DM algorithme $n°2$, 2nde

1 Devoir maison : algorithmique, deuxième partie. A rendre par binôme le lundi après les vacances scolaires.

Dans ce devoir maison, nous poursuivons l'étude du langage Python. Pour rappels : vous pouvez utiliser un éditeur de programme en ligne pour vous entrainer : il suffit de s'inscrire (« sign in ») sur le site **https ://trinket.io** puis de cliquer sur l'onglet bleu ***« new trinket »*** pour ouvrir la console de programmation. Une fois les instructions écrites, cliquer sur le bouton ***« run »***.

1.1 L'instruction conditionnelle « if »

Il est parfois utile d'introduire des boucles conditionnelles dans un programme. En python, de telles boucles s'obtiennent à l'aide des instructions « *if, elif et else* ». Voici un résumé de ce qu'il faut savoir à ce sujet.

MÉMO

▶ Pour aiguiller dans différentes directions l'exécution d'un programme, il est possible d'avoir recours à des instructions conditionnelles qui permettent de déterminer si les instructions qui suivent doivent être, ou non, exécutées.
En langage naturel, la syntaxe d'une instruction conditionnelle est du type ci-contre.

Si condition **alors**
    instruction(s) 1
Sinon
    instruction(s) 2

▶ **La syntaxe des instructions conditionnelles en langage Python**

Syntaxe en Python	Exemple en langage naturel	Exemple en langage Python
Une seule condition		

if condition :
    instruction(s)
	Si n est supérieur ou égal à 3 **alors**
	    $m \leftarrow 2 \times n$
		`1 if n>=3:`
		`2     m=2*n`

Une condition et une alternative		

if condition :
    instruction(s) 1
else :
    instruction(s) 2
	Si n est supérieur ou égal à 3 **alors**
	    $m \leftarrow 2 \times n$
	Sinon
	    $m \leftarrow 3 \times n + 1$
		`1 if n>=3:`
		`2     m=2*n`
		`3 else:`
		`4     m=3*n+1`

Deux conditions ou plus		

if condition 1 :
    instruction(s) 1
elif condition 2 :
    instruction(s) 2
else :
    instruction(s) 3
	Si n est inférieur ou égal à 2 **alors**
	    $m \leftarrow 2 \times n$
	Sinon si n est strictement compris entre 2 et 5 **alors**
	    $m \leftarrow -2 \times n + 5$
	Sinon
	    $m \leftarrow n^2$
		`1 if n<=2:`
		`2     m=2*n`
		`3 elif 2<n<5:`
		`4     m=-2*n+5`
		`5 else:`
		`6     m=n**2`

Remarques :
- Le mot clé « alors » n'existe pas en langage Python. C'est l'indentation, c'est-à-dire le décalage automatique du retour à la ligne vers la droite, qui le remplace.
- elif est la contraction de else if.
- Pour tester l'égalité de deux valeurs en langage Python, on peut utiliser le signe « = = ».

Exercice 1. Considérons la fonction f définie par le programme ci-dessous.

```
>>> def f(x) :
        if x>+2 :
                y=3*x-4
        elif -1<x<2 :
                y=-x+4
        else :
                y=2*x+7
        return y
```

1. Compléter l'expression de $f(x)$ donnée ci-dessous pour qu'elle coïncide avec le programme écris en Python.

$$\begin{cases} \ldots\ldots\ldots & \text{si} \quad x \in \ldots\ldots \\ \ldots\ldots\ldots & \text{si} \quad x \in \ldots\ldots \\ \ldots\ldots\ldots & \text{si} \quad x \in \ldots\ldots \end{cases}$$

Exercice 2. Dans une école de Rugby, il y a quatre groupes :

- le groupe $U8$ pour les joueurs entre 8 ans (inclus) et 10 ans (exclus) ;
- le groupe $U10$ pour les joueurs entre 10 ans (inclus) et 12 ans (exclus) ;
- le groupe $U12$ pour les joueurs entre 12 ans (inclus) et 14 ans (exclus) ;
- le groupe $U4$ pour les joueurs entre 14 ans (inclus) et 16 ans (exclus) ;

Compléter (en remplaçant les points d'interrogations) le script pour qu'il affiche le groupe lorsque l'utilisateur entre l'âge du joueur.

```
a=int(input("Donner l'âge du joueur :"))
if a<8 :
print("Trop jeune")
elif 8<=a<10 :
    print("U8")
elif ????? :
    print("???")
elif ??????
    ????????
else :
????????
```

Exercice 3. Dans les années 1960, un jeu télévisé américain proposait au participant de partir avec ce qu'il trouvait derrière une porte. Dans ce jeu, le candidat est face à trois portes. Il sait que derrière l'une de ces portes est cachée une voiture et derrière les deux autres, une chèvre.
Il choisit une porte sans l'ouvrir. Alors, le présentateur, Monty Hall, qui sait où se cache la voiture, ouvre l'une des deus portes derrière laquelle se trouve une chèvre. Il demande ensuite au candidat de conserver son choix initial ou bien de changer de porte.

Le but de cet exercice est de déterminer la meilleure stratégie à adopter du point de vue des probabilités.

Voici un script qui servira à répondre aux questions de la première partie.

```
from random import randint
voiture=randint(1,3)
choix=randint(1,3)
if choix==voiture :
    gagne_changeant =0
    gagne_sans_changer=1
else :
    gagne_changeant=1
    gagne_sans_changer=0
print("s'il change de porte :")
if gagne_changeant ==1 :
    print("le candidat gagne")
else :
    print("le candidat perd")
print("s'il garde sa porte:")
if gagne_changeant == 1 :
    print("le candidat perd")
else :
    print("le candidat gagne")
```

1. **Partie** 1 **:**
   (a) Quelle est la signification de la ligne 1 ?

   (b) Expliquer la ligne 2.

   (c) Expliquer les lignes 10 à 14.
   (d) Copier ce programme dans un éditeur Python et l'exécuter plusieurs fois. Conjecturer la meilleure stratégie entre changer de porte ou garder la porte initiale.

2. **Partie** 2 **:** afin d'automatiser le dénombrement des parties, on décide de modifier le programme précédent pour qu'il renvoie la fréquence de parties gagnées en changeant de porte et la fréquence de partie gagnées sans changer de porte.

```
from random import randint
def jeu(n)
    garde =0
    change =0
    for i in range(?) :
        voiture=randint(0,2)
        choix=randint(0,2)
        if choix == ? :
            ouverte=(voiture+1+randint(0,1))%3
            garde=garde+1
        else :
            ouverte = 3- choix-voiture
            change = ?
    return (garde/n,change/n)
```

(a) Compléter la fonction ci-contre (en remplaçant les points d'interrogations) pour qu'elle réponde au problème posé.

(b) Expliquer la ligne 9.

(c) Expliquer la ligne 12.
Copier la fonction dans l'éditeur Python, déterminer les fréquences pour 1000 parties jouées et conjecturer la meilleure stratégie.

Exercice 4 (Bonus). Proposer une fonction *degre1(a,b)* permettant de résoudre une équation $ax+b=0$ lorsque les paramètres a et b sont donnés par l'utilisateur. Ce programme doit renvoyer la solution (si elle existe).