Exercices POO
Correction

Christophe Viroulaud

Terminale - NSI

Lang 02

Code

L'ensemble des programmes se trouvent ici.

Sommaire

Exercice 1

```
1  class Livre:
2
3      def __init__(self, t: str, a: str, p: int):
4          self.titre = t
5          self.auteur = a
6          self.prix = p
7
8      def get_titre(self) -> str:
9          return self.titre
```

```
1     def afficher(self) -> str:
2         # return "{} est écrit par {}.".
      format(self.titre, self.auteur)
3         return f"{self.titre} est écrit par {
      self.auteur}."
```

```
1 livre1 = Livre("La guerre des mondes", "
     Herbert Wells", 7)
2 print(livre1.get_titre())
```

Sommaire

Exercice 2


```
1  class Rectangle:
2
3      def __init__(self, L: float, l: float):
4          self.longueur = L
5          self.largeur = l
```

```
1     def get_largeur(self) -> float:
2         return self.largeur
3
4     def set_largeur(self, l: float) -> None:
5         self.largeur = l
```

Sommaire

Exercice 3


```
1  class Complexe:
2
3      def __init__(self, re: float, im: float):
4          self.a = re
5          self.b = im
6
7      def addition(self, z) -> tuple:
8          return (self.a + z.a, self.b + z.b)
```

Sommaire

Exercice 4


```
1  class Date:
2      def __init__(self, j: int, m: int, a: int):
3          self.jour = j
4          self.mois = m
5          self.annee = a
```

```
1  def est_avant(self, d) -> bool:
2      # Le \ permet d'écrire sur plusieurs
       lignes
3      # and est prioritaire devant or
4      return self.annee < d.annee or \
5          self.annee == d.annee and \
6              (self.mois < d.mois or \
7              self.mois == d.mois and \
8                  self.jour < d.jour)
```

```
1 def afficher(self) -> str:
2     nom_mois = ["janvier", "février", "mars",
      "avril", "mai", "juin", "juillet", "août"
     , "septembre", "octobre", "novembre", "dé
     cembre"]
3     return f"{self.jour} / {nom_mois[self.
     mois - 1]} / {self.annee}"
```

Sommaire

Exercice 5


```
1  class Loto:
2
3      def __init__(self, num: list, c: int):
4          self.numeros = num
5          self.complementaire = c
```

```
1     def est_gagnant(self, mes_num: list,
      mon_compl: int) -> bool:
2         if mon_compl != self.complementaire:
3             return False
4         i = 0
5         while i < 6 and self.est_present(
      mes_num[i]):
6             i += 1
7         return i == 6
```

```
1  def creer_tirage() -> Loto:
2      """
3      crée un tirage avec des entiers distincts
4      """
5      numeros = []
6      while len(numeros) < 6:
7          n = randint(1, 49)
8          if n not in numeros:
9              numeros.append(n)
10
11     complementaire = randint(1, 49)
12     while complementaire in numeros:
13         complementaire = randint(1, 49)
14
15     return Loto(numeros, complementaire)
```