Asie – 2025 – sujet2 - Correction

Exercice 1 (6 points)

Partie A – Accès et modification des données

1. Compléter le constructeur `__init__` (lignes 3 à 5)

Réponse :

```
self.jour = jour
self.mois = mois
self.annee = annee
```

*Commentaire : Le constructeur initialise les **attributs de l'instance** à partir des paramètres reçus.*
L'utilisation de `self` est indispensable pour rendre les attributs accessibles aux autres méthodes.

---

2. Date correspondant à `d = Date(1, 5, 2000)`

👉 **1er mai 2000**

Commentaire : Les paramètres sont passés dans l'ordre `(jour, mois, année)`.

---

3. Créer une instance représentant le 19 juin 2024

```
d = Date(19, 6, 2024)
```

Commentaire : On appelle simplement le constructeur avec les valeurs numériques de la date.

---

4. Méthode `get_annee` (lignes 15 et 16)

```
def get_annee(self):
    return self.annee
```

Commentaire : Une méthode getter *permet d'accéder à un attribut sans le modifier.*

---

5. Méthode `set_mois` (lignes 21 et 22)

```
def set_mois(self, mois):
    self.mois = mois
```

Commentaire : Une méthode `setter` *modifie la valeur d'un attribut de l'instance courante.*

---

6. Ajustement du mois de février pour les années bissextiles

Code donné :

```
self.nb_jours_par_mois = [31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30,
31]
```

Réponse (lignes 7 et 8) :

```
if self.est_bissextile():
    self.nb_jours_par_mois[1] = 29
```

Commentaire :

- *Février correspond à l'indice **1** (indexation à partir de 0)*
- *On adapte la structure de données dès l'initialisation*

Partie B – Sur l'année de l'instance courante

7. Méthode `est_bissextile`

```
def est_bissextile(self):
    if (self.annee % 400 == 0) or (self.annee % 4 == 0 and
self.annee % 100 != 0):
        return True
    else:
        return False
```

Commentaire : Cette condition correspond exactement à la définition officielle d'une année bissextile.

---

8. Affichage de `d1.nb_jours_passes()`

```
d1 = Date(20, 3, 2001)
d1.nb_jours_passes()
```

Calcul :

- Janvier : 31 jours
- Février 2001 : 28 jours (année non bissextile)
- Mars : 20 jours

👉 Total : **31 + 28 + 20 = 79**

Réponse :

```
79
```

*Commentaire : La méthode compte les jours **écoulés depuis le début de l'année**, jour inclus.*

---

9. Méthode `nb_jours_restants`

```
def nb_jours_restants(self):
    j = 365
    if self.est_bissextile():
        j = 366
    return j - self.nb_jours_passes()
```

Commentaire :

- *On choisit le nombre total de jours selon l'année*
- *On soustrait les jours déjà écoulés*

Partie C – Entre deux dates

10. Résultats de `nb_jours_depuis`

Dates :

```
d1 = Date(15, 6, 2024)
d2 = Date(15, 6, 2024)
d3 = Date(15, 7, 2024)
d4 = Date(15, 6, 2025)
d5 = Date(15, 6, 2022)
```

⚠️ On précise : **2024 est bissextile**

---

a) `d1.nb_jours_depuis(d2)`

Réponse :

```
0
```

Commentaire : Même date → aucun jour écoulé.

---

b) `d1.nb_jours_depuis(d3)`

Du 15 juin au 15 juillet 2024 → **30 jours**

Réponse :

```
-30
```

*Commentaire : La date `other` est **postérieure**, donc le résultat est négatif.*

---

c) `d1.nb_jours_depuis(d4)`

Du 15/06/2024 au 15/06/2025 :

- Fin 2024 (année bissextile) : 366 − jours passés au 15 juin = 200 jours
- Début 2025 jusqu'au 15 juin : 166 jours

👉 Total : **366**

Réponse :

```
-366
```

---

d) `d1.nb_jours_depuis(d5)`

Du 15/06/2022 au 15/06/2024 :

- 2022 → 365 jours
- 2023 → 365 jours

👉 Total : **730**

Réponse :

```
730
```

*Commentaire : Ici, la date `other` est **antérieure**, donc le résultat est positif.*

---

11. Méthode `timestamp`

```
def timestamp(self):
    d = Date(1, 1, 1970)
    return self.nb_jours_depuis(d) * 24 * 3600
```

Commentaire :

- Le timestamp est calculé à partir du **1er janvier 1970**
- On convertit les jours en secondes (24 × 3600)