TP2

INSTRUCTION CONDITIONNELLE IF

Corrigé

Exercice 1

1. Écrire une suite d'instructions en langage Python qui demande à l'utilisateur de saisir deux réels a et b, puis affiche lequel est le plus petit des deux ou s'il y a égalité.

```
from math import *
a = float(input("donnez-moi un réel svp"))
b = float(input("donnez-moi encore un réel svp"))
if a>b:
    print("le plus grand réel est a =",a)
elif a<b:
    print("le plus grand réel est b =",b)
else:
    print("il y a égalité")
```

2. Tester avec plusieurs valeurs *pertinentes* de a et b que l'on détaille ici :

```
a = 8
b = 8
il y a égalité

a = 9
b = 1
le plus grand réel est a = 9

a = -7.1
b = -2
le plus grand réel est b = -2
```

Exercice 2 Second degré

Écrire une fonction prenant en entrée trois flottants a, b, c (avec $a \neq 0$) et renvoyant les éventuelles solutions réelles de l'équation $ax^2 + bx + c = 0$.

```
from math import *
def second_degre(a,b,c):
    '''second_degre(a:float, b:float, c:float) -> racines reelles de
    ax**2+bx+c = 0'''
    Delta = b**2-4*a*c  # discriminant
    if Delta > 0 :
        r1 = (-b-sqrt(Delta))/(2*a)   # attention aux parentheses
        r2 = (-b+sqrt(Delta))/(2*a)
        return r1,r2
    elif Delta == 0 : # attention le test d'egalite  ==
        r = -b/(2*a)
        return r
    else:
        return None
```

Proposer des équations permettant de tester tous les cas de figure.

```
>>> second_degre(1,1,1) # x^2+x+1 n'a pas de racines réelles

>>> second_degre(1,2,1) # x^2+2x+1 = (x+1)^2
-1.0

>>> second_degre(1,3,2) # x^2+3x+2 = (x+2)(x+1)
(-2.0, -1.0)
```

Exercice 3

```
n = int(input("donner un entier svp"))

if n%2 == 0:
    if n >= 14 : # indentation
        k = n/2-7
    else :
        k = n/2+7
else:
    k = (n+3)/2 # attention aux parenthèses

print(k)

# autre solution
if n%2 == 0 and n >= 14:
    k = n/2-7
elif n%2 and n < 14 :
    k = n/2+7
else:
    k = (n+3)/2
print(k)
```

Proposer des valeurs de n permettant de tester tous les cas de figure.

```
n = 20 # pair >= 14
k = 3

n = 14 # pair >=14, cas limite
k = 0

n = 8 # pair <14
k = 11

n = 13 # impair
k = 8
```

Exercice 4

1.
2.

```
>>> randint(1,6)  # résultat aléatoire d'un dé à 6 faces
5
>>> randint(1,16) # résultat aléatoire d'un dé à 16 faces
12
```

```
    S = float(input("quelle est votre mise ?"))

    de = randint(1,16)

    if de%2 == 0:
        if 1 <=  de <=  5:
            G = 3*S - S
        elif 6 <=  de <=  11:
            G = 2*S - S
        else:
            G = S - S
    else:
        G = -S
    print(S)
```

3.

Exercice 5 Écrire une fonction qui, étant donnée une année, renvoie True si celle-ci est bissextile et renvoie False sinon.
Une année est bissextile (366 jours) si l'un des deux cas suivants se produit :
— l'année est divisible par 4 et non divisible par 100 ;
— l'année est divisible par 400.
Sinon, elle est non-bissextile (365 jours).

```
def bissextile(annee):
    '''bissextile(annee:int) -> bool, True si annee est bissextile,
    False sinon'''
    if (annee%4 == 0 and annee%100 != 0) or annee%400 == 0 : # parentheses
        return True # attention ecriture de True
    else:
        return False
```

Proposer un jeu de test pour cette fonction, permettant de vérifier tous les cas de figure.

```
>>> bissextile(2024) # divisible par 4 mais pas par 100
True

>>> bissextile(2000) # divisible par 400
True

>>> bissextile(2100) # divisible par 100 mais pas par 400
False

>>> bissextile(2021) # pas divisible par 4
False
```