

Formation Python

Atelier Pratique AP-PY6

Atelier Pratique AP-PY6 : Exercice 6.12.1

a) Créer dynamiquement un programme qui affiche des **puissances** comme suit :

```
n = 5
p = 2
for i in range(n):
    result = i**p
    print( "{} puissance 2 = {}".format(i, result)
```

b) Modifier le programme. Les deux données n et p sont **saisies par l'utilisateur** .

c) Générer un fichier *prog-exec.py* contenant le code python créé, puis l'exécuter



SOLUTION

Atelier Pratique AP-PY6 : Exercice 6.8.2

Objectif : Créer une fonction *Create_matrice(*args)* :

- Entrée une suite de listes : *liste1, liste2, ..., listen*
- Sortie un tableau *T*

Exemple :

Entrée :

```
liste1 = [1, 2, 3]  
liste2 = [11, 12, 13]  
liste3 = [21, 22, 23]  
liste4 = [31, 32, 33]
```

Sortie :

T :

1	2	3
11	12	13
21	22	23
31	32	33



SOLUTION

Atelier Pratique AP-PY6 : Exercice 6.8.3

Objectif : Créer une fonction `Create_matrice( *args )` :

- Entrée une suite de listes : *liste1, liste2, ..., listen*
- Sortie un tableau ***T*** , qui est en fait une **transposée** où les colonnes deviennent des lignes

En utilisant une boucle `for v1, v2, ..., vn in zip(liste1, liste2, ..., listen):`

Exemple :

Entrée : `liste1 = [1, 2, 3]`
`liste2 = [11, 12, 13]`
`liste3 = [21, 22, 23]`
`liste4 = [31, 32, 33]`

↑
colonne

Sortie :

T :

1	11	21	31
2	12	22	23
3	13	23	33

← ligne

➡
SOLUTION

Atelier Pratique AP-PY6 : Exercice 6.13

Appliquer la méthode **sys.path** au jeu **Pendu**

- a) Placer le programme pendu.py dans le dossier C:\Users\uname\test
- b) Créer un dossier modules : C:\Users\uname\test\modules
- c) Placer le module fonctions.py dans le dossier C:\Users\uname\test\modules
- d) Lancer pendy.py à partir du dossier C:\Users\uname\test, et observez le message d'erreur
- e) Apporter les modifications nécessaires et retester



SOLUTION

Atelier Pratique AP-PY6 : Exercice 6.14.1

- a) Affecter le nombre hexadécimal **0xF** à une variable **a**
- b) Afficher la variable **a** en integer, binaire et hexadécimal
- c) Afficher le type de chacune des 3 sorties précédentes

- d) Affecter le nombre décimal 10 à la variable **a**
- e) Afficher la variable **a** en binaire et hexadécimal

- f) Affecter 10 à la variable **a** et 11 à la variable **b**, puis les afficher en binaire
- g) Afficher le résultat de **a OR b** et **a AND b**

- h) Décaler la variable **a** d'un seul bit vers la **gauche**. Afficher le résultat en binaire et en integer
- i) Décaler la variable **a** d'un seul bit vers la **droite**. Afficher le résultat en binaire et en integer

- j) Affecter 10 à la variable **a**. Afficher le bit de rang n de la variable à partir de la droite (n = 0, 1, 2, 3, ...)

Atelier Pratique AP-PY6 : Exercice 6.14.1 (Cont)

- k) Ecrire une fonction **puissance**. Entrée un nombre **a** et une puissance **n**. Sortie: **a puissance n**
- l) Ecrire une fonction **bin_to_int**.
En entrée : un nombre **binaire** et sa **taille** en nb de bits. En sortie: la valeur **décimale** correspondants
Calculer le nombre maximum (décimal) que l'on peut écrire sur un octet (8 bits)
- m) Ecrire une fonction **val_bit**.
Entrées: un nombre binaire et un rang n
Sortie: le bit de rang n à partir de la droite (n = 0, 1, 2, 3, ...)
- n) Ecrire une fonction **create_mask**.
Entrée: un rang n.
Sortie: un MASK où le bit de rang n est positionné à "1", (n = 0, 1, 2, 3, ...)
Afficher le MASK en binaire
- o) Afficher un nombre en binaire format 16 bits, 32 bits, 64 bits



SOLUTION