé par mail à l'adresse brice.mayag@dauphine.fr pour correction. Tout retard entraînera une lourde pénalité. Les étudiants veilleront à éteindre leur téléphone portable durant l'épreuve.*

### Exercice 1 : Les nombres

- On appelle **diviseur propre** d'un entier naturel  $n$ , un diviseur quelconque de  $n$ , excepté  $n$ . *Exemple : 2 est un diviseur propre de 6.*
- On appelle **nombre abondant** tout entier naturel non nul  $n$  qui est strictement inférieur à la somme de ses diviseurs propres.

- *Exemple : 12 est donc un nombre abondant.*

En effet,

- \* Les diviseurs propres de 12 sont 1, 2, 3, 4 et 6.
- \* La somme  $1 + 2 + 3 + 4 + 6$  donne 16.
- \* Et 16 est supérieur à 12.

- *Par contre, 10 n'est pas un nombre abondant.*

En effet,

- \* Les diviseurs propres de 10 sont 1, 2, et 5.
- \* La somme  $1 + 2 + 5$  donne 8.
- \* Or 8 est inférieur à 10.

- Deux nombres (entiers strictement positifs) sont dits **amicaux** s'ils sont distincts et si chacun des deux nombres est égal à la somme des diviseurs propres de l'autre.
- *Les nombres entiers 220 et 284 sont amicaux.*

En effet,

- \* Les diviseurs propres de 220 sont 1, 2, 4, 5, 10, 11, 20, 22, 44, 55 et 110 ;
- \* La somme  $1 + 2 + 4 + 5 + 10 + 11 + 20 + 22 + 44 + 55 + 110$  donne 284.
- \* Les diviseurs propres de 284 sont 1, 2, 4, 71 et 142 ;
- \* La somme  $1 + 2 + 4 + 71 + 142$  donne 220.

1. Écrire une fonction python `estAbondant(n)` qui indique si un nombre entier  $n$  est Abondant.
2. Écrire une fonction python `listeNombreAbondant(100)` qui retourne la liste des nombres abondants inférieurs ou égaux à 100.
3. Écrire une fonction python `estAmicaux(n, m)` qui indique si deux nombres entiers  $n$  et  $m$  sont amicaux.

## Exercice 2 : Le jury de classe

On considère le tableau ci-dessous représentant les notes de huit étudiants sur trois matières :

|             | 1 : Maths (M) | 2 : Physique (P) | 3 : Langues (L) |
|-------------|---------------|------------------|-----------------|
| Solidor     | 14            | 11               | 17              |
| Baker       | 12            | 9                | 18              |
| Colette     | 8             | 7                | 19              |
| Mistinguett | 10            | 18               | 5               |
| Alain       | 13            | 17               | 14              |
| Prado       | 6             | 16               | 10              |
| Gabin       | 18            | 15               | 8               |
| Olivia      | 11            | 12               | 16              |

Les évaluations par matière sont données sur une échelle de 0 à 20. Les ECTS (crédits ou poids) associés à chaque matière sont donnés par le tableau suivant :

|      | 1 : Maths (M) | 2 : Physique (P) | 3 : Langues (L) |
|------|---------------|------------------|-----------------|
| ECTS | 6             | 5                | 4               |

1. Construire un dictionnaire `Etudiants` contenant les informations des huit étudiants (nom, notes par matière).

`Etudiants` sera donc un dictionnaire de dictionnaires. Par exemple, le dictionnaire associé aux évaluations de Baker sera obtenu en tapant :

```
>>> Etudiants['Baker']
{'M': 12, 'P': 9, 'L': 18}.
```

2. Écrire une fonction python `notefinale` qui calcule la moyenne pondérée d'un étudiant en tenant compte des ECTS.

Par exemple, pour l'étudiant Baker, la fonction `notefinale` renverra le nombre 12.6. :

```
>>> notefinale(Etudiants['Baker'], ECTS)
12.6
```

3. Écrire une fonction Python `Classementfinal` qui retourne la liste ordonnée des étudiants avec leur note finale obtenue à la question précédente.

4. Écrire une fonction Python `ClassementFinalsansECTS` qui retourne la liste ordonnée des étudiants, avec leur note finale obtenue lorsque les ECTS associés aux trois matières sont égaux.
5. Écrire une fonction Python `ClassementFinalMention` qui retourne la liste ordonnée des étudiants avec leur mention attribuée suivant les règles suivantes :

- Mention **Bien** : Note finale  $\geq 14$  ;
- Mention **Assez Bien** :  $14 > \text{Note finale} \geq 12$  ;
- Mention **Passable** :  $12 > \text{Note finale} \geq 10$  ;
- Mention **Insuffisant** : Note finale  $< 10$  ;

La note finale est obtenue à partir des ECTS ci-dessus.

6. On souhaite à présent comparer les étudiants entre eux sans calculer une moyenne. Pour cela, on adopte la règle suivante :

Un étudiant  $x$  “*bat*” un étudiant  $y$  si le total d’ECTS des matières pour lesquelles l’étudiant  $x$  a une note au moins égale à celle de l’étudiant  $y$  est supérieur à 8.

Dans notre exemple, l’étudiante Solidor “*bat*” l’étudiante Baker. En effet,

- \* En Maths, Solidor a une note de 14/20, meilleure que celle de 12/20 obtenue par Baker ;
- \* En Physique, Solidor a une note de 11/20, meilleure que celle de 9/20 obtenue par Baker ;
- \* En Langues, Baker a une note de 18/20, meilleure que celle de 17/20 obtenue par Solidor ;
- \* Le total des ECTS des matières favorables à Solidor (Maths et Physique) par rapport à Baker est  $6 + 5 = 11 > 8$  ;
- \* Le total des ECTS des matières favorables à Baker (Langues) par rapport à Solidor est 4 ;
- \* On jugera donc Solidor meilleure que Baker.

- (a) Écrire une fonction Python `compare(etu1, etu2)` qui renvoie `True` ou `False` si l’étudiant `etu1` “*bat*” l’étudiant `etu2`.
- (b) Écrire une fonction Python `NombreEtudiantBattusParcompare(etu1)` qui renvoie le nombre total d’étudiants “*battus*” par l’étudiant `etu1`.
- (c) Écrire une fonction Python `ClassementFinalOrdinal` qui renvoie une liste ordonnée des étudiants en fonction du nombre total d’étudiants battus par chacun d’entre eux.
- (d) Le jury adopte, en plus, la règle suivante :

Dans une comparaison entre deux étudiants  $x$  et  $y$ , si  $x$  bat  $y$ , alors on attribut 2 points à  $x$  et  $-1$  points à  $y$ .

Écrire une fonction Python `ClassementFinalPoints` qui renvoie une liste ordonnée des étudiants en fonction du nombre total de points obtenus après toutes les comparaisons effectuées.