

Séries d'exercices des structures de données 1

Exercice 1

Ecrire un programme qui supprime toutes les occurrences du nombre 20 de la liste suivante :

[5, 20, 15, 20, 25, 50, 20]

Exercice 2

Ecrire un programme qui affiche les éléments d'une liste qui sont supérieurs à leurs précédents.

Entrée : [11, 8, 85, 4, 6, 3, 45, 9, 5, 47]

Sortie : 85, 6, 45, 47

Exercice 3

Ecrire un programme qui permute les éléments voisins, par exemple dans une liste A on échange A[0] et A[1], A[2] et A[3], etc..., si la liste a un nombre impair d'éléments on laisse le dernier élément à sa place.

Entrée : ['Red', 'Orange', 'Yellow', 'Green', 'Blue', 'Indigo', 'Violet']

Sortie : ['Orange', 'Red', 'Green', 'Yellow', 'Indigo', 'Blue', 'Violet']

Exercice 4

Ecrire un programme qui déplace tous les zéros vers la fin de la liste.

Entrée :

[3, 4, 0, 0, 0, 6, 2, 0, 6, 7, 6, 0, 0, 0, 9, 10, 4, 0, 0, 2, 9, 7, 1]

Sortie :

[3, 4, 6, 2, 6, 7, 6, 9, 10, 4, 2, 9, 7, 1, 0, 0, 0, 0, 0, 0, 0, 0]

Exercice 5

Ecrire un programme qui affiche la moyenne des éléments d'un tuple dans une liste

Entrée : ((10, 10, 10, 12), (30, 45, 56, 45), (81, 80, 39, 32), (1, 2, 3, 4))

Sortie : [10.5, 44.0, 58.0, 2.5]

Exercice 6

Ecrire un programme qui crée une liste contenant la somme des éléments des tuples dans une liste de tuples.

Entrée : [(1, 2, 6), (2, 3, -6), (3, 4), (2, 2, 2, 2)]

Sortie : [9, -1, 7, 8]

Exercice 7

Ecrire un programme qui remplace la dernière valeur des tuples dans une liste de tuples.

Entrée : [(10, 20, 40), (40, 50, 60), (70, 80, 90)]

Sortie : [(10, 20, 100), (40, 50, 100), (70, 80, 100)]

Exercice 8

Ecrire un programme qui supprime les éléments dupliqués dans un tuple.

Exercice 9

Ecrire un programme qui concatène les éléments des dictionnaires suivant dans un nouveau dictionnaire :

dic1={1:10, 2:20}

dic2={3:30, 4:40}

dic3={5:50, 6:60}

Exercice 10

Ecrire un programme qui convertit une chaîne de caractères en dictionnaire ayant pour clé les lettres et pour valeur le nombre d'occurrences de la clé.

Entrée : "Bonjour"

Sortie : {'B': 1, 'o': 2, 'n': 1, 'j': 1, 'u': 1, 'r': 1}

Exercice 11

Ecrire un programme qui élimine les valeurs vides d'un dictionnaire.

Entrée : {'c1': 'Red', 'c2': 'Green', 'c3': None}

Sortie : {'c1': 'Red', 'c2': 'Green'}

Exercice 12

Ecrire un programme qui ajoute les éléments du set2 dans set1 sauf les éléments en commun.

Entrées : set1 = {10, 20, 30, 40, 50}

set2 = {30, 40, 50, 60, 70}

Sortie : {70, 10, 20, 60}

Exercice 13

Ecrire une fonction qui prend en paramètre une chaîne de caractères et renvoie le nombre de mots qu'elle contient.

Exercice 14

Ecrire une fonction qui accepte comme paramètre une chaîne de caractères et qui retourne le nombre de caractères majuscules dans cette chaîne.

Exercice 15

Ecrire un programme qui compte les caractères répétés dans une chaîne.

Entrée : 'thequickbrownfoxjumpsoverthelazydog'

Sortie : o=4, e=3, u=2, h=2, r=2, t=2

Exercice 16

Ecrire une fonction qui accepte comme paramètre une chaîne de caractères et qui retourne le nombre de chiffres dans cette chaîne.

Exercice 17

Ecrire un programme qui compte les voyelles et affiche leur nombre dans un texte donné par l'utilisateur.

Exercice 18

Ecrire une fonction qui prend en paramètres une liste de mots en renvoie le mot le plus long avec sa longueur sous forme de tuple.

Exercice 19

Ecrire une fonction qui renvoie le nombre d'occurrences d'un mot dans une phrase. Le mot et la phrase doivent être des paramètres de la fonction.

Exercice 20

Ecrire un programme qui permet de déclarer un dictionnaire de données contenant des pays, chaque pays est associé à plusieurs villes, et chaque ville à un entier qui représente sa population. La structure du dictionnaire est la suivante :

```
Dict{'NomPays1' : {'Ville1' : Population1, 'Ville2' :  
Population2, .....}, .....}
```

Le programme doit avoir les fonctions suivantes :

- 1) Une procédure d'ajout de pays qui prend en paramètre le nom d'un pays et donne la possibilité d'ajouter des villes.
- 2) Une procédure pour ajouter des villes à un pays.
- 3) Une procédure pour modifier la population d'une ville passée en paramètre.
- 4) Une procédure pour supprimer un pays, ou une ville.
- 5) Une fonction qui retourne la population d'un pays passé en paramètre.
- 6) Une fonction qui retourne la ville la plus peuplée d'un pays.
- 7) Afficher les fonctions précédentes dans un menu.

Séries d'exercices des structures de données 2

Exercice 1

Ecrire un programme qui traite un dictionnaire avec différents produits et leurs prix respectifs et renvoie un dictionnaire des produits avec un prix minimum de 400 dans l'ordre décroissant.

Entrée : {"Computer" : 600, "TV" : 800, "Radio" : 50, "Tablet" : 450, "Smartphone" : 700, "Router" : 250}

Exercice 2

Soit est un dictionnaire avec l'alphabet Morse. Ecrivez une fonction qui prend une chaîne, soit en lettres, soit en code Morse du dictionnaire. La fonction génère une chaîne de lettres chiffrée en code Morse. Le code morse des lettres doit être séparé d'un espace.

A	• —	U	• • —
B	• • • —	V	• • • • —
C	— • • •	W	• — • —
D	• — • •	X	• — • • —
E	• • • •	Y	• — • — •
F	• • • • •	Z	• — • — • —
G	• — • • •		
H	• • • • •		
I	• • • •		
J	• — • • •		
K	• • • • •		
L	• — • • •	1	• — • • • •
M	• — • — •	2	• • • • •
N	• — • — •	3	• • • • •
O	• — • — •	4	• • • • •
P	• — • — •	5	• • • • •
Q	• — • — •	6	• • • • •
R	• — • — •	7	• • • • •
S	• — • — •	8	• • • • •
T	• — • — •	9	• • • • •
		0	• • • • •

Exercice 3

Ecrivez une fonction qui accepte un entier positif compris entre 0 et 999 inclus et renvoie une représentation sous forme de chaîne de cet entier écrite en anglais.

Exercice 4

Créez une fonction qui prend une liste de clubs de football avec des propriétés : nom, victoires, défaites, nuls, marqué, concédé et renvoie le nom de l'équipe avec le plus grand nombre de points. Si deux équipes ont le même nombre de points, renvoyez l'équipe avec la plus grande différence de buts. **(Voir plus bas pour les données)**

Exercice 5

Créez une fonction unique qui prend une chaîne ou une liste et renvoie un message codé ou décodé.

La première lettre de la chaîne ou le premier élément de la liste représente le code de caractère (ASCII) de cette lettre. Les éléments suivants sont les différences entre les caractères.

Représentez les caractères avec leur codes ASCII dans un dictionnaire. **(Voir plus bas pour les données)**

Entrée : Hello
Sortie : [72, 29, 7, 0, 3]
Entrée : [79, -4]
Sortie : OK

Exercice 6

Créez une fonction qui prend une chaîne composée de lettres minuscules, de lettres majuscules et de chiffres et renvoie la chaîne triée de la même manière que les exemples ci-dessous.

"Re4r" → "erR4"
"6jnM31Q" → "jMnQ136"
"846ZIbo" → "bIoZ468"

Exercice 7

Créer une fonction qui simule un distributeur automatique. Étant donné une somme d'argent et un numéro_produit, le distributeur automatique doit afficher le nom de produit correct et rendre le montant correct de la monnaie.

Les pièces utilisées pour le rendu sont les suivantes : [500, 200, 100, 50, 20, 10]

La valeur de retour est un dictionnaire avec 2 propriétés :

Produit : le nom du produit sélectionné par l'utilisateur.

Rendu : un tableau de pièces (peut être vide, doit être trié par ordre décroissant).

Si numéro_produit n'est pas valide (hors plage), vous devez renvoyer la chaîne : "Entrez un numéro de produit valide".

Si l'argent ne suffit pas pour acheter un certain produit, vous devez renvoyer la chaîne : "Pas assez d'argent pour ce produit".

Si'il n'y a pas de rendu, retournez un tableau vide comme rendu. **(Voir plus bas pour les données)**

Exercice 8

Ecrire un programme qui lit une phrase cachée depuis un fichier. Le fichier doit contenir un ensemble de phrase chacun dans une ligne. Une phrase aléatoire doit être choisie depuis le fichier.

Le programme affiche la phrase choisie en remplaçant les caractères par '*' puis donne à l'utilisateur la possibilité de deviner la phrase.

Lorsque l'utilisateur saisit une lettre le programme affiche sa position dans la phrase si elle existe sinon le programme affiche une lettre aléatoire pour aider l'utilisateur.

Si L'utilisateur réussie à deviner la phrase, son score sera calculé par rapport au lettre non affichées.

Pour chaque lettre non affichées, l'utilisateur gagne 100 points après la découverte de la phrase.

N.B : Vous pouvez utiliser le module **random** pour générer un indice aléatoire de la lettre à révéler.

Exemple :

Phrase recherchée : Techniques de développement

Affichage initial : *****

Ressources :

Exercice 4

Equipes = [

```
{
  "nom": "Manchester United",
  "victoires": 30,
  "défaites": 3,
  "nuls": 5,
  "marqué": 88,
  "concédé": 20,
},
{
  "nom": "Arsenal",
  "victoires": 24,
  "défaites": 6,
  "nuls": 8,
  "marqué": 98,
  "concédé": 29,
},
{
  "nom": "Chelsea",
  "victoires": 22,
  "défaites": 8,
  "nuls": 8,
  "marqué": 98,
  "concédé": 29,
},
]
```

Exercice 5

Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL	(null)	32	20	040	 	Space	64	40	100	@	@
1	001	SOH	(start of heading)		33	21	041	!	!	65	41	101	A	A
2	002	STX	(start of text)		34	22	042	"	"	66	42	102	B	B
3	003	ETX	(end of text)		35	23	043	#	#	67	43	103	C	C
4	004	EOT	(end of transmission)		36	24	044	$	\$	68	44	104	D	D
5	005	ENQ	(enquiry)		37	25	045	%	%	69	45	105	E	E
6	006	ACK	(acknowledge)		38	26	046	&	&	70	46	106	F	F
7	007	BEL	(bell)		39	27	047	'	'	71	47	107	G	G
8	010	BS	(backspace)		40	28	050	((72	48	110	H	H
9	011	TAB	(horizontal tab)		41	29	051))	73	49	111	I	I
10	A 012	LF	(NL line feed, new line)		42	2A	052	*	*	74	4A	112	J	J
11	B 013	VT	(vertical tab)		43	2B	053	+	+	75	4B	113	K	K
12	C 014	FF	(NP form feed, new page)		44	2C	054	,	,	76	4C	114	L	L
13	D 015	CR	(carriage return)		45	2D	055	-	-	77	4D	115	M	M
14	E 016	SO	(shift out)		46	2E	056	.	.	78	4E	116	N	N
15	F 017	SI	(shift in)		47	2F	057	/	/	79	4F	117	O	O
16	10 020	DLE	(data link escape)		48	30	060	0	0	80	50	120	P	P
17	11 021	DC1	(device control 1)		49	31	061	1	1	81	51	121	Q	Q
18	12 022	DC2	(device control 2)		50	32	062	2	2	82	52	122	R	R
19	13 023	DC3	(device control 3)		51	33	063	3	3	83	53	123	S	S
20	14 024	DC4	(device control 4)		52	34	064	4	4	84	54	124	T	T
21	15 025	NAK	(negative acknowledge)		53	35	065	5	5	85	55	125	U	U
22	16 026	SYN	(synchronous idle)		54	36	066	6	6	86	56	126	V	V
23	17 027	ETB	(end of trans. block)		55	37	067	7	7	87	57	127	W	W
24	18 030	CAN	(cancel)		56	38	070	8	8	88	58	130	X	X
25	19 031	EH	(end of medium)		57	39	071	9	9	89	59	131	Y	Y
26	1A 032	SUB	(substitute)		58	3A	072	:	:	90	5A	132	Z	Z
27	1B 033	ESC	(escape)		59	3B	073	;	;	91	5B	133	[[
28	1C 034	FS	(file separator)		60	3C	074	<	<	92	5C	134	\	\
29	1D 035	GS	(group separator)		61	3D	075	=	=	93	5D	135]]
30	1E 036	RS	(record separator)		62	3E	076	>	>	94	5E	136	^	^
31	1F 037	US	(unit separator)		63	3F	077	?	?	95	5F	137	_	_

Source : www.LookupTables.com

Exercice 7

produits = [

```
{ 'numéro': 1, 'prix': 100, 'nom': 'Orange juice' },
```

```
{ 'numéro': 2, 'prix': 200, 'nom': 'Soda' },
{ 'numéro': 3, 'prix': 150, 'nom': 'Chocolate snack' },
{ 'numéro': 4, 'prix': 250, 'nom': 'Cookies' },
{ 'numéro': 5, 'prix': 180, 'nom': 'Gummy bears' },
{ 'numéro': 6, 'prix': 500, 'nom': 'Condoms' },
{ 'numéro': 7, 'prix': 120, 'nom': 'Crackers' },
{ 'numéro': 8, 'prix': 220, 'nom': 'Potato chips' },
{ 'numéro': 9, 'prix': 80, 'nom': 'Small snack' }
]
```