

Exercice 1 - Analyse de code

Tu viens d'être nommé développeur dans une entreprise qui gère les systèmes d'alarmes pour les maisons des particuliers. Ton prédécesseur a commencé à écrire une classe `Alarme`, qui doit normalement répondre aux exigences suivantes :

- l'alarme peut être activée et désactivée;
- chaque intrusion détectée doit être systématiquement consignée dans un journal;
- en cas d'intrusion, si l'alarme est activée, un SMS doit être envoyé au centre de télésurveillance.

Voici son code :

code Python

```
1 from utils import date, envoie_sms
2
3 ACTIVE = True
4 INACTIVE = False
5
6
7 class Alarme:
8     def __init__(self, lieu, telephone, etat):
9         self.lieu = lieu
10        self.telephone = telephone
11        self.active = etat
12        self.journal = []
13
14    def intrusion(self):
15        evenement = date() + " Intrusion"
16        if self.active:
17            sms = self.lieu + ' : ' + evenement
18            envoie_sms(self.telephone, sms)
19            evenement = evenement + " envoi sms au " + self.telephone
20            self.journal.append(evenement)
21
22    def activer(self):
23        self.active = ACTIVE
24        evenement = date() + " Activation "
25        self.journal.append(evenement)
26
27    def desactiver(self):
28        self.active = INACTIVE
29        evenement = date() + " Désactivation "
30        self.journal.append(evenement)
```

Le module `utils` permet d'importer les fonctions suivantes :

La fonction `date`

- ne prend rien en entrée;

- renvoie la date et l'heure sous la forme d'un `str`.

La fonction `envoie_sms`

- prend en entrée `numero` et `sms`, qui sont deux `str`, et envoie le sms au numéro du centre de télésurveillance.
- renvoie `True` si l'envoi est réussi, `False` sinon.

Pour ce devoir, on va considérer que la fonction `date` renvoie

- `'0000'` au premier appel;
- `'0001'` au deuxième appel;
- et ainsi de suite.

1. On exécute le script suivant :

Code Python

```
from alarme import Alarme

alarme1 = Alarme("Loritz", "971971971", False)
alarme2 = Alarme("Poincaré", "971971971", True)
alarme1.intrusion()
alarme1.activer()
alarme1.intrusion()
alarme1.desactiver()
alarme2.intrusion()
```

Donner les contenus des SMS envoyés.

2. Que contient `alarme1.journal` à la fin du script?

3. En testant le code, on constate que lorsque l'alarme est désactivée les intrusions ne sont pas enregistrées dans le journal.

D'où vient l'erreur? Proposer une correction.

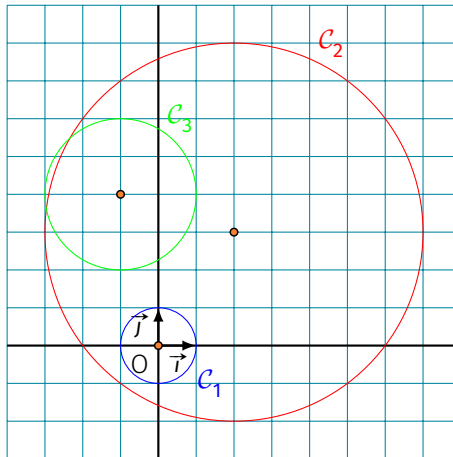
4. On constate que le journal de bord consigne parfois des envois de SMS que le centre n'a jamais reçus : le système a bien tenté de les envoyer mais l'envoi a échoué.

Proposer une correction du code telle que si l'envoi d'un SMS échoue, cela soit consigné dans le journal.

5. Écrire une méthode d'instance `efface_journal` qui efface le journal de l'instance.

Exercice 2 - Cercles

On considère les 3 cercles suivants :



Écrire la classe `Cercle` pour qu'on puisse s'en servir de la sorte :

Code Python

```
from Cercle import *

c1 = Cercle(0, 0, 1) # crée un cercle c1 de centre (0,0) et de rayon 1
c2 = Cercle(3, 1, 5) # idem
c3 = Cercle(-1, 4, 1) # idem

c2.decale(-1, 2) # décale le centre de c2 de (-1,2)
c3.dilate(2) # multiplie le rayon de c3 par 2

print(c1) # affiche correctement c1
print(c2) # idem
print(c3) # idem

print(c2.contient(c1)) # va afficher True car c1 est « à l'intérieur de c2 »
print(c2.contient(c3)) # False
print(c1.chevauche(c3)) # False car c1 et c2 n'ont aucun point commun
print(c2.chevauche(c3)) # True
```

Résultat :

```
Cercle : centre(0, 0) rayon 1.
Cercle : centre(2, 3) rayon 5.
Cercle : centre(-1, 4) rayon 2.
True
False
False
True
```

On suppose que tu as à ta disposition la fonction `distance` qui

- prend en entrée 4 `float` `xa`, `ya`, `xb` et `yb` qui sont les coordonnées de points A et B;
- renvoie la distance AB.