TD 2 (programmation) - Conditions et boucles.

Conditions

Exercice 1 (Conditions multiples)

Voici un petit programme (on suppose que l'utilisateur rentrera bien un nombre entier):

```
1  if __name__ == "__main__":
2      a = int(input("Entrez un nombre: "))
3      if a <= 10:
4          print('A')
5      elif a > 10 and a <= 50:
6          print('B')
7      elif a > 50 and a < 100:
8          print('C')
9      elif a >= 100:
10         print('D')
```

Réduisez le nombre de comparaisons dans ce programme tout en gardant le même comportement.

Indices

Certaines conditions sont inutiles, car elles ont déjà été testées.

Exercice 2 (Trinôme)

Voici un programme:

```
1  if __name__ == "__main__":
2      a = float(input('a = '))
3      b = float(input('b = '))
4      c = float(input('c = '))
5      delta = b * b - 4 * a * c
6      if delta > 0:
7          print('deux solutions')
8      elif delta == 0:
9          print('une seule solution')
10     else:
11         print('pas de solution')
```

1. Modifiez le programme pour qu'il affiche les solutions, quand il y en a. Pour calculer $\sqrt{x}$, vous pouvez élever la variable `x` à la puissance 1/2.

   Indices

   Appliquez les formules vues au cours de mathématiques.

2. Modifiez le programme pour qu'il gère correctement les cas où `a` vaut 0, ce qui n'est pas le cas actuellement.

Indices

En fonction de l'endroit où vous les avez insérées, les formules utilisées provoquent une erreur pour division par 0 quand `a` est nul. Attention, il reste encore d'autres cas à traiter (`b`, `c` sont-elles nulles?).

Boucle for

Exercice 3 (Boucle de comptage)

Que fait le programme ci-dessous?

```
if __name__ == "__main__":
    for i in range(10):
        print(i)
    print('fini')
```

1. Modifiez-le pour qu'il affiche les nombres de 1 à 10.
2. Modifiez-le pour qu'il affiche les nombres de 3 à 12.
3. Modifiez-le pour qu'il affiche les nombres pairs entre 3 et 12.

Exercice 4 (Factorielle)

On rappelle que $n!$ (la *factorielle* de n) vaut $1 \times 2 \times 3 \times \cdots \times n$ pour tout entier $n \geq 1$ et que $0! = 1$. Écrivez un programme qui demande n et affiche la valeur de $n!$.

Indices

Construisez le résultat pas à pas, en calculant d'abord 1×2, puis $1 \times 2 \times 3$, puis $1 \times 2 \times 3 \times 4$, ...

Exercice 5 (Trois mais pas cinq)

Écrivez un programme qui prend un entier positif n en entrée, et affiche tous les nombres de 1 à n qui sont divisibles par 3 mais pas par 5.

Exercice 6 (Fibonacci)

La suite de Fibonacci F_n est définie par:

$$F_n = \begin{cases} 0 & \text{si } n = 0, \\ 1 & \text{si } n = 1, \\ F_{n-1} + F_{n-2} & \text{sinon.} \end{cases}$$

1. Calculez à la main les premières valeurs de la suite, pour $n \leq 10$.
2. Écrivez un programme qui demande un entier n à l'utilisateur et affiche F_n.

Indices

Pour un n arbitraire, il est facile de calculer F_n si l'on connaît les deux valeurs précédentes ... mais ces deux valeurs nécessitent d'en connaître d'autres, et ainsi de suite. On va donc calculer tous les F_i jusqu'à la valeur demandée en partant de F_0, et en stockant à chaque étape les deux valeurs dont on a besoin pour connaître la suivante dans des variables u et v. Ces valeurs "glisseront" ensuite, comme illustré ci-dessous, pour pouvoir calculer la prochaine valeur demandée:

	F_0	F_1	F_2	F_3	F_4	F_5	F_6	F_7	F_8	F_9	F_{10}
	0	1	1	2	3	5	8	13	21	34	55
étape 0:	u	v									
étape 1:		u	v								
étape 2:			u	v							
⋮											