e1.* Décrivez le plus clairement et le plus complètement possible ce qui se passe à chacune destrois lignes de l'exemple ci-dessous :

```
>>> largeur = 20
>>> hauteur = 5 * 9.3
>>> largeur * hauteur
930
```

e2.* Assignez les valeurs respectives 3, 5, 7 à trois variables a, b, c.
Effectuez l'opération a - b/c. Le résultat est-il mathématiquement correct ?Si ce n'est pas le cas, comment devez-vous procéder pour qu'il le soit.

e3.* Testez les lignes d'instructions suivantes. Décrivez dans votre cahier ce qui se passe :

```
>>> r, pi = 12, 3.14159
>>> s = pi, r**2
>>> print(s)
>>> print( type(s), type(pi), type(r) )
```

Quel est, à votre avis, l'utilité de la fonction type () ?
(Note : les fonctions seront décrites en détail, plus loin dans ce cours)

e4.* Ecrivez un programme qui affiche les 20 premiers termes de la table de multiplication par 7.

e5.* Ecrivez un programme qui affiche une table de conversion de sommes d'argent exprimées en euros, en dollars canadiens. La progression des sommes de la table sera "géométrique", comme dans l'exemple ci-dessous :

1 euro (s) = 1.65 dollar (s)

2 euro (s) = 3.30 dollar (s)

4 euro (s) = 6.60 dollar (s)

8 euro (s) = 13.20 dollar (s)
etc. (S'arrêter à 16384 euros)

e6.* Ecrivez un programme qui affiche une suite de 12 nombres dont chaque terme soit égal au triple du terme précédent.

e7.* Ecrivez un programme qui calcule le volume d'un parallélépipède rectangle dont sont fournisau départ la largeur, la hauteur et la profondeur.

e8.* Ecrivez un programme qui convertisse un nombre entier de secondes fourni au départ, en unnombre d'années, de mois, de jours, de minutes et de secondes. (Utilisez l'opérateur modulo : %).

e9.** Ecrivez un programme qui affiche les 20 premiers termes de la table de multiplication par 7,en signalant au passage (à l'aide d'un astérisque) ceux qui sont des multiples de 3.
Exemple : 7 14 21 * 28 35 42 * 49

e10. ** Ecrivez un programme qui calcule les 50 premiers termes de la table de multiplication par13, mais n'affiche que ceux qui sont des multiples de 7.

e11. ** Ecrivez un programme qui affiche la suite de symboles suivante :

```
*
**
***
****
*****
******
*******
********
```

e12.* Écrivez un programme qui convertisse en radians un angle fourni au départ en degrés, minutes, secondes.

e13.* Écrivez un programme qui convertisse en degrés, minutes, secondes un angle fourni au départ en radians.

e14.* Écrivez un programme qui convertisse en degrés, Celsius une température exprimée au départ en degrés Fahrenheit, ou l'inverse. Formule : $T_F = T_C * 1.8 + 32$.

e15.* Écrivez un programme qui calcule les intérêts accumulés chaque année pendant 20 ans, par capitalisation d'une somme de 100 euros placée en banque au taux fixe de 4,3 %

e16.*Une légende de l'Inde ancienne raconte que le jeu d'échecs a été inventé par un vieux sage, que son roi voulut remercier en lui affirmant qu'il lui accorderait n'importe quel cadeau en récompense. Le vieux sage demanda qu'on lui fournisse simplement un peu de riz pour ses vieux jours, et plus précisément un nombre de grains de riz suffisant pour que l'on puisse en déposer 1 seul sur la première case du jeu qu'il venait d'inventer, deux sur la suivante, quatre sur la troisième, et ainsi de suite jusqu'à la 64 case.

Écrivez un programme Python qui affiche le nombre de grains à déposer sur chacune des 64 cases du jeu. Calculez ce nombre de deux manières :

- Le nombre exact de grains (nombre entier)
- Le nombre de grains en notation scientifique (nombre réel)

e17.*Écrivez un script qui détermine si une chaîne contient ou non le caractère "e".

e18. * Écrivez un script qui compte le nombre d'occurrences du caractère "e" dans une chaîne.

e 19.* Écrivez un script qui recopie une chaîne (dans une nouvelle variable), en insérant des astérisques entre les caractères.

Ainsi par exemple, "gaston" devra devenir "g*a*s*t*o*n"

e 20. * Écrivez un script qui recopie une chaîne (dans une nouvelle variable) en l'inversant.

Ainsi par exemple, "zorglub" deviendra "bulgroz".

e 21. * En partant de l'exercice précédent, écrivez un script qui détermine si une chaîne de caractères donnée est un palindrome (c.à.d. une chaîne qui peut se lire indifféremment dans les deux sens), comme par exemple "radar" ou "sos".

e22.* Soient les listes suivantes :

t1 = [31,28,31,30,31,30,31,31,30,31,30,31] t2 = ['Janvier','Février','Mars','Avril','Mai','Juin','Juillet' ,'Août','Septembre','Octobre','Novembre','Décembre']

Écrivez un petit programme qui crée une nouvelle liste t3. Celle-ci devra contenir tous les éléments des deux listes en les alternant, de telle manière que chaque nom de mois soit suivi du nombre de jours correspondant : **['Janvier', 31,'Février', 28,'Mars',31, etc….]**

e23.* Écrivez un programme qui affiche "proprement" tous les éléments d'une liste. Si on l'appliquait par exemple à la liste t2 de l'exercice ci-dessus, on devrait obtenir :

Janvier Février Mars Avril Mai Juin Juillet Août Septembre Octobre Novembre Décembre

e24. * Écrivez un programme qui recherche le plus grand élément présent dans une liste donnée. Par exemple, si on l'appliquait à la liste [32, 5, 12, 8, 3, 75, 2, 15], ce programme devrait afficher :

Le plus grand élément de cette liste a la valeur 75.

e25. * Écrivez un programme qui analyse un par un tous les éléments d'une liste de nombres (par exemple celle de l'exercice précédent) pour générer deux nouvelles listes. L'une contiendra seulement les nombres pairs de la liste initiale, et l'autre les nombres impairs. Par exemple, si la liste initiale est celle de l'exercice précédent, le programme devra construire une liste paire qui contiendra **[32, 12, 8, 2]**, et une liste impaire qui contiendra **[5, 3, 75, 15]**. Astuce : pensez à utiliser l'opérateur modulo (%) déjà cité précédemment.

e 26.* Écrivez un programme qui analyse un par un tous les éléments d'une liste de mots (par exemple : **['Jean', 'Maximilien', 'Brigitte', 'Sonia', 'Jean-Pierre', 'Sandra']** pour générer deux nouvelles listes. L'une contiendra les mots comportant moins de 6 caractères, l'autre les mots comportant 6 caractères ou davantage.

e27.* Écrivez un programme qui convertisse en mètres par seconde et en km/h une vitesse fourniepar l'utilisateur en miles/heure. (Rappel : 1 mile = 1609 mètres)

e 28.* Écrivez un programme qui calcule le périmètre et l'aire d'un triangle quelconque dont l'utilisateur fournit les 3 côtés.(Rappel : l'aire d'un triangle quelconque se calcule à l'aide de la formule : S = RACINECARREE(d .(d-a).(d-b).(d-c)).
dans laquelle d désigne la longueur du demi-périmètre, et a, b, c celles des trois côtés.)

e29.* Écrivez un programme qui calcule la période d'un pendule simple de longueur donnée. La formule qui permet de calculer la période d'un pendule simple est **T = 2.pi. RACINECARREE(l/g)**, l représentant la longueur du pendule et g la valeur de l'accélération de la pesanteur au lieu d'expérience.

e 30.* Écrivez un programme qui permette d'encoder des valeurs dans une liste. Ce programme devrait fonctionner en boucle, l'utilisateur étant invité à entrer sans cesse de nouvelles valeurs, jusqu'à ce qu'il décide de terminer en frappant <enter> en guise d'entrée. Le programme se terminerait alors par l'affichage de la liste. Exemple de fonctionnement :

```
Veuillez entrer une valeur : 25

Veuillez entrer une valeur : 18

Veuillez entrer une valeur : 6284

Veuillez entrer une valeur :
[25, 18, 6284]
```

e31.* Que fait le programme ci-dessous, dans les quatre cas où l'on aurait défini au préalable que la variable a vaut 1, 2, 3 ou 15 ?

```
if != 2 :
  print('perdu')
elif (a==3):
   print ('un instant,s.v.p')

else :
  print('gagné')
```

e32.* Que font ces programmes ?

a)
```
a = 5
b = 2
if (a==5) & (b<2): print (“”&”
        signifie “e”; on peut
        aussi utiliser le mot
        “and” ’)
```

b)
```
a, b = 2, 4
if (a==4) or (b!=4):print
        ('gagné')
elif (a==4) or (b==4): print 'presque
        gagné'
```

c)
```
a = 1
if not a:
        print ('gagné')
elif a:
        print ('perdu')
```

Reprendre le programme c) avec a = 0 au lieu de a = 1. Que se passe-t-il ? Conclure !

e33.** Écrire un programme qui, étant données deux bornes entières a et b, additionne les nombres multiples de 3 et de 5 compris entre ces bornes.
Prendre par exemple a = 0, b = 32 → le résultat devrait être alors 0 + 15 + 30 = 45.
Modifier légèrement ce programme pour qu'il additionne les nombres multiples de 3 ou de 5 compris entre les bornes a et b. Avec les bornes 0 et 32, le résultat devrait donc être : 0 + 3 + 5 + 6 + 9 + 10 + 12 + 15 + 18 + 20 + 21 + 24 + 25 + 27 + 30 = 225.

e34.* Déterminer si une année (dont le millésime est introduit par l'utilisateur) est bissextile ou non. (Une année A est bissextile si A est divisible par 4. Elle ne l'est cependant pas si A est un multiple de 100, à moins que A ne soit multiple de 400).

e35.* Demander à l'utilisateur son nom et son sexe (M ou F). En fonction de ces données, afficher "Cher Monsieur" ou "Chère Mademoiselle" suivi du nom de l'élève.

e36.* Demander à l'utilisateur d'entrer trois longueurs a, b, c. A l'aide de ces trois longueurs, déterminer s'il est possible de construire un triangle. Déterminer ensuite si ce triangle est rectangle, isocèle, équilatéral ou quelconque. Attention : un triangle rectangle peut être isocèle.

e37.* Demander à l'utilisateur qu'il entre un nombre. Afficher ensuite : soit la racine carrée de cenombre, soit un message indiquant que la racine carrée de ce nombre ne peut être calculée.

e38.*Convertir une note scolaire N quelconque, entrée par l'utilisateur sous forme de points (parexemple 27 sur 85), en une note standardisée suivant le code suivant :

Note	Appréciation
N >= 80 %	A
80 % > N >= 60 %	B
60 % > N >= 50 %	C
50 % > N >= 40 %	D
N < 40 %	E

e39.* Soit la liste suivante : **['Jean-Michel', 'Marc', 'Vanessa', 'Anne', Maximilien', 'Alexandre-Benoît', 'Louise']**. Ecrivez un script qui affiche chacun de ces noms avec le nombre de caractères correspondant.

e40.** Écrire une boucle de programme qui demande à l'utilisateur d'entrer des notes d'élèves. La boucle se terminera seulement si l'utilisateur entre une valeur négative. Avec les notes ainsi entrées, construire progressivement une liste. Après chaque entrée d'une nouvelle note (et donc à chaque itération de la boucle), afficher le nombre de notes entrées, la note la plus élevée, la note la plus basse, la moyenne de toutes les notes.

e41.** Ecrivez un script qui affiche la valeur de la force de gravitation s'exerçant entre deux masses de 10000 kg, pour des distances qui augmentent suivant une progression géométrique de raison 2, à partir de 5 cm (0,05 mètre).
La force de gravitation est régie par la formule $F = 6.6710^{-10} .m.m'/d^2$

Exemple d'affichage :

d = .05 m : la force vaut 2.668 N

d = .01 m : la force vaut 0.667 N

d = .02 m : la force vaut 0.167 N

d = .04 m : la force vaut 0.0417 N

etc.

e42.* Définissez une fonction **ligneCar(n, ca)** qui renvoie une chaîne de n caractères ca.

e43.* Définissez une fonction **surfCercle(R)**. Cette fonction doit renvoyer la surface (l'aire) d'un cercle dont on lui a fourni le rayon R en argument. Par exemple, l'exécution de l'instruction : **print(surfCercle(2.5))** doit donner le résultat **19.635**.

e 44.* Définissez une fonction **volBoite(x1,x2,x3)** qui renvoie le volume d'une boîte parallélépipédique dont on fournit les trois dimensions x1, x2, x3 en arguments.
Par exemple, l'exécution de l'instruction :
print(volBoite(x1,x2,x3)) doit donner le résultat.

e45.* Définissez une fonction **maximum(n1,n2,n3)** qui renvoie le plus grand de 3 nombres **n1, n2,n3** fournis en arguments. Par exemple, l'exécution de l'instruction :
print(maximum(2, 5, 4)) doit donner le résultat : **5**

e46.* Définissez une fonction **compteCar(ca,ch)** qui renvoie le nombre de fois que l'on rencontre le caractère ca dans la chaîne de caractères ch. Par exemple, l'exécution de l'instruction :
print(compteCar('e','Cette phrase est un exemple'))doit donner le résultat : **7**

e47.* Définissez une fonction **indexMax(liste)** qui renvoie l'index de l'élément ayant la valeur la plus élevée dans la liste transmise en argument. Exemple d'utilisation :
serie = [5, 8, 2, 1, 9, 3, 6, 7]
print (indexMax(serie)4

e48.* Définissez une fonction **nomMois(n)** qui renvoie le nom du n mois de l'année.
Par exemple, l'exécution de l'instruction :
6 doit donner le résultat : **Avril**

e49.* Définissez une fonction **inverse(ch)** qui permette d'inverser les l'ordre des caractères d'une chaîne quelconque. (La chaîne inversée sera renvoyée au programme appelant).

e50.* Définissez une fonction **compteMots(ph)** qui renvoie le nombre de mots contenus dans la phrase ph (On considère comme mots les ensembles de caractères inclus entre des espaces).

e51.* Modifiez la fonction **volBoite(x1,x2,x3)** que vous avez définie dans un exercice précédent, de manière à ce qu'elle puisse être appelée avec trois, deux, un seul, ou même aucun argument. Utilisez pour ceux-ci des valeurs par défaut égales à) 10.
Par exemple :

print(volBoite())	doit donner le résultat : 1000
print(volBoite(5.2))	doit donner le résultat : 520.0
print(volBoite(5.2,3)	doit donner le résultat : 156.0

e52.** Modifiez la fonction **volBoite(x1,x2,x3)** ci-dessus de manière à ce qu'elle puisse être appeléeavec un, deux, ou trois arguments. Si un seul est utilisé, la boîte est considérée comme cubique (l'argument étant l'arête de ce cube). Si deux sont utilisés, la boîte est considérée comme un prisme à base carrée. (Dans ce cas le premier argument est le côté du carré, et le second la hauteur du prisme). Si trois arguments sont utilisés, la boîte est considérée comme un parallélépipède. Par exemple :

print (volBoite())	doit donner le résultat : -1 (→ indication d'une erreur).
print (volBoite(5.2))	doit donner le résultat : 140.608
print (volBoite(5.2, 3))	doit donner le résultat : 81.12
print (volBoite(5.2, 3, 7.4))	doit donner le résultat : 115.44

e53.* Définissez une fonction **changeCar(ch,ca1,ca2,debut,fin)** qui remplace tous les caractères ca1 par des caractères ca2 dans la chaîne de caractères ch, à partir de l'indice debut et jusqu'à l'indice fin, ces deux derniers arguments pouvant être omis (et dans ce cas la chaîne est traitée d'une extrémité à l'autre). Exemples de la fonctionnalité attendue :

```
>>> phrase = 'Ceci est une toute petite phrase.'
>>> print changeCar(phrase, ' ', '*')
Ceci*est*une*toute*petite*phrase.
>>> print changeCar(phrase, ' ', '*', 8, 12)
Ceci est*une*toute petite phrase.
>>> print changeCar(phrase, ' ', '*', 12)
Ceci est une*toute*petite*phrase.
>>> print changeCar(phrase, ' ', '*', fin = 12)
Ceci*est*une*toute petite phrase.
```

e54.* Définissez une fonction eleMax(liste,debut,fin) qui retourne l'élément ayant la plus grande valeur dans la liste transmise. Les deux arguments debut et fin indiqueront les indices entre lesquels doit s'exercer la recherche, et chacun d'eux pourra être omis (comme dans l'exercice précédent). Exemples de la fonctionnalité attendue :

```
>>> serie = [9, 3, 6, 1, 7, 5, 4, 8, 2]
>>>print(eleMax(serie))
9
>>> print eleMax(serie, 2, 5)
7
>>> print eleMax(serie, 2)8
>>> print eleMax(serie, fin =3, debut =1)
6
```

e55.** Écrivez un script qui permette de créer et de relire aisément un fichier texte. Votre programme demandera d'abord à l'utilisateur d'entrer le nom du fichier. Ensuite il lui proposera le choix, soit d'enregistrer de nouvelles lignes de texte, soit d'afficher le contenu du fichier.

L'utilisateur devra pouvoir entrer ses lignes de texte successives en utilisant simplement la touche <Enter> pour les séparer les unes des autres. Pour terminer les entrées, il lui suffira d'entrer une ligne vide (c.à.d. d'utiliser la touche <Enter> seule).

L'affichage du contenu devra montrer les lignes du fichier séparées les unes des autres de la manière la plus naturelle (les codes de fin de ligne ne doivent pas apparaître).

e56.** Considérons que vous avez à votre disposition un fichier texte contenant des phrases de différentes longueurs. Écrivez un script qui recherche et affiche la phrase la plus longue.

e57.** Écrivez un script qui génère automatiquement un fichier texte contenant les tables de multiplication de 2 à 30 (chacune d'entre elles incluant 20 termes seulement).

e 58. ** Écrivez un script qui recopie un fichier texte en triplant tous les espaces entre les mots.

e59.** Vous avez à votre disposition un fichier texte dont chaque ligne est la représentation d'une valeur numérique de type réel (mais sans exposants). Par exemple :
14.896
7894.6
123.278
etc.
Écrivez un script qui recopie ces valeurs dans un autre fichier en les arrondissant en nombres entiers (l'arrondi doit être correct).

e60.** Écrivez un script qui compare les contenus de deux fichiers et signale la première différence rencontrée.

e61.*** A partir de deux fichiers préexistants A et B, construisez un fichier C qui contienne alternativement un élément de A, un élément de B, un élément de A, ... et ainsi de suite jusqu'à atteindre la fin de l'un des deux fichiers originaux. Complétez ensuite C avec les éléments restant sur l'autre.

e62.** Écrivez un script qui permette d'encoder un fichier texte dont les lignes contiendront chacune les noms, prénom, adresse, code postal et n° de téléphone de différentes personnes (considérez par exemple qu'il s'agit des membres d'un club)

e63.** Écrivez un script qui recopie le fichier utilisé dans l'exercice précédent, en y ajoutant la date de naissance et le sexe des personnes (l'ordinateur devra afficher les lignes une par une, et demander à l'utilisateur d'entrer pour chacune les données complémentaires).

e64.*** Considérons que vous avez fait les exercices précédents et que vous disposez à présent d'un fichier contenant les coordonnées d'un certain nombre de personnes. Écrivez un script qui permette d'extraire de ce fichier les lignes qui correspondent à un code postal bien déterminé.

e65.*** Modifiez le script de l'exercice précédent, de manière à retrouver les lignes correspondant à des prénoms dont la première lettre est située entre F et M (inclus) dans l'alphabet.

e66.*** Écrivez des fonctions qui effectuent le même travail que celles du module pickle. Ces fonctions doivent permettre l'enregistrement de variables diverses dans un fichier texte, en les accompagnant systématiquement d'informations concernant leur format exact.

e67.* Déterminez vous-même ce qui se passe lorsque l'un ou l'autre des indices de découpage est erroné, et décrivez cela le mieux possible. (Si le second indice plus petit que le premier, par exemple, ou bien si le second indice est plus grand que la taille de la chaîne).

e68.** Découpez une grande chaîne en fragments de 5 caractères chacun. Rassemblez ces morceaux dans l'ordre inverse.

e69.* Tâchez d'écrire une petite fonction trouve() qui fera exactement le contraire de ce que fait l'opérateur d'indexage (c.à.d. les crochets []). Au lieu de partir d'un index donné pour retrouver le caractère correspondant, cette fonction devra retrouver l'index correspondant à un caractère donné.
En d'autres termes, il s'agit d'écrire une fonction qui attend deux arguments : le nom de la chaîne à traiter et le caractère à trouver. La fonction doit fournir en retour l'index du premier caractère de ce type dans la chaîne. Ainsi par exemple, l'instruction :

print(trouve (“Juliette & Roméo”, “&”)) devra afficher : 9

Attention : Il faut penser à tous les cas possibles. Il faut notamment veiller à ce que la fonction renvoie une valeur particulière (par exemple la valeur -1) si le caractère recherché n'existe pas dans la chaîne traitée.

e70.** Améliorez la fonction de l'exercice précédent en lui ajoutant un troisième paramètre : l'index à partir duquel la recherche doit s'effectuer dans la chaîne. Ainsi par exemple, l'instruction :
print (trouve("César & Cléopâtre", "r", 5)) devra afficher : 15 (et non 4 !)

e71.** Écrivez une fonction **comptecar()** qui compte le nombre d'occurrences d'un caractère donné dans une chaîne. Ainsi l'instruction :
print(comptecar("Ananas au jus", "a")) devra afficher : 4

e72.* Dans un conte américain, huit petits canetons s'appellent respectivement : Jack, Kack, Lack, Mack, Nack, Oack, Pack et Qack. Écrivez un script qui génère tous ces noms à partir des deux chaînes suivantes :
prefixes = 'JKLMNOP' et suffixe = 'ack'
Si vous utilisez une instruction for ... in ... , votre script ne devrait comporter que deux lignes.

e73.* Rechercher le nombre de mots contenus dans une phrase donnée.

e74.* Écrivez une fonction majuscule() qui retourne "vrai" si l'argument transmis est une majuscule.

e75.* Écrivez une fonction qui retourne "vrai" si l'argument transmis est un chiffre.

e.76.* Écrivez une fonction qui convertit une phrase en une liste de mots.

e77.** Utilisez les fonctions définies dans les exercices précédents pour écrire un script qui puisse extraire d'un texte tous les mots qui commencent par une majuscule.

e78.* Écrivez une fonction majuscule() qui renvoie "vrai" si l'argument transmis est une majuscule (utilisez une autre méthode que celle exploitée précédemment)

e79.* Écrivez une fonction qui renvoie "vrai" si l'argument transmis est un caractère alphabétique quelconque (majuscule ou minuscule). Dans cette nouvelle fonction, utilisez les fonctions minuscule() et majuscule() définies auparavant.

e 80.* Écrivez une fonction qui renvoie "vrai" si l'argument transmis est un chiffre.

e81.* Écrivez une fonction qui renvoie le nombre de caractères majuscules contenus dans une phrase donnée en argument.
Afin que vous puissiez effectuer plus aisément toutes sortes de traitements sur les caractères, python met à votre disposition un certain nombre de fonctions prédéfinies : La fonction ord(ch) accepte n'importe quel caractère comme argument. En retour, elle fournit le code ASCII correspondant à ce caractère. Ainsi ord('A') renvoie la valeur 65. La fonction chr(num) fait exactement le contraire. L'argument qu'on lui transmet doit être un entier compris entre 0 et 255. En retour, on obtient le caractère ASCII correspondant : ainsi chr(65) renvoie le caractère A.

e82.** Écrivez un petit script qui affiche une table des codes ASCII. Le programme doit afficher tous les caractères en regard des codes correspondants. A partir de cette table, établissez les relations numériques reliant chaque caractère majuscule à chaque caractère minuscule.

e83.* A partir des relations trouvées dans l'exercice précédent, écrivez une fonction qui convertit tous les caractères d'une phrase donnée en minuscules.

e 84.* A partir des mêmes relations, écrivez une fonction qui convertit tous les caractères minuscules en majuscules, et vice-versa (dans une phrase fournie en argument).

e85.* Écrivez une fonction qui compte le nombre de fois qu'apparaît tel caractère (fourni en argument) dans une phrase donnée.

e86.* Écrivez une fonction qui renvoie le nombre de voyelles contenues dans une phrase donnée.

e87.* Écrivez un script qui compte dans un fichier texte quelconque le nombre de lignes contenant des caractères numériques.

e88.* Écrivez un script qui compte le nombre de mots contenus dans un fichier texte.

e89.** Écrivez un script qui recopie un fichier texte en veillant à ce que chaque ligne commence par une majuscule.

e90.*** Écrivez un script qui recopie un fichier texte en fusionnant (avec la précédente) les lignes qui ne commencent pas par une majuscule.

e91.*** Vous disposez d'un fichier contenant des valeurs numériques. Considérez que ces valeurs sont les diamètres d'une série de sphères. Écrivez un script qui utilise les données de ce fichier pour en créer un autre, organisé en lignes de texte qui exprimeront "en clair" les autres caractéristiques de ces sphères (surface de section, surface extérieure et volume), dans des phrases telles que :

```
Diam.  46.20 cm Section =  1676.39 cm² Surf. =  6705.54 cm². Vol. =  38724.50 cm³
Diam. 120.00 cm Section = 11309.73 cm² Surf. = 45238.93 cm². Vol. = 678584.01 cm³
Diam.   0.03 cm Section =     0.00 cm² Surf. =     0.00 cm². Vol. =      0.00 cm³
Diam.  13.90 cm Section =   151.67 cm² Surf. =   606.70 cm². Vol. =   1053.89 cm³
Diam.  88.80 cm Section =  6193.21 cm² Surf. = 24772.84 cm². Vol. = 274978.53 cm³
etc.
```

e92.** Vous avez à votre disposition un fichier texte dont les lignes représentent des valeurs numériques de type réel, sans exposant (et encodées sous forme de chaînes de caractères). Écrivez un script qui recopie ces valeurs dans un autre fichier en les arrondissant de telle sorte que leur partie décimale ne comporte plus qu'un seul chiffre après la virgule, celui-ci ne pouvant être que 0 ou 5 (l'arrondi doit être correct).

e93.* Écrivez un script qui génère la liste des carrés et des cubes des nombres de 20 à 40.

e94.* Écrivez un script qui crée automatiquement la liste des sinus des angles de 0° à 90°, par pas de 5°. Attention : la fonction sin() du module math considère que les angles sont fournis en radians (360° = 2 radians).

e95.* Écrivez un script qui permette d'obtenir à l'écran les 15 premiers termes des tables de multiplication par 2, 3, 5, 7, 11, 13, 17, 19 (ces nombres seront placés au départ dans une liste) sous la forme d'une table similaire à la suivante :

```
2   4   6   8  10  12  14  16  18  20  22  24  26  28  30
3   6   9  12  15  18  21  24  27  30  33  36  39  42  45
5  10  15  20  25  30  35  40  45  50  55  60  65  70  75
etc.
```

e96.* Vous disposez d'une liste de nombres entiers quelconques, certains d'entre eux étant présents en plusieurs exemplaires. Écrivez un script qui recopie cette liste dans une autre, en omettant les doublons. La liste finale devra être triée.

e97.* Écrivez un script qui recherche le mot le plus long dans une phrase donnée (l'utilisateur du programme doit pouvoir entrer une phrase de son choix).

e98.* Écrivez un script capable d'afficher la liste de tous les jours d'une année imaginaire, laquelle commencerait un Jeudi. Votre script utilisera lui-même trois listes : une liste des noms de jours de la semaine, une liste des noms des mois, et une liste des nombres de jours que comportent chacun des mois (ne pas tenir compte des années bissextiles).

Exemple de sortie :

```
Jeudi 1 Janvier   Vendredi 2 Janvier   Samedi 3 Janvier   Dimanche 4 Janvier
```
... et ainsi de suite jusqu'au Jeudi 31 Décembre.

e99.** Vous avez à votre disposition un fichier texte qui contient un certain nombre de noms d'élèves. Écrivez un script qui effectue une copie triée de ce fichier.

e100.** Écrivez une fonction permettant de trier une liste. Cette fonction ne pourra pas utiliser la méthode intégrée sort() de Python : Vous devez donc définir vous-même l'algorithme de tri. (Note : cette question devra faire l'objet d'une discussion-synthèse en classe)

e101.** Soient les listes suivantes :

```
t1 = [31,28,31,30,31,30,31,31,30,31,30,31]
t2 = ['Janvier','Février','Mars','Avril','Mai','Juin',
 'Juillet','Août','Septembre','Octobre','Novembre','Décembre']
```

Écrivez un petit programme qui insère dans la seconde liste tous les éléments de la première, de telle sorte que chaque nom de mois soit suivi du nombre de jours correspondant :

```
['Janvier',31,'Février',28,'Mars',31, etc...].
```

e102.** Créez une liste A contenant quelques éléments. Effectuez une vraie copie de cette liste dans une nouvelle variable B. Suggestion : créez d'abord une liste B de même taille que A mais ne contenant que des zéros. Remplacez ensuite tous ces zéros par les éléments tirés de A.

e103.** Même question, mais autre suggestion : créez d'abord une liste B vide. Remplissez-la ensuite à l'aide des éléments de A ajoutés l'un après l'autre.

e104.** Même question, autre suggestion encore : pour créer la liste B, découpez dans la liste A une tranche incluant tous les éléments (à l'aide de l'opérateur [:]).

e105.** Un nombre premier est un nombre qui n'est divisible que par un et par lui-même. Écrivez un programme qui établisse la liste de tous les nombres premiers compris entre 1 et 1000, en utilisant la méthode du crible d'Eratosthène :

Créez une liste de 1000 éléments, chacun initialisé à la valeur 1. Parcourez cette liste à partir de l'élément d'indice 2 : si l'élément analysé possède la valeur 1, mettez à zéro tous les autres éléments de la liste, dont les indices sont des multiples entiers de l'indice auquel vous êtes arrivé. Lorsque vous aurez parcouru ainsi toute la liste, les indices des éléments qui seront restés à 1 seront les nombres premiers recherchés. En effet : A partir de l'indice 2, vous annulez tous les éléments d'indices pairs : 4, 6, 8, 10, etc. Avec l'indice 3, vous annulez les éléments d'indices 6, 9, 12, 15, etc., et ainsi de suite. Seuls resteront à 1 les éléments dont les indices sont effectivement des nombres premiers.

e106.** Réécrivez la fonction list_aleat() ci-dessus, en utilisant la méthode append() pour construire la liste petit à petit à partir d'une liste vide (au lieu de remplacer les zéros d'une liste préexistante comme nous l'avons fait).

e107.** Ecrivez une fonction imprime_liste() qui permette d'afficher ligne par ligne tous les éléments contenus dans une liste de taille quelconque. Le nom de la liste sera fourni en argument. Utilisez cette fonction pour imprimer la liste de nombres aléatoires générés par la fonction list_aleat(). Ainsi par exemple, l'instruction imprime_liste(liste_aleat(8)) devra afficher une colonne de 8 nombres réels aléatoires.

e108.** Vous allez écrire un programme destiné à vérifier le fonctionnement du générateur de nombres aléatoires de Python en appliquant la théorie exposée ci-dessus.

Votre programme devra donc :

Demander à l'utilisateur le nombre de valeurs à tirer au hasard à l'aide de la fonction random(). Il serait intéressant que le programme propose un nombre par défaut (1000 par exemple).

Demander à l'utilisateur en combien de fractions il souhaite partager l'intervalle des valeurs possibles (c.à.d. l'intervalle de 0 à 1). Ici aussi, il faudrait proposer un nombre de par défaut (5 fractions, par exemple). Vous pouvez également limiter le choix de l'utilisateur à un nombre compris entre 2 et le 1/10 du nombre de valeurs tirées au hasard.

Construire une liste de N compteurs (N étant le nombre de fractions souhaitées). Chacun d'eux sera évidemment initialisé à zéro.

Tirer au hasard toutes les valeurs demandées, à l'aide de la fonction random() , et mémoriser ces valeurs dans une liste.

Mettre en oeuvre un parcours de la liste des valeurs tirées au hasard (boucle), et effectuer un test sur chacune d'elles pour déterminer dans quelle fraction de l'intervalle 0-1 elle se situe. Incrémenter d'une unité le compteur correspondant.

Lorsque c'est terminé, afficher l'état de chacun des compteurs.

Exemple de résultats affichés par un programme de ce type :

Nombre de valeurs à tirer au hasard (défaut = 1000) : 100

Nombre de fractions dans l'intervalle 0-1 (entre 2 et 10, défaut =5) : 5 Tirage au sort des 100 valeurs ...

Comptage des valeurs dans chacune des 5 fractions ... 11 30 25 14 20

Nombre de valeurs à tirer au hasard (défaut = 1000) : 10000

Nombre de fractions dans l'intervalle 0-1 (entre 2 et 1000, défaut =5) : 5 Tirage au sort des 10000 valeurs ...

Comptage des valeurs dans chacune des 5 fractions ... 1970 1972 2061 1935 2062

e109.* Créez un dictionnaire qui contienne les noms d'une série d'élèves, leur âge et leur taille. Le nom de l'élève servira de clé d'accès. Exprimez l'âge en années (nombre entier), et exprimez la taille en mètres (ainsi vous devrez employer pour celle-ci une variable de type float !). Écrivez un petit script qui affiche le contenu de ce dictionnaire en utilisant le formatage des chaînes de caractères.

e110.**Écrivez une fonction qui échange les clés et les valeurs d'un dictionnaire (ce qui permettra par exemple de transformer un dictionnaire anglais/français en un dictionnaire français/anglais).(On suppose que le dictionnaire ne contient pas plusieurs valeurs identiques).

e111.** Vous avez à votre disposition un fichier texte quelconque (pas trop gros). Écrivez un script qui compte les occurrences de chacune des lettres de l'alphabet dans ce texte (on ne tiendra pas compte du problème des lettres accentuées)..

e112.*** Modifiez le script ci-dessus afin qu'il établisse une table des occurrences de chaque mot dans le texte. Conseil : dans un texte quelconque, les mots ne sont pas seulement séparés par des espaces, mais également par divers signes de ponctuation. Pour simplifier le problème, vous pouvez commencer par remplacer tous les caractères non-alphabétiques par des espaces, et convertir la chaîne résultante en une liste de mots à l'aide de la méthode split().

CLASSE

e113.** Ecrivez une fonction distance() qui permette de calculer la distance entre deux points.
Cette fonction attendra évidemment deux objets Point() comme arguments.

e114.** Définissez une classe Domino() qui permette d'instancier des objets simulant les pièces d'un jeu de dominos. Le constructeur de cette classe initialisera les valeurs des points présents sur les deux faces A et B du domino (valeurs par défaut = 0).
Deux autres méthodes seront définies :
une méthode affiche_points() qui affiche les points présents sur les deux faces
une méthode valeur() qui renvoie la somme des points présents sur les 2 faces
Exemples d'utilisation de cette classe :

```
>>> d1 = Domino(2,6)
>>> d2 = Domino(4,3)
>>> d1.affiche_points()
face A : 2  face B : 6
>>> d2.affiche_points()
face A : 4  face B : 3
print "total des points :", d1.valeur() + d2.valeur()
15
>>> liste_dominos = []
>>> for i in range(7):
        liste_dominos.append(Domino(6, i))

>>> print liste_dominos

          etc., etc.
```

e115.** Définissez une classe CompteBancaire(), qui permette d'instancier des objets tels que compte1, compte2, etc. Le constructeur de cette classe initialisera deux attributs d'instance nom et solde, avec les valeurs par défaut 'Dupont' et 1000.

Trois autres méthodes seront définies :

- depot(somme) permettra d'ajouter une certaine somme au solde
- retrait(somme) permettra de retirer une certaine somme du solde
- affiche() permettra d'afficher le nom du titulaire et le solde de son compte. Exemples d'utilisation de cette classe :

```
>>> compte1 = CompteBancaire('Duchmol', 800)
>>> compte1.depot(350)
>>> compte1.retrait(200)
>>> compte1.affiche()
Le solde du compte bancaire de Duchmol est de 950 euros.
>>> compte2 = CompteBancaire()
>>> compte2.depot(25)
>>> compte2.affiche()
Le solde du compte bancaire de Dupont est de 1025 euros.
```

e116.** Définissez une classe Voiture() qui permette d'instancier des objets reproduisant le comportement de voitures automobiles. Le constructeur de cette classe initialisera les attributs d'instance suivants, avec les valeurs par défaut indiquées :
marque = 'Ford', couleur = 'rouge', pilote = 'personne', vitesse = 0.
Lorsque l'on instanciera un nouvel objet Voiture(), on pourra choisir sa marque et sa couleur, mais pas sa vitesse, ni le nom de son conducteur.
Les méthodes suivantes seront définies :
- choix_conducteur(nom) permettra de désigner (ou changer) le nom du conducteur
- accelerer(taux, duree) permettra de faire varier la vitesse de la voiture. La variation de vitesse obtenue sera égale au produit : taux x duree. Par exemple, si la voiture accélère au taux de 1,3 m/s pendant 20 secondes, son gain de vitesse doit être égal à 26 m/s. Des taux négatifs seront acceptés (ce qui permettra de décélérer). La variation de vitesse ne sera pas autorisée si le conducteur est 'personne'.
- affiche_tout() permettra de faire apparaître les propriétés présentes de la voiture, c.à.d. sa marque, sa couleur, le nom de son conducteur, sa vitesse.

Exemples d'utilisation de cette classe :

```
>>> a1 = Voiture('Peugeot', 'bleue')
>>> a2 = Voiture(couleur = 'verte')
>>> a3 = Voiture('Mercedes')
>>> a1.choix_conducteur('Roméo')
>>> a2.choix_conducteur('Juliette')
>>> a2.accelerer(1.8, 12)
>>> a3.accelerer(1.9, 11)
Cette voiture n'a pas de conducteur !
>>> a2.affiche_tout()
Ford verte pilotée par Juliette, vitesse = 21.6 m/s.
>>> a3.affiche_tout()
Mercedes rouge pilotée par personne, vitesse = 0 m/s.
```

e117.** Définissez une classe Satellite() qui permette d'instancier des objets simulant des satellites artificiels lancés dans l'espace, autour de la terre. Le constructeur de cette classe initialisera les attributs d'instance suivants, avec les valeurs par défaut indiquées :
masse = 100, vitesse = 0.
Lorsque l'on instanciera un nouvel objet Satellite(), on pourra choisir son nom, sa masse et sa vitesse.
Les méthodes suivantes seront définies :
- impulsion(force, duree) permettra de faire varier la vitesse du satellite. Pour savoir comment, rappelez-vous votre cours de physique : la variation de vitesse subie par un objet de masse m soumis à l'action d'une force F pendant un temps t vaut ..... Par exemple : un satellite de 300 kg qui subit une force de 600 Newtons pendant 10 secondes voit sa vitesse augmenter (ou diminuer) de 20 m/s.
- affiche_vitesse() affichera le nom du satellite et sa vitesse courante.
- energie() renverra au programme appelant la valeur de l'énergie cinétique du satellite.

Rappel : l'énergie cinétique se calcule à l'aide de la formule $E_c = \frac{m \times v^2}{2}$

Exemples d'utilisation de cette classe :

```
>>> s1 = Satellite('Zoé', masse =250, vitesse =10)
>>> s1.impulsion(500, 15)
>>> s1.affiche_vitesse()
vitesse du satellite Zoé = 40 m/s.
>>> print s1.energie()
200000
>>> s1.impulsion(500, 15)
>>> s1.affiche_vitesse()
vitesse du satellite Zoé = 70 m/s.
>>> print s1.energie()
612500
```

e118.** Définissez une classe Cercle(). Les objets construits à partir de cette classe seront des cercles de tailles variées. En plus de la méthode constructeur (qui utilisera donc un paramètre rayon), vous définirez une méthode surface(), qui devra renvoyer la surface du cercle.

Définissez ensuite une classe Cylindre() dérivée de la précédente. Le constructeur de cette nouvelle classe comportera les deux paramètres rayon et hauteur. Vous y ajouterez une méthode volume() qui devra renvoyer le volume du cylindre.
(Rappel : Volume d'un cylindre = surface de section x hauteur).

Exemple d'utilisation de cette classe :

```
>>> cyl = Cylindre(5, 7)
>>> print (cyl.surface() )
78.54
>>> print( cyl.volume())
 549.78
```

e119.** Complétez l'exercice précédent en lui ajoutant encore une classe Cone(), qui devra dériver cette fois de la classe Cylindre(), et dont le constructeur comportera lui aussi les deux paramètres rayon et hauteur. Cette nouvelle classe possédera sa propre méthode volume(), laquelle devra renvoyer le volume du cône.
(Rappel : Volume d'un cône = volume du cylindre correspondant divisé par 3).
Exemple d'utilisation de cette classe :

```
>>> co = Cone(5,7)
>>> pri
183.26
```