Notre Dame de La Merci
Exercices généraux sur Python

Exercice 1
Transformer 42569 secondes en heures, minutes, secondes.

Exercice 2
Un magasin de reprographie propose un tarif dégressif. Les 20 premières photographies sont facturées à 10 centimes et les suivantes à 8 centimes.
1. Calculer à la main le coût de 15 puis de 30 photocopies.
2. Écrire une fonction prix(n) qui renvoie le prix en euros pour n photocopies. La tester.

Exercice 3
Voici deux fonctions nommées truc et bidule.

```
def truc(x):
    print(x)
    return(2*x)
    print(3*x)
    return(4*x)
```

```
def bidule(x):
    print(x)
    print(2*x)
    return(3*x)
    print(4*x)
```

On exécute truc(10).
1. Quelle(s) valeur(s) (est) sont affichée(s)? Quelle valeur est renvoyée?
2. Même question avec bidule(10).

Exercice 4 (Diviseurs d'un nombre triangulaire)
Un nombre est dit triangulaire d'indice n s'il égal à $1+2+3+\cdots+n$.
Par exemple, le nombre triangulaire d'indice 5 vaut 15 car $1+2+3+4+5=15$.

1. Écrire une fonction triangle qui renvoie la valeur du nombre triangulaire d'indice n.
   Par exemple triangle(5) renverra 15.
2. Écrire une fonction nbre_diviseurs qui renvoie le nombre de diviseurs d'un entier $n \in \mathbb{N}*$.
   Par exemple, les diviseurs de 6 sont 1, 2, 3, 6. Il y a donc 4 diviseurs, ainsi nbre_diviseurs(6) renverra 4
3. Écrire un script qui détermine le plus petit nombre triangulaire qui admette au moins 50 diviseurs.

Exercice 5 (Découverte des factorielles)
Soit $n \in \mathbb{N}*$, on appelle «factorielle n» noté n!, l'entier $n!=1\times2\times\cdots\times n$ avec la convention $0!=1$.
Par exemple, $4!=1\times2\times3\times4=24=4\times3!$.

1. Écrire le nombre $15\times14\times13\times12\times11$ comme un quotient de deux factorielles.
2. Exprimer à l'aide de factorielles les deux produits suivants : $2\times4\times6\times\cdots\times100$ et $1\times3\times5\times\cdots\times99$.
3. Combien y-a-t-il d'anagrammes du mot fleur? Et du mot tennis?
4. Écrire un script qui calcule 64!
5. Écrire une fonction factorielle qui prend en argument un entier naturel n et renvoie n!

Exercice 6 (Combien de truc et bidule?)
Pour les trois scripts suivants, dire ce qui est affiché et combien de fois.

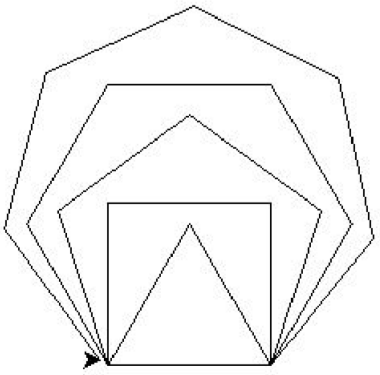
```
for i in range(3):
    print('bidule')
print('truc')
```

```
for i in range(3):
    print('bidule')
for j in range(4):
    print('truc')
```

```
for i in range(3):
    print('bidule')
    for j in range(4):
        print('truc')
```

Exercice 7 (polygone régulier)

1. Tracer un triangle équilatéral, un carré puis un pentagone régulier.
2. On automatise tout cela! Définir une fonction polygone(n, L) qui trace un polygone régulier à n côtés, chacun de longueur L, puis reproduire à l'aide de cette fonction la figure 2.
3. En déduire une méthode pour tracer un cercle.



Exercice 8 (Nombres d'Amstrong)

On souhaite déterminer les entiers naturels qui sont égaux à la somme des cubes de leurs chiffres. De tels entiers seront appelés des nombres d'Armstrong.

Par exemple, l'entier 0 est un nombre d'Armstrong car $0^3 = 0$

mais l'entier 59 n'en est pas un car $5^3 + 9^3 = 854$ donc $5^3 + 9^3 \neq 59$.

1. Écrire une fonction somme_cubes_chiffres qui prend en argument un entier naturel et renvoie la somme des cubes de ses chiffres.

   Par exemple, somme_cubes_chiffres(256) devra renvoyer $2^3 + 5^3 + 6^3 = 349$.
2. Écrire un script qui détermine les nombres d'Amstrong inférieurs à 10 000.

Notre Dame de La Merci – CORRIGE

Exercice 1

Transformer 42569 secondes en heures, minutes, secondes.

```
n=int(input("Entrez le nombre de secondes"))
H=n//3600
r=n%3600
M=r//60
S=r%60
print(n,'secondes =',H,'heures,',M,'minutes et',S,'secondes')
```

→si n = 42569, alors 42569 secondes = 11 heures, 49 minutes et 29 secondes

Exercice 2

Un magasin de reprographie propose un tarif dégressif. Les 20 premières photographies sont facturées à 10 centimes et les suivantes à 8 centimes.

1. Calculer à la main le coût de 15 puis de 30 photocopies.
   15 photocopies : $15 \times 10 = 150$ centimes
   30 photocopies : $20 \times 10 + 10 \times 8 = 280$ centimes
2. Écrire une fonction prix(n) qui renvoie le prix en euros pour n photocopies. La tester.

```
def prix(n):
    if n<=20:
        print(20*n, "centimes")
    else:
        print(200+(n-20)*8, "centimes")

n=int(input("Saisir le nombre de photocopies :"))
prix(n)
```

→si n = 50 : 440 centimes

Exercice 3

Voici deux fonctions nommées truc et bidule.

```
def truc(x):
    print(x)
    return(2*x)
    print(3*x)
    return(4*x)
```

```
def bidule(x):
    print(x)
    print(2*x)
    return(3*x)
    print(4*x)
```

1. On exécute truc(10). Quelle(s) valeur(s) (est) sont affichée(s)? Quelle valeur est renvoyée?
   Le programme affiche 10 puis renverra la valeur 20
   Une fonction s'arrête après la commande **return**.
2. Même question avec bidule(10).
   Le programme affiche 10 et 20 puis renverra la valeur 30
   Une fonction s'arrête après la commande **return**.

Exercice 4 (Diviseurs d'un nombre triangulaire)

Un nombre est dit triangulaire d'indice n s'il égal à $1+2+3+\cdots+n$.
Par exemple, le nombre triangulaire d'indice 5 vaut 15 car $1+2+3+4+5=15$.

1. *Écrire une fonction triangle qui renvoie la valeur du nombre triangulaire d'indice n.*
   Par exemple triangle(5) renverra 15.
2. *Écrire une fonction nbre_diviseurs qui renvoie le nombre de diviseurs d'un entier $n \in N*$.*

Par exemple, les diviseurs de 6 sont 1, 2, 3, 6. Il y a donc 4 diviseurs, ainsi nbre_diviseurs(6) renverra 4

3. *Écrire un script qui détermine le plus petit nombre triangulaire qui admette au moins 50 diviseurs.*

```
def triangle(n):
  S=0
  for i in range(1,n+1):
    S=S+i
  return(S)

def nb_diviseurs(n):
  D=0
  for i in range(1,n+1):
    if n%i==0:
      D=D+1
  return(D)

i=2
nbdiv=0
while nbdiv<50:
  i=i+1
  k=triangle(i)
  nbdiv=nb_diviseurs(k)
print(triangle(i))
```

→25200

Exercice 5 (Découverte des factorielles)

Soit $n \in \mathbb{N}*$, on appelle «factorielle n» noté n!, l'entier $n!=1\times2\times\cdots\times n$ avec la convention $0!=1$.

Par exemple, $4!=1\times2\times3\times4=24=4\times3!$.

1. Écrire le nombre $15\times14\times13\times12\times11$ comme un quotient de deux factorielles.
2. Exprimer à l'aide de factorielles les deux produits suivants : $2\times4\times6\times\cdots\times100$ et $1\times3\times5\times\cdots\times99$.
3. Combien y-a-t-il d'anagrammes du mot fleur? Et du mot tennis?
4. Écrire un script qui calcule 64!
5. Écrire une fonction factorielle qui prend en argument un entier naturel n et renvoie n!

Exercice 6 (Combien de truc et bidule?)

Pour les trois scripts suivants, dire ce qui est affiché et combien de fois.

```
for i in range(3):
    print('bidule')
print('truc')
```

→bidule, bidule, bidule, truc

```
for i in range(3):
    print('bidule')
for j in range(4):
    print('truc')
```

→ bidule, bidule, bidule, truc, truc, truc, truc

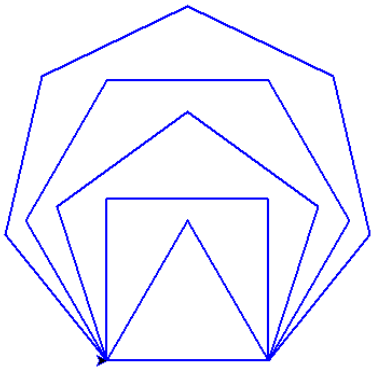
```
for i in range(3):
    print('bidule')
    for j in range(4):
        print('truc')
```

→ bidule, truc, truc, truc, truc, bidule, truc, truc, truc, truc, bidule, truc, truc, truc, truc

Exercice 7 (polygone régulier)

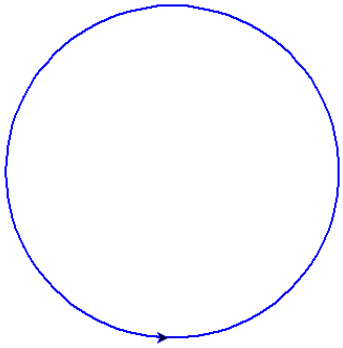
1. *Tracer un triangle équilatéral, un carré puis un pentagone régulier.*
2. *On automatise tout cela! Définir une fonction polygone(n, L) qui trace un polygone régulier à n côtés, chacun de longueur L, puis reproduire à l'aide de cette fonction la figure 2.*

```
def polygone(n,l):
    for i in range(n):
        forward(l)
        left(2*180/n)

from turtle import *
pensize(2)
pencolor("blue")
for i in range (3,8):
    polygone(i,150)
mainloop()
```



3. *En déduire une méthode pour tracer un cercle.*
Il faudrait choisir un grand nombre de côtés de petite dimension (pour qu'il tienne dans la fenêtre d'écriture :

```
def polygone(n,l):
    for i in range(n):
        forward(l)
        left(2*180/n)

from turtle import *
pensize(2)
pencolor("blue")
polygone(100,10)
mainloop()
```



Exercice 8 (Nombres d'Amstrong)

On souhaite déterminer les entiers naturels qui sont égaux à la somme des cubes de leurs chiffres. De tels entiers seront appelés des nombres d'Armstrong.

Par exemple, l'entier 0 est un nombre d'Armstrong car $0^3 = 0$

mais l'entier 59 n'en est pas un car $5^3 + 9^3 = 854$ *donc* $5^3 + 9^3 \neq 59$.

1. *Écrire une fonction somme_cubes_chiffres qui prend en argument un entier naturel et renvoie la somme des cubes de ses chiffres.*

Par exemple, somme_cubes_chiffres(256) devra renvoyer $2^3 + 5^3 + 6^3 = 349$.

```
def somme_cubes_chiffres(n):
    S=0
    while n != 0:
        S=S+(n%10)**3
        n=n//10
        print(S)     # n'est pas utile mais permet de visualiser le bon déroulement du processus
    print(S)

somme_cubes_chiffres(421)

→
1
9
73
73
```

2. *Écrire un script qui détermine les nombres d'Amstrong inférieurs à 10000.*

```
def somme_cubes_chiffres(n):
    S=0
    i=1
```

```
    while n != 0:
        S=S+(n%10**i)**3
        n=n//10**i
    return(S)

for i in range(10001):
    if somme_cubes_chiffres(i)==i:
        print(i)
```

→

```
0
1
153
370
371
407
```