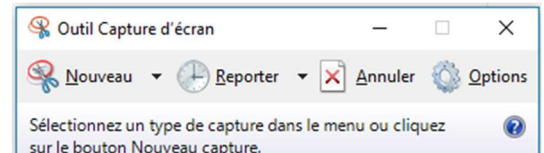


OBJECTIFS : L'objectif principal de ce troisième TP est d'utiliser la structure « *if .. else* » dans des situations différentes.

DOCUMENT A RENDRE : Ce travail est évalué. Vous en rédigerez un compte-rendu sous le nom « *tp3_nomfamille.doc* » ou « *tp3_nomfamille.odt* » et vous le transférez en fin d'activité par l'intermédiaire de l'onglet transfert du site https://nsibrantly.fr/transferts_premiere.php en utilisant comme identifiant : **tp3** et mot de passe : **tp3** . Ce compte-rendu contiendra :

- les réponses aux différentes questions posées,
- les captures d'écran des morceaux de codes écrits et celles des résultats des exécutions données dans le shell. Pour faire ces captures, utiliser l'*Outil Capture d'écran* de Windows.



1. Code pour les mathématiques :

Rappel de maths : Les diviseurs de 6 sont : 1 , 2 , 3 et 6 car le reste de la division euclidienne de 6 par 1 ou de 6 par 2 ou de 6 par 3 ou de 6 par 6 **est nul**.

Rappel pour python : Le reste de la division euclidienne de 6 par 2 par exemple se calcule en utilisant l'opérateur % . On a par exemple : $6 \% 2 = 0$. Par contre $6 \% 4 = 2$ et donc 4 n'est pas un diviseur de 6.

Comment faire pour trouver les diviseurs d'un nombre n quelconque ? : On calcule le reste de la division euclidienne de n par 1, puis par 2, puis par, jusqu'à celui de n par n . Si ce reste est nul, on ajoute ce diviseur au message de sortie.

⇒ Compléter le code ci-dessous, qui demande un entier n à l'utilisateur et affiche en retour tous ses diviseurs.

En exécutant ce code pour $n = 6$ on obtiendra par exemple :

```
>>> (executing file "ex1.py")
Entrez un entier : 6
Les diviseurs de 6 sont : 1 2 3 6
```

```
1 n = int(input("Entrez un entier : "))
2 message = "Les diviseurs de "+str(n)+" sont : "
3 for i in range(1,n+1) :
4     reste = n % i
5     if reste == 0 :
6         message = message + str(i) + " "
7
8 print(message)
```

2. Ecart entre 2 nombres :

⇒ Ecrire un programme qui demande 2 nombres à l'utilisateur et affiche en retour l'écart entre ces nombres (différence sans signe). Attention, le premier nombre donné ne sera pas nécessairement le plus petit des deux.

Exemples d'exécution :

```
>>> (executing file "ex2.py")
Premier nombre : 5
Second nombre : 1
Ecart : 4.0
```

```
>>> (executing file "ex2.py")
Premier nombre : 1
Second nombre : 5
Ecart : 4.0
```

```
1 a = float(input("Premier nombre : "))
2 b = float(input("Second nombre : "))
3 if a > b :
4     print("Ecart : ",a-b)
5 else :
6     print("Ecart : ",b-a)
```

3. Années bissextiles :

Info tirée de Wikipédia :

Depuis l'ajustement du calendrier grégorien, l'année n'est bissextile (elle aura 366 jours)¹ que dans l'un des deux cas suivants :

1. si l'année est divisible par 4 et non divisible par 100 ;
2. si l'année est divisible par 400.

Dans un autre cas, l'année n'est pas bissextile : elle a la durée habituelle de 365 jours.

⇒ Ecrire un programme qui demande une année à l'utilisateur et indique s'il s'agit d'une année bissextile. Exemples d'exécution :

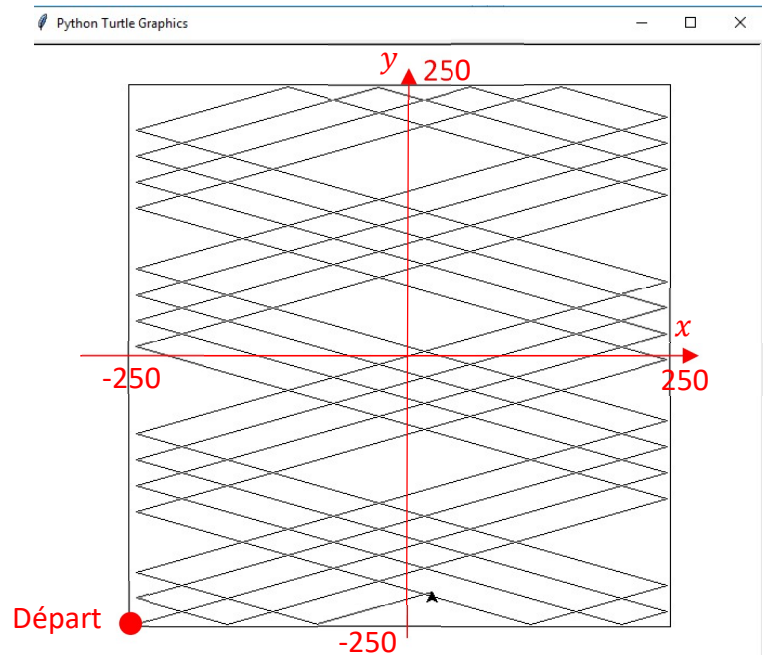
```
>>> (executing file "ex3.py")
Année : 2020
Année Bissextile
```

```
1 n = int(input("Année : "))
2
3 if n%4 == 0 and n%100 != 0 :
4     message = "Année Bissextile"
5 elif n%400 == 0 :
6     message = "Année Bissextile"
7 else :
8     message = "Année non Bissextile"
9
10 print(message)
```

4. Tortue folle dans sa cage :

⇒ Ecrire un code qui :

- trace un carré centré sur l'origine du repère, de 500 px sur 500 px,
- positionne la tortue sur le point de départ de coordonnées (-250, -250)
- comporte une boucle « *for i in range(2000)* : » dans laquelle, la tortue se déplace à chaque itération d'une valeur $dx = 5$ et $dy = 2$. Si elle arrive au bord du carré, le signe de dx ou de dy est modifié ($dx = -dx$ ou $dy = -dy$) afin que la tortue puisse rebondir et repartir dans l'autre sens. Ainsi la tortue reste piégée dans sa cage.



```

1  from turtle import *
2
3  speed(10)
4  #Tracé d'un carré de coté a = 250 centré sur l'origine
5  a = 250
6  up()
7  goto(-a,-a)
8  down()
9  for i in range(4) :
10     forward(2*a)
11     left(90)
12
13 #Position initiale de la tortue
14 up()
15 goto(-250,-250)
16 down()
17
18 #Déplacement tortue
19 dx = 5
20 dy = 2
21 for i in range(1000) :
22     #On récupère les coordonnées actuelles de la tortue
23     x,y = pos()
24     #On calcule les nouvelles coordonnées
25     x_new = x + dx
26     y_new = y + dy
27     #Si ces nouvelles coordonnées font sortir du carré,
28     #on change le signe du déplacement :
29     if x_new >= 250 or x_new <= -250 :
30         dx = -dx
31         x_new = x + dx
32
33     if y_new >= 250 or y_new <= -250 :
34         dy = -dy
35         y_new = y + dy
36
37     #On déplace la tortue
38     goto(x_new , y_new)
39
40
41 exitonclick()

```

5. Modifier les lettres d'une chaîne de caractère :

a- Découverte d'une nouvelle structure utilisant « for » :

⇒ Exécuter le code ci-contre :

```
1 phrase = "Aujourd'hui c'est mardi"
2 for l in phrase :
3     print(l)
```

On voit que la structure « for » permet d'avoir accès à chacun des caractères de la chaîne de caractères contenue ici dans la variable « phrase ».

b- Utilisation de cette structure pour faire du cryptage :

⇒ Compléter le code ci-dessous, qui utilise cette structure, pour réécrire une phrase en remplaçant les « a » par des « e », les « i » par des « o » et les « u » par des « y ».

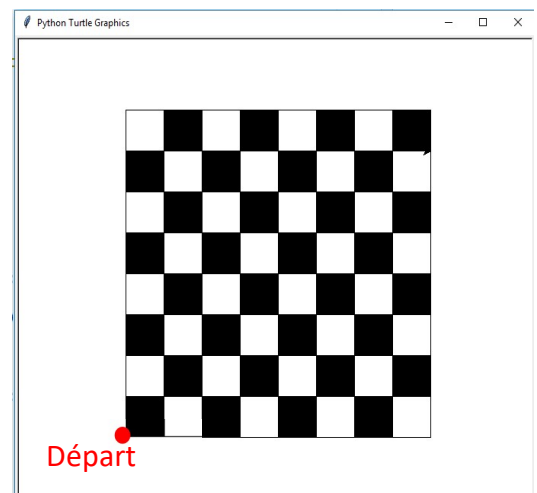
Exemples d'exécution :

```
>>> (executing file "ex5.py")
eyjoyrd'hyo c'est merdo
```

```
1 phrase = "aujourd'hui c'est mardi"
2 phrase_modifiee = ""
3
4 for l in phrase :
5     if l == "a" :
6         phrase_modifiee = phrase_modifiee + "e"
7     elif l == "i" :
8         phrase_modifiee = phrase_modifiee + "o"
9     elif l == "u" :
10        phrase_modifiee = phrase_modifiee + "y"
11    else :
12        phrase_modifiee = phrase_modifiee + l
13
14 print(phrase_modifiee)
```

6. Créer un damier :

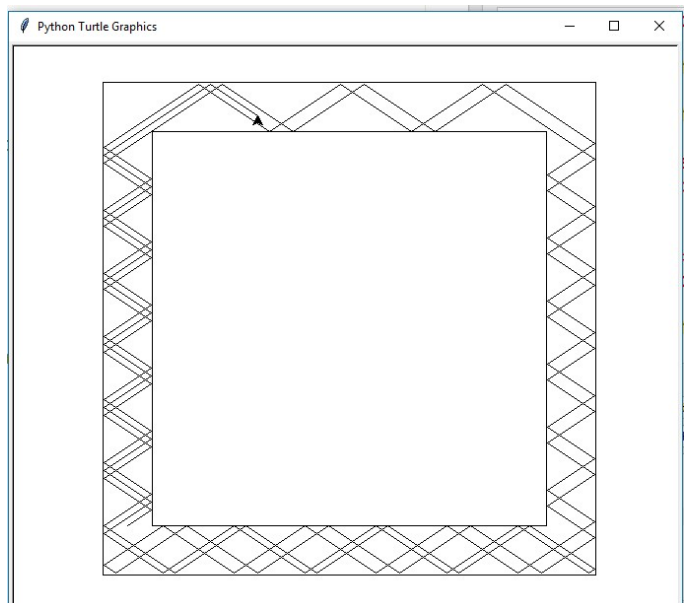
⇒ Créer un code qui crée le damier ci-contre, centré sur l'origine du repère et dont chaque case a une taille de 50px par 50px. Pour y arriver, procéder par étapes :



```
1 from turtle import *
2 from random import *
3 speed(10)
4 # Tracé d'un carré de coté a = 250 centré sur l'origine
5 b = 200
6 up()
7 goto(-b, -b)
8 down()
9 a = 50
10 for k in range(8) :
11     up()
12     goto(-b, -b+k*a)
13     down()
14     for j in range(8) :
15         if (k+j)%2 == 0 :
16             begin_fill()
17             for i in range(4) :
18                 forward(a)
19                 left(90)
20             forward(a)
21             if (k+j)%2 == 0 :
22                 end_fill()
23
24 exitonclick()
```

7. S'il vous reste du temps :

⇒ Reprendre l'exercice 4 de la tortue folle. Modifier le code pour que la tortue se déplace entre le carré de largeur 500px et un autre de largeur 400 px.



```
1 from turtle import *
2 speed(10)
3 # Tracé d'un carré de coté a = 250 centré sur l'origine
4 a = 250
5 up()
6 goto(-a,-a)
7 down()
8 for i in range(4) :
9     forward(2*a)
10    left(90)
11 # Tracé d'un carré de coté a = 200 centré sur l'origine
12 a = 200
13 up()
14 goto(-a,-a)
15 down()
16 for i in range(4) :
17     forward(2*a)
18     left(90)
19 # Position initiale de la tortue
20 up()
21 goto(-225,-200)
22 left(90)
23 down()
24 # Déplacement tortue
25 dx = 3
26 dy = 2
27 for i in range(2000) :
28     x,y = pos()
29     x_new = x + dx
30     y_new = y + dy
31     if x_new >= 250 or x_new <= -250 :
32         dx = -dx
33         x_new = x + dx
34
35     if y_new >= 250 or y_new <= -250 :
36         dy = -dy
37         y_new = y + dy
38
39     if y > -200 and y < 200 :
40         if x_new > -200 and x_new < 200 :
41             dx = -dx
42             x_new = x + dx
43
44     if x > -200 and x < 200 :
45         if y_new > -200 and y_new < 200 :
46             dy = -dy
47             y_new = y + dy
48
49     goto(x_new , y_new)
50
51 exitonclick()
```