6.4 — Activités

ACTIVITÉ 1

Au jeu de mölkky, chaque joueur marque à son tour de jeu entre 0 et 12 points, qui viennent s'ajouter à son score précédent. Le premier à atteindre un score de 51 gagne. Mais gare! Quiconque dépasse le score cible de 51 revient immédiatement à 25 points.

Écrire un algorithme demandant un score `"Entrer le score : "` et un nombre de points marqués `"Entrer le gain : "`, et qui affiche le nouveau score `"Nouveau score : "` ou signale une éventuelle victoire `"Victoire"`.

CORRECTION

```
[ ]: score = int(input("Entrer le score : "))
     gain  = int(input("Entrer le gain : "))
     nouveau_score = score + gain
     if nouveau_score == 51:
         print("Victoire")
     elif nouveau_score < 51:
         print("Nouveau score :", nouveau_score)
     else:
         print("Nouveau score : 25")
```

ACTIVITÉ 2

Au bowling, on a deux chances pour faire tomber un total de dix quilles. Écrire un algorithme qui demande le nombre de quilles renversées avec chacune des deux boules et affiche `"X"` si toutes les quilles sont tombées à la première boule, `"/"` si toutes les quilles sont tombées, et sinon le nombre de quille renversées.

Exemples :

```
>>> Score première boule : 2
>>> Score deuxième boule : 3
5

>>> Score première boule : 7
>>> Score deuxième boule : 3
/

>>> Score première boule : 10
>>> Score deuxième boule : 0
X
```

CORRECTION

```
[ ]: score_1 = int(input("Score première boule : "))
score_2 = int(input("Score deuxième boule : "))
if score_1 == 10:
    print("X")
elif score_1 + score_2 == 10:
    print("/")
else:
    print(score_1 + score_2)
```

ACTIVITÉ 3

Implémenter un programme qui demande deux nombres à l'utilisateur et affiche `"divisible"` si le premier est divisble par le second et qui affiche `"pas divisible"` sinon.

CORRECTION

```
[ ]: a = int(input("saisir un premier nombre: "))
     b = int(input("saisir un premier nombre: "))

     if a % b == 0:
         print("divisible")
     else:
         print("pas divisible")
```

ACTIVITÉ 4

Écrire un programme qui demande un entier `"Entrer un nombre entier : "` à l'utilisateur et affiche tous ses diviseurs.

Exemples :

```
>>> Entrer un nombre entier : 26
1
2
13
26

>>> Entrer un nombre entier : 10
1
2
5
10
```

CORRECTION

```
[ ]: n = int(input("Entrer un nombre entier positif : "))
     for i in range(1, n+1):
         if n % i == 0:
             print(i)
```

ACTIVITÉ 5

Écrire un programme qui demande un entier `n` à l'utilisateur et affiche tous ses diviseurs en précisant à la fin de l'affichage si le nombre est premier ou pas.

Exemples :

```
>>> Entrer un nombre entier positif : 26
1
2
13
26
26 n'est pas premier

>>> Entrer un nombre entier positif : 41
1
41
41 est premier
```

CORRECTION

```
[ ]: n = int(input("Entrer un nombre entier positif : "))
if n < 0:
    print(n, "n'est pas positif")
else:
    nb_diviseurs = 2
    print(1)
    for i in range(2, n//2 + 1):
        if n % i == 0:
            nb_diviseurs = nb_diviseurs + 1
            print(i)
    print(n)
    if nb_diviseurs == 2:
        print(n, "est premier")
    else:
        print(n, "n'est pas premier")
```

ACTIVITÉ 6

Au bowling, on a deux chances pour faire tomber un total de dix quilles. Écrire un programme qui demande le nombre de quilles renversées avec chacune des deux boules et affiche `X` si toutes les quilles sont tombées à la première boule, `/` si toutes les quilles sont tombées, et sinon le nombre de quille renversées.

Améliorer votre programme en ne demandant les informations de la deuxième boule que si elle a besoin d'être lancée.

Exemples :

```
>>> Score première boule : 2
>>> Score deuxième boule : 3
5

>>> Score première boule : 7
>>> Score deuxième boule : 3
/

>>> Score première boule : 10
X
```

CORRECTION

[ ]:
```python
score_1 = int(input("Score première boule : "))
if score_1 == 10:
    print("X")
else:
    score_2 = int(input("Score deuxième boule : "))
    if score_1 + score_2 == 10:
        print("/")
    else:
        print(score_1 + score_2)
```

ACTIVITÉ 7

Au bowling, on a deux chances pour faire tomber un total de dix quilles. Écrire un programme qui demande le nombre de quilles renversées avec chacune des deux boules et affiche `X` si toutes les quilles sont tombées à la première boule, `/` si toutes les quilles sont tombées, et sinon le nombre de quille renversées.

Améliorer votre programme en ne demandant les informations de la deuxième boule que si elle a besoin d'être lancée.

Améliorer votre programme en affichant `!` si les scores saisis sont impossibles.

Exemples :

```
>>> Score première boule : 2
>>> Score deuxième boule : 3
5

>>> Score première boule : 7
>>> Score deuxième boule : 3
/

>>> Score première boule : 10
X

>>> Score première boule : -5
!

>>> Score première boule : 13
!
```

```
>>> Score première boule : 2
>>> Score deuxième boule : -3
!

>>> Score première boule : 2
>>> Score deuxième boule : 9
!
```

CORRECTION

```
[ ]: score_1 = int(input("Score première boule : "))
if score_1 < 0 or score_1 > 10:
    print("!")
elif score_1 == 10:
    print("X")
else:
    score_2 = int(input("Score deuxième boule : "))
    if score_2 < 0 or score_1 + score_2 > 10:
        print("!")
    elif score_1 + score_2 == 10:
        print("/")
    else:
        print(score_1 + score_2)
```

ACTIVITÉ 8

Écrire un programme qui demande trois longueurs à l'utilisateur, indique si ces trois longueurs peuvent être les longueurs des trois côtés d'un triangle et, le cas échéant, s'il s'agit d'un triangle équilatéral, isocèle, rectangle ou scalène (trois côtés de longueurs différentes)

Exemples :

```
>>> Entrer la première distance : 2
>>> Entrer la deuxième distance : 3
>>> Entrer la troisième distance : 10
Ceci n'est pas un triangle
```

```
>>> Entrer la première distance : 3
>>> Entrer la deuxième distance : 5
>>> Entrer la troisième distance : 6
Ceci est un triangle
scalène

>>> Entrer la première distance : 3
>>> Entrer la deuxième distance : 3
>>> Entrer la troisième distance : 5
Ceci est un triangle
isocèle

>>> Entrer la première distance : 5
>>> Entrer la deuxième distance : 5
>>> Entrer la troisième distance : 5
Ceci est un triangle
équilatéral

>>> Entrer la première distance : 3
>>> Entrer la deuxième distance : 4
>>> Entrer la troisième distance : 5
Ceci est un triangle
scalène
rectangle
```

CORRECTION

```
[ ]: a = int(input("Entrer la première distance : "))
b = int(input("Entrer la deuxième distance : "))
c = int(input("Entrer la troisième distance : "))
if a+b >= c and b+c >= a and c+a >= b:
    print("Ceci est un triangle")
    if a == b and b == c:
        print("équilatéral")
    elif a == b or b == c or c == a:
        print("isocèle")
    else:
        print("scalène")

    if a*a+b*b == c*c or b*b+c*c == a*a or c*c+a*a == b*b:
        print("rectangle")

else:
    print("Ceci n'est pas un triangle")
```

ACTIVITÉ 9

Pour limiter les erreurs dans la transmission d'un message, une technique simple appelée code de répétition d'ordre k consiste à répéter k fois chaque lettre.

Lors de la réception du message, il suffit alors de regarder quelle lettre est majoritaire dans chaque paquet de lettres.

Dans cet exercice, on ne considérera comme lettre que les nombres `0` et `1`.

Écrire un programme qui demande à l'utilisateur de saisir deux lettres et qui affiche la lettre majoritaire, ou `X` si aucune n'est majoritaire.

Exemples :

```
>>> Première lettre : 0
>>> Deuxième lettre : 0
0


>>> Première lettre : 1
```

```
>>> Deuxième lettre : 0
X

>>> Première lettre : 0
>>> Deuxième lettre : 1
X

>>> Première lettre : 1
>>> Deuxième lettre : 1
1
```

CORRECTION

[ ]:
```
l1 = input("Première lettre : ")
l2 = input("Deuxième lettre : ")
if l1 == l2:
    print(l1)
else:
    print("X")
```

ACTIVITÉ 10

Pour limiter les erreurs dans la transmission d'un message, une technique simple appelée code de répétition d'ordre k consiste à répéter k fois chaque lettre.

Lors de la réception du message, il suffit alors de regarder quelle lettre est majoritaire dans chaque paquet de lettres.

Dans cet exercice, on ne considérera comme lettre que les nombres `0` et `1`.

Écrire un programme qui demande à l'utilisateur de saisir trois lettres et qui affiche la lettre majoritaire.

Exemples :

```
>>> Première lettre : 0
>>> Deuxième lettre : 1
>>> Troisième lettre : 1
1

>>> Première lettre : 1
>>> Deuxième lettre : 0
>>> Troisième lettre : 0
0
```

CORRECTION

[ ]:
```
l1 = input("Première lettre : ")
l2 = input("Deuxième lettre : ")
l3 = input("Troisième lettre : ")

if l1 == l2 or l1 == l3:
    print(l1)
else:
    print(l3)
```

ACTIVITÉ 11

Pour limiter les erreurs dans la transmission d'un message, une technique simple appelée code de répétition d'ordre k consiste à répéter k fois chaque lettre.

Lors de la réception du message, il suffit alors de regarder quelle lettre est majoritaire dans chaque paquet de lettres.

Dans cet exercice, on ne considérera comme lettre que les nombres `0` et `1`.

Écrire un programme qui demande à l'utilisateur la valeur de l'ordre k, puis demande de saisir k lettres. Le programme affiche la lettre majoritaire, ou X si aucune n'est majoritaire.

Exemples :

```
>>> Entrer l'ordre du code : 5
>>> Saisir le lettre : 0
>>> Saisir le lettre : 1
>>> Saisir le lettre : 1
>>> Saisir le lettre : 1
>>> Saisir le lettre : 0
1
```

CORRECTION

[ ]:
```python
n = 0
k = int(input("Entrer l'ordre du code : "))
for _ in range(k):
    l = input("Saisir le lettre : ")
    if l == "1":
        n = n + 1
if n > k - n:
    print("1")
elif n < k - n:
    print("0")
else:
    print("X")
```