1 Exercices de connaissance du cours

1.1 Syntaxe des expressions conditionnelles

Parmi les expressions suivantes, indiquer celles qui contiennent une erreur de syntaxe et proposer une correction.

a.
```
if x < 0:
      y = 1
  else x >= 0:
      y = 2
```

b.
```
if x < 0:
      y = 1
      else:
         y = 2
```

c.
```
if x < 0:
      y = 1
  elif x >= 10:
      y = 2
```

1.2 Sémantique des instructions conditionnelles

1. Quelles valeurs renvoient les appels de fonctions suivants ?

1. `fonction1(2)`
2. `fonction1(0)`
3. `fonction1(5)`
4. `fonction2(2)`
5. `fonction2(0)`
6. `fonction2(5)`
7. `fonction3(5)`
8. `fonction3(0)`

avec les définitions de fonction suivantes :

```
1 def fonction1(a):
2    res = 0
3    if a >= 4:
4       res = res + 2
5    if a >= 1:
6       res = res + 4
7    return res
```

```
1 def fonction2(a):
2    if a >= 4:
3       res = 2
4    elif a >= 1:
5       res = 4
6    else:
7       res = 6
8    return res
```

```
1 def fonction3(a):
2    if a >= 4:
3       res = 2
4    return res
```

2 Récriture de code, bonnes pratiques

Pour les besoins de la mise en page, la documentation des fonctions n'est pas donnée.

2. Les fonctions suivantes ne tiennent pas compte des bonnes pratiques de programmation enseignées dans l'UE. L'objectif est de les récrire. Vous pouvez raturer en rouge le sujet de TD pour qu'il apparaisse clairement que ces fonctions montrent un exemple **à ne pas suivre**.

```
def m(x, y):
    if x + y >= 10:
       return True
    else:
       return False
```

```
def g(x, y, b):
    if b == True:
        return - x - y
    else:
        return x + y
```

```
def h(x, y):
   mon_max = x
   if x<y:
      mon_max = y
    else:
      mon_max = mon_max
    return mon_max
```

```
def p(x,y):
    if x >= 0:
       return True
    else:
       if y >= 0:
          return True
       else:
          return False
```

```
def t(x):
    if x>=0:
       res = x
    if x<0:
       res = -x
    return res
```

```
def f(x,y):
    if x >= 0:
       if y >= 0:
          return True
       else:
          return False
    else:
       return False
```

3 Exercice de programmation

3.1 Bonjour Mme Python

3. Écrire une fonction `en_tete1` dont le but est de construire un en-tête de lettre (de type chaîne), qui prend en paramètre :

- une chaîne représentant un nommage (un titre, une profession…)
- un booléen valant True si l'entête commence par "Madame" et False si l'entête commence par "Monsieur".

La fonction renvoie une chaîne comme indiqué par ces exemples :

```
>>> en_tete1('la directrice', True)
'Madame la directrice'
>>> en_tete1('le clown', False)
'Monsieur le clown'
```

4. Écrire une fonction `en_tete2` qui a un troisième paramètre de type booléen valant `True` ssi le résultat doit commencer par "Bonjour".

La fonction renvoie une chaîne comme indiqué par ces exemples :

```
>>> en_tete2('la directrice', True, False)
'Madame la directrice'
>>> en_tete2('le clown', False, True)
'Bonjour Monsieur le clown'
```

Pour les besoins de l'exercice vous ne devez pas utiliser un appel à `en_tete1` et vous devez utiliser une seule variable locale.

3.2 Calcul de score à un jeu de dé

On suppose qu'on a lancé 2 dés à 6 faces et qu'on souhaite calculer un score comme suit : le score vaut 20 si les 2 dés ont donné la même valeur, sauf dans le cas du double 6 qui apporte un score de 30. Le score vaut 15 si la somme des dés vaut 7. Dans les autres cas, le score vaut la somme des dés.

5. Écrire un prédicat paramétré par deux entiers quelconques qui renvoie `True` ssi les 2 entiers ont la même valeur.

6. Écrire la fonction `score` qui calcule le score d'un lancer de 2 dés à 6 faces.