```
# I:\stage_python\scripts_corriges\scripts_cours.py
001| ## Exemples vus pendant la formation Python l'après-midi
002| 
003| 
004| ## Nombres entiers
005| 
006| # addition, multiplication et exponentiation
007| 4**3 # 4 puissance 3
008| 
009| # modulo et division entière
010| """ convertir 4562 minutes: 3 jours 4 heures et 2 minutes """
011| 4562 % 60 # nombre de minutes
012| 
013| # cas du modulo 10 et du modulo 2 (décimales et test de parité)
014| 
015| 
016| # entiers de taille arbitraire mais pas les nombres flottants:
017| 2**5646 # OK
018| 2.0**5646 # overflow
019| 
020| 
021| ## Nombres flottants
022| 
023| # attention aux nbres flottants, calcul approché
024| 2*0.1 == 0.2
025| 3*0.1 == 0.3
026| 
027| #pas de test d'égalités entre les nbres flottants
028| 
029| 1 + 2**-52 == 1
030| 1 + 2**-53 == 1
031| 
032| # tentative d'explication avec l'écriture scientifique
033| """
034| 15 ou 16 chiffres significatifs en base 10
035| >>> 1 + 10**16 == 10**16
036| False
037| 
038| >>> 1.0 + 10**16 == 10**16
039| True """
040| 
041| 
042| ### chaînes de caractères
043| 
044| # concaténation
045| 'toto'+ 'retour'
046| 'retour' + 'sinju'
047| 2*'toto'
048| 
049| # comparaison des chaînes + notion de code ascii ou unicode
050| 'volley' > 'football'
051| 'Volley' > 'football'
052| 
053| # remarque:
054| ord('a')
055| ord('b')
056| ord('A')
057| 
058| help(chr)   # 0x10ffff est un entier écrit en hexadécimal, de moins de 3 octets
...
059| 
060| # un peu de grec
061| print(chr(945))
062| print( chr(0x3B1)) # chr(945)
063| 
064| 
065| ## affectation
```

```
066|·
067|·x·=·5
068|·y·=·2*x
069|·x·=·x·+·3
070|·#·que·valent·x·et·y?·L'affectation·n'est·pas·une·égalité
071|·
072|·'''EXO·:·échange·de·deux·valeurs'''
073|·x·=·3
074|·y·=·10
075|·temp·=·x
076|·x·=·y
077|·y·=·temp
078|·
079|·###·notion·de·modules
080|·
081|·#·la·fonction·racine·carrée
082|·
083|·import·math
084|·
085|·sqrt(25)·#·error
086|·math.sqrt(25)·#·on·préfixe
087|·#·ou·faire·25**0.5
088|·
089|·#·autre·méthode·d'import
090|·from·math·import·floor,·ceil
091|·
092|·floor(12.3)·#·pas·besoin·de·préfixer
093|·ceil(12.4)
094|·
095|·
096|·###·un·peu·de·graphismes·avec·la·tortue
097|·from·turtle·import·*·#·on·importe·toutes·les·fonctions·du·module·math
098|·
099|·#·dessin·d'un·rectangle·de·60·pixels·par·30·pixels
100|·forward(60)
101|·left(90)
102|·forward(30)
103|·left(90)
104|·forward(60)
105|·left(90)
106|·forward(30)
107|·left(90)·#·on·remet·la·tortue·en·direction·initiale
108|·
109|·
110|·#·j'automatise,·ma·première·fonction
111|·
112|·def·rectangle(longueur,·largeur):
113|·····forward(longueur)
114|·····left(90)
115|·····forward(largeur)
116|·····left(90)
117|·····forward(longueur)
118|·····left(90)
119|·····forward(largeur)
120|·····left(90)·#·on·remet·la·tortue·en·direction·initiale
121|·
122|·reset()
123|·rectangle(40,20)
124|·rectangle(50,30)
125|·rectangle(60,40)
126|·
127|·
128|·#·et·pour·dessiner·un·carré?
129|·reset()
130|·rectangle(40,40)
131|·
132|·#·puis·on·réautomatise
```

```
133|·
134|·"""·EXO:·coder·une·fonction·carré"""
135|·def·carre(longueur_cote):
136|·····rectangle(longueur_cote,·longueur_cote)
137|·
138|·#·voici·3·carrés
139|·reset()
140|·carre(20)
141|·carre(40)
142|·carre(80)
143|·
144|·##·Une·fonction·doit·renvoyer!!·Return·est·mon·ami
145|·
146|·def·f(x):
147|·····print(2*x)
148|·
149|·"""·EXO:·que·résultat·après·exécution?"""
150|·
151|·f(5),·f('to'),·f(f(5))·
152|·
153|·#·différence·entre·print·et·return:·ma_fonction(10)?
154|·
155|·def·ma_fonction(x):
156|·····print(x)
157|·····return(2*x)
158|·····print(3*x)
159|·····return(4*x)
160|·
161|·#·une·fonction·qui·qui·prend·un·mot·et·renvoie·ce·mot·écrit·à·l'envers
162|·def·verlan(mot):
163|·····nouveau_mot·=·''
164|·····for·lettre·in·mot:
165|·········nouveau_mot·=·lettre·+·nouveau_mot
166|·····return·nouveau_mot
167|·
168|·def·norme(x,y):
169|·····return·(x**2·+·y**2)**0.5
170|·
171|·###·tests
172|·"""·EXO:·qu'affiche·le·script·suivant?"""
173|·
174|·taille·=·80
175|·if·taille·<·110:
176|·····print('trop·petit')
177|·elif·taille·>·200:
178|·····print('trop·grand')
179|·else:
180|·····print("t'as·les·pieds·qui·touchent·le·sol")
181|·
182|·
183|·taille·=·80
184|·if·taille·<·110:
185|·····print('trop·petit')
186|·if·taille·>·200:
187|·····print('trop·grand')
188|·else:
189|·····print("t'as·les·pieds·qui·touchent·le·sol")
190|·
191|·
192|·###··boucles·conditionnelles
193|·
194|·#·1·compte·à·rebours
195|·
196|·temps·=·5
197|·while·temps··>·0:
198|·····print(temps)
199|·····temps·=·temps·-·1
```

```
# arrivé ici temps vaut 0
print("décollage")


'''EXO temps d'attente d'un double six'''

from random import randint
de1 = randint(1,6)
de2 = randint(1,6)
nbre_lancers = 1

while de1 + de2 != 12:
    de1 = randint(1,6)
    de2 = randint(1,6)
    nbre_lancers = nbre_lancers + 1
print(nbre_lancers)


# 2 temps d'attente d'un double six
from random import randint

nbre_lancers = 0
pas_double_six = True
while pas_double_six:
    de1 = randint(1,6)
    de2 = randint(1,6)
    if de1 == 6 and de2 == 6:
        pas_double_six = False
    nbre_lancers = nbre_lancers + 1
print(nbre_lancers)

## itérations

# notion de liste

maliste = [4,6,12,21]

maliste[0] = 21 # attention, les indices démarrent à zéro
maliste[0] = 're'
maliste[0] = 5
maliste.append(25) # méthode pour ajouter l'élément 25 en bout de la liste appelée maliste


# notion d'itérateur

for nbre in liste:
    print(2*nbre)


# for k in range(2,20):  list(range(2,20))

    
# affichage des vingt premières lettres de l'alphabet grec
for k in range(20):
    print(chr(945+k))

# affichage des carrés

for k in range(2,10):
    print(k**2)


""" EXO""" # la somme des entiers de 50 à 100

s = 0
for k in range(50, 101):
```

```
266|·····s·=·s·+·k
267|·
268|·
269|·#·carré·avec·la·tortue
270|·
271|·def·carre(longueur_cote):
272|·····n·=·longueur_cote
273|·····for·_·in·range(4):
274|·········forward(n)
275|·····left(90)
276|·
277|·
278|·#·chaînes·ou·les·listes·sont·itérables
279|·
280|·liste_couleurs·=·['bleu',·'rouge',·'vert']
281|·for·couleur·in·liste_couleurs:
282|····print(couleur)
283|·
284|·
285|·for·lettre·in·'bonjour':
286|·····print(lettre.upper())
287|·
288|·##·FIN...............
289|·
290|·
291|·
292|·
293|·
294|·
295|·
296|·
297|·
298|·
299|·
300|·##
301|·
302|·
303|·
304|·
305|·#·3·tables·de·multiplications
306|·for·i·in·range(10):
307|·····print("voici·la·table·de·"·+·str(i)·+':')
308|·····for·j·in·range(11):
309|·········print(str(i)·+·'·*·'·+·str(j)·+·'·=·'·+·str(i*j))
310|·
311|·
312|·def·affiche_table_multiplication(entier):
313|·····i·=·entier
314|·····print("voici·la·table·de·"·+·str(i)·+':')
315|·····for·j·in·range(11):
316|·········print(str(i)·+·'·*·'·+·str(j)·+·'·=·'·+·str(i*j))
317|·
318|·
319|·##·écrire·une·punition·de·1000·lignes·(à·exécuter·en·mode·script·dans·un·autre
fichier)
320|·
321|·"""·idée·1:·avec·du·copier-coller.·Au·bout·de·10·copier-coller,·on·obtient·2^10
=·1024·lignes!!"""
322|·
323|·"""idée·2:·on·écrit·dans·un·fichier"""
324|·
325|·
326|·num_ligne·=·1
327|·texte·=·"Voici·la·ligne·n°"·+·str(num_ligne)·+·"·de·ma·punition.\n"
328|·
329|·
330|·f·=·open("ma_punition.txt",'w')·
```

```
for num_ligne in range(1,1001):
    texte = "Voici la ligne n°" + str(num_ligne) + " de ma punition.\n"
    f.write(texte)
f.close()
```