ALIASE
IA - SERVICE & FORMATION

Formation Python

Atelier Pratique AP-PY1

ALIASE

Atelier Pratique AP-PY1 : Exercice 1.0.0

format() :

Initialiser 3 variables :

jour	*(lundi ou mardi, ...)*
date	*(jj/mm/aaaa)*
temperature	*(25)*

Afficher le message suivant en utilisant la fonction ***format()*** :

'***demain jour date il fera temperature degrés***'

SOLUTION

Atelier Pratique AP-PY1 : Exercice 1.1.2

Les boucles et tableaux

1) Créer un tableau de taille 6 qui contiendra des notes de maths de chaque mois (Jan-Juin) :
nom du tableau : ***notes*** , contenu : 15, 18, 5, 19, 0, 13

2) Parcourir le tableau avec une boucle ***for***. Pour chaque mois, si la note est < 10, afficher le message :
« ***la note en math du mois*** *<no du mois 1-6>* ***est :*** *<note>* ***. Vous devez travailler plus »***

3) Faire une boucle ***while*** pour demander à l'utilisateur de **saisir un mois <1-6>**.
Si mois = 0, arrêter la boucle,
Si mois = 1-6, afficher le message « ***la note en math du mois*** *<no du mois 1-6>* ***est :*** *<note>* ***»***

4) Calculer la note moyenne sur les 6 mois

5) Ajouter un message d'erreur si le mois saisi est incorrect.

6) Créer un tableau ***Tabmois*** avec les **noms des mois (*'Jan' .. 'Juin'*)**
Modifier le code afin d'afficher le nom d'un mois, plutôt que son numéro (1-6)

SOLUTION

Atelier Pratique AP-PY1 : Exercice 1.1.3

Les boucles et tableaux

1) Créer un tableau de taille 10 avec le contenu : 6, 7, 15, 18, 5, 19, 0, 13, 14, 20,

2) Parcourir le tableau avec une boucle ***for***. Pour chaque élement, si le nombre est pair , afficher le message :
« ***le nombre <nombre> est : pair .»*** ***sinon afficher*** « ***le nombre <nombre> est : impair .»***

3) Compter le nombre de valeurs pairs

SOLUTION

Atelier Pratique AP-PY1 : Exercice 1.2.1

Range

a) Créer une liste avec les nombres 10, 20, 30, ..., 100

b) Afficher les éléments à partir du 5ème élément de la liste

c) Afficher 3 éléments à partir du 5ème élément

d) Afficher un élément sur 2 de la liste (sauter un élément sur 2)

SOLUTION

Atelier Pratique AP-PY1 : Exercice 1.4.2

a) Définir une fonction ***totaux (…)***

En entrée : les paramètres suivants: ***nb_ordis , prix_ordi, nb_printers, prix_printer***

En sortie : les deux totaux calculés : ***total HT et total TTC***

b) Définir une fonction ***factureB*** :

En entrée : les paramètres suivants: ***nb_ordis , prix_ordi, nb_printers, prix_printer***

La fonction fera ceci :

. Appeler la fonction ***totaux (…)*** afin de récupérer les totaux HT et TTC calculés,

. Afficher sur l'écran les informations ci-dessous :

quantité ordinateurs	= …	prix unité	= …
quantité imprimantes	= …	prix unité	= …
Total Facture HT	= …		
Total Facture TTC	= …		

SOLUTION

Atelier Pratique AP-PY1 : Exercice 1.4.3

Fonctions

a) Définir une fonction ***puissancen(…)*** *qui* calcule et affiche la puissance *n* d'un nombre *a*
Entrees : *a* et *n*
Sortie : affichage de a puissance n (rappel: a puissance n s'écrit comme ceci en python : a**n)

b) Afficher 3 au carré et 4 au cube en appelant la fonction

c) Demander à l'utilisateur de **saisir *a et n***, *puis a*fficher le résultat de *a* puissance *n*

d) Créer une liste avec les valeurs 1, 2, 3, …, 10

e) Afficher la puissance carrée des 4 premiers éléments de la liste

f) Afficher la puissance carrée de 3 éléments de la liste à partir du 5ème élément

g) Définir une fonction ***fact(n)*** qui retourne le factoriel d'un nombre n (***fact(n) = 1 x 2 x 3 … x n*****.** Afficher *fact(3)*

h) Définir une fonction ***factR(n)*****,** la même que la factoriel ***fact(n)*** mais **récursive.** Afficher *factR(3)*

SOLUTION

Atelier Pratique AP-PY1 : Exercice 1.6.1

Listes

a) Créer une liste avec les éléments : 10, 20, 30, 40, 50
b) Ajouter l'élément 60 à la fin de la liste
c) Modifier les 2 premiers éléments de la liste en 100, 200
d) Modifier les 3 éléments de la liste à partir du 3ème élément en 11, 22, 33
e) Afficher le dernier élément de la liste
f) Afficher l'avant dernier élément de la liste
g) Modifier les 2 derniers éléments de la liste en 0, 1

SOLUTION

Atelier Pratique AP-PY1 : Exercice 1.7.2

a) En utilisant la technique de **DICT COMPREHENSION**, créer un ***dictionnaire k:v*** composé des éléments suivants :

k = 0 . . 19

v = k**2

b) Créer le même dictionnaire en utilisant la méthode du ***zip***.

SOLUTION

Atelier Pratique AP-PY1 : Exercice 1.7.3

ALIASE
IA - SERVICE & FORMATION

Liste des employés

a) Créer la liste (Li1) des employés dans une équipe en ne mettant que leurs noms ('e1', 'e2')
b) Afficher le nombre d'employés dans l'équipe
c) Ajouter un nouvel employé ('e3') dans l'équipe

Annuaire des employés

a) Créer un dictionnaire ***dico*** avec en clé le nom de l'équipe (***'equipe-1'***) et en valeur **la liste** des employés
b) Ajouter deux nouvelles équipes, Li2 avec un employé ('e4') et Li3 avec 4 employés ('e5', 'e6', 'e7', 'e8'))
c) Ajouter un nouvel employé ('e9') dans la 3ème équipe
d) Supprimer la 2ème équipe
e) Retirer le 1er employé dans l'***équipe-3***

SOLUTION

Atelier Pratique AP-PY1 : Exercice 1.8.1

ALIASE
IA - SERVICE & FORMATION

Dictionnaires

a) Créer une liste ***Liste1*** avec les nombres 1, 5, 1, 3, 4, 5
b) Créer un dico ***dico1*** avec chacun des éléments de la liste et son nb d'occurrence
c) Créer une liste ***Liste2*** de prénoms ("Sylvie", "Eric", "Bernard", "Eric", "Catherine", "Sylvie")
d) Créer un dico ***dico2*** avec chacun des éléments de la liste et son nb d'occurrence

Trier les listes

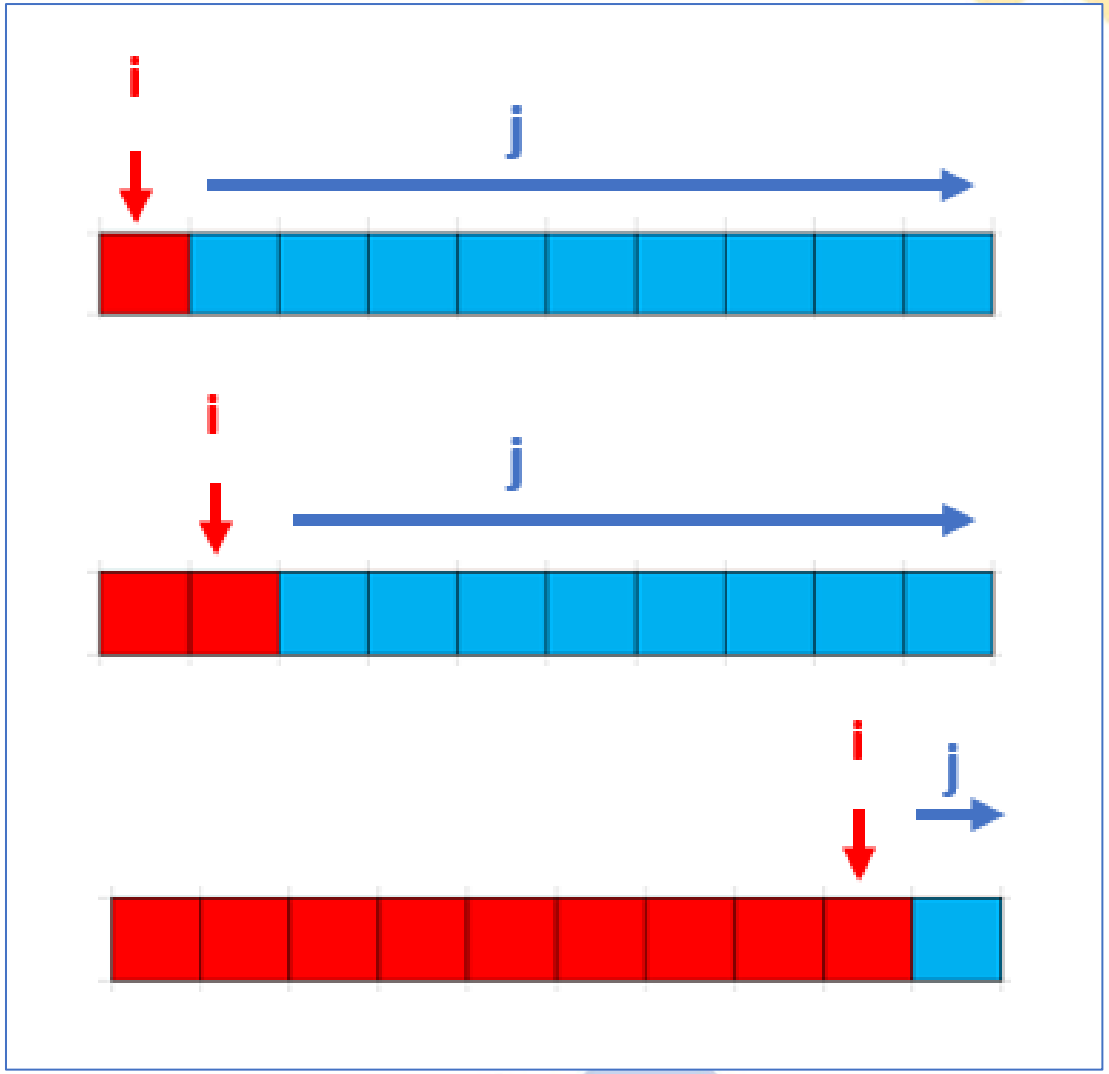
a) Ecrire une fonction (***tri_nb***) qui trie une liste de nombres avec la **méthode de tri par selection**
b) Ecrire une fonction (***tri_str***) qui trie une liste de prénoms du plus court au plus long

SOLUTION

Méthode de tri d'une liste

Tri par selection :

```
tri (Tableau T)
        lg = taille de T
        pour i allant de 0 à lg-1
            pour j allant de i+1 à lg
                si T[j] < T[i]
                    échanger( T[j], T[i] )
```



Atelier Pratique AP-PY1 : Exercice 1.8.2

a) Isoler les deux fonctions de tri dans un fichier **Module1.py** .

b) Dans le fichier de l'exercice précédent importer les deux fonctions à partir du Module1 avant de les appeler

SOLUTION

Atelier Pratique AP-PY1 : Exercice 1.10

ALIASE
IA - SERVICE & FORMATION

Liste des employés

a) Créer une liste avec les noms : ***'anna', 'bernard', 'marie', 'emile', 'joe'***

b) Initiales : pour chaque nom de la liste, afficher le premier caractère du nom.
Si un nom commence et se termine par la même lettre, afficher le nom entier

c) Définir une fonction qui retourne une liste composée des initiales (1er caractère de chaque nom)

d) Définir une fonction qui affiche les noms commençant et se terminant par la lettre passée en paramètre

Annuaire des employés (dictionnaire)

a) Associer à
- l'**équipe-1** la **liste1** avec les noms : ***'anna', 'bernard', 'marie-antoinette', 'emile'***
- l'**équipe-2** la **liste2** avec les noms : ***'Joe', 'michel'***
- l'**équipe-3** la **liste3** avec les noms : ***'frederique', 'sylvie'***

b) Mettre tous les noms de l'**équipe-1** en majuscule

c) Afficher l'employé qui a le nom le plus long ainsi que son équipe

d) Ecrire une fonction mettant les noms d'une seule équipe en majuscule ou en minuscule

e) Ecrire une fonction qui retourne l'employé qui a le nom le plus long ainsi que son équipe

SOLUTION

Atelier Pratique AP-PY1 : Exercice 1.11

Trier les listes

a) Ecrire une fonction ***tri_nb_v1*** identique à la fonction *tri_nb* avec :

. Entrée : une liste à trier . Sortie: la même liste triée

La liste passée en paramètre est triée par la fonction.

Dans le programme appelant, après l'appel de la fonction de tri, l'**état de la liste change** (devient triée).

Li1 = [1, 35, 2, 7, 4, 84]

→ Appel fonction_tri (Li1)

Li1 = [1, 2, 4, 7, 35, 84]

Atelier Pratique AP-PY1 : Exercice 1.11 (cont)

ALIASE
IA - SERVICE & FORMATION

Trier les listes

b) Ecrire une fonction ***tri_nb_v2*** identique à la fonction *tri_nb* et avec la particularité suivante :

. Entrée : une liste

. Sortie : une 2nd liste différente, triée et contenant les même éléments de la liste passée en paramètre

L'état de la liste passée en paramètre **NE change PAS dans le programme appelant** après l'appel.

La fonction doit générer une **nouvelle liste** triée contenant les éléments de la liste passée en paramètres

Li1 = [1, 35, 2, 7, 4, 84]

→ Appel fonction_tri (Li1)

Li1 = [1, 35, 2, 7, 4, 84]

Liste retournée = [1, 2, 4, 7, 35, 84]

SOLUTION

Atelier Pratique AP-PY1 : Exercice 1.12

Random

a) Créer une liste avec les éléments : 'Pierre', 'Marie', 'Jean', 'Sylvie', 'Marc'
b) Tirer aléatoirement un élément de la liste
c) Tirer aléatoirement deux éléments de la liste
d) Modifier la liste en permutant de manière aléatoire les éléments de la liste

e) Tirer un nombre float aléatoire entre 0 et 1 (Ex: 0.63)
f) Tirer un nombre entier aléatoire entre 1 et 25

SOLUTION

Atelier Pratique AP-PY1 : Exercice 1.13

ALIASE
IA - SERVICE & FORMATION

Programmer le jeu *Pendu*

- Tirer un mot aléatoirement dans une liste de mots. Il sera le **mot à deviner**.

- Afficher un **mot masqué** comme ceci **:** '- - - - - - -' ayant la taille du mot à deviner.

- Pour chaque caractère saisi par le joueur, le placer sur le mot masqué s'il fait partie du mot, puis réafficher le mot masqué , Exemple : **'- - e - - t –'**

- l'utilisateur aura droit à ***8*** erreurs

SOLUTION

Atelier Pratique AP-PY1 : Exercice 1.14

ALIASE
IA - SERVICE & FORMATION

Objectif : Déterminer les *relations de parenté* *dans un graphe orienté acyclique*

Etant donné une liste **L** de tuples qui définissent un **graphe** dirigé acyclique.

Un tuple ***(i, j)*** indique que le graphe contient un arc du sommet **i** vers le sommet **j** **(i est ancêtre de j).**

Créer la fonction ***est_ancetre_de(L, a, b)*** qui retourne True si **a** est ancêtre de **b**, sinon False.

Exemple 1 :

L = [(**0, 1**)]

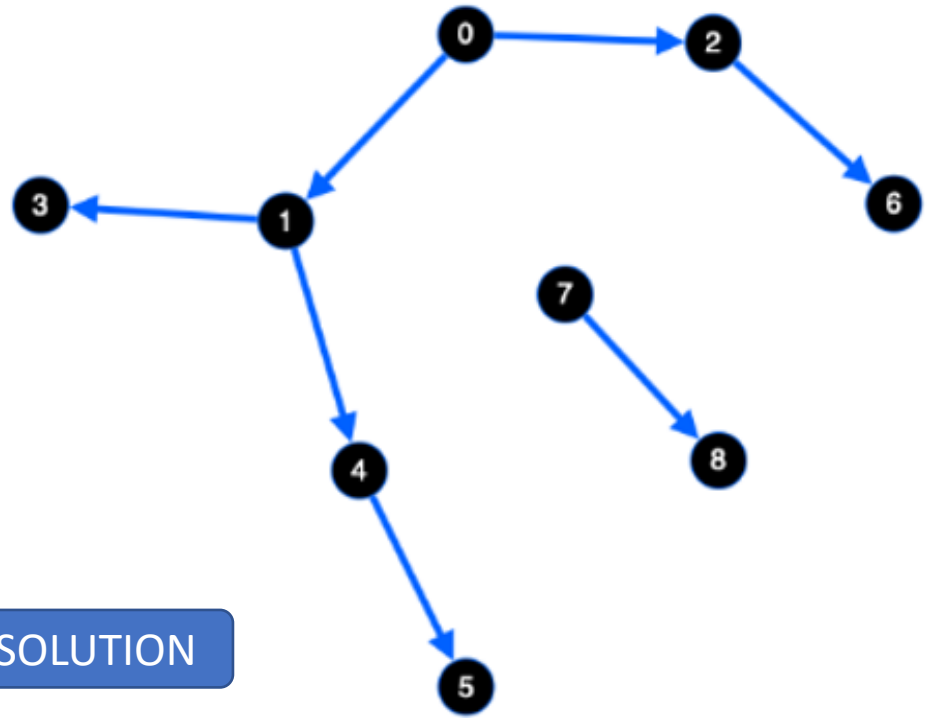
Ici ***est_ancetre_de(L, 0, 1)*** doit retourner **True**

Exemple2 :

L = [(**4, 5**), (**7, 8**), (**0, 2**), (**1, 3**), (**2, 6**), (**0, 1**), (**1, 4**)]

Ici ***est_ancetre_de(L, 1, 5)*** doit retourner **True**

et ***est_ancetre_de(L, 6, 4)*** doit retourner **False**



SOLUTION