Frédéric
LURET

Python
Exercices niveau 1

Université de TOURS

1 Niveau facile

Question 1

Écrire un programme qui demande votre âge et qui écrit « Votre âge est... » avec l'âge qui s'affiche.

Code python :

```python
age = input('Quel est votre age ? ')
print(f'Votre âge est {age}')
```

Question 2

Écrire un programme qui demande l'année de naissance de l'utilisateur et qui lui donne son age en 2040

Code python :

```python
annee = input('Quelle est votre année de naissance ? ')
print(f'Votre age en 2040 sera de {2040 - int(annee)} ans')
```

Question 3

Dans une école de Rugby, il y a quatre groupes :
— le groupe U8 pour les joueurs entre 8 ans inclus et 10 ans exclus ;
— le groupe U10 pour les joueurs entre 10 ans inclus et 12 ans exclus ;
— le groupe U12 pour les joueurs entre 12 ans inclus et 14 ans exclus ;
— le groupe U14 pour les joueurs entre 14 ans inclus et 16 ans exclus.
Écrire le script pour qu'il affiche le groupe lorsque l'utilisateur entre l'âge du joueur.

Code python :

```python
# Demander l'âge du joueur
age = int(input("Entrez l'âge du joueur : "))

# Vérifier à quel groupe appartient le joueur
if 8 <= age < 10:
    print("Le joueur est dans le groupe U8.")
elif 10 <= age < 12:
    print("Le joueur est dans le groupe U10.")
elif 12 <= age < 14:
    print("Le joueur est dans le groupe U12.")
elif 14 <= age < 16:
    print("Le joueur est dans le groupe U14.")
else:
    print("Âge hors des catégories de groupe.")
```

Question 4

Écrire un programme qui demande à l'utilisateur son nom et son sexe (M ou F). En fonction de ces données, afficher « Cher Monsieur » ou « Chère Madame » suivi du nom de l'utilisateur.

Code python :

```python
# Demander le nom de l'utilisateur
nom = input("Entrez votre nom : ")

# Demander le sexe de l'utilisateur
sexe = input("Entrez votre sexe (M ou F) : ")

# Vérifier le sexe et afficher le message approprié
if sexe.upper() == 'M':
    print(f"Cher Monsieur {nom}")
elif sexe.upper() == 'F':
    print(f"Chère Madame {nom}")
else:
    print("Sexe non reconnu. Veuillez entrer M ou F.")
```

Question 5

Écrire un programme qui demande 2 entiers A et B, puis renvoie le quotient et le reste de la division euclidienne de A par B.

Code python :

```python
# Demander les deux entiers A et B
A = int(input("Entrez l'entier A : "))
B = int(input("Entrez l'entier B : "))

# Calculer le quotient et le reste de la division euclidienne
quotient = A // B
reste = A % B

# Afficher le quotient et le reste
print(f"Le quotient de la division de {A} par {B} est {quotient}")
print(f"Le reste de la division de {A} par {B} est {reste}")
```

Question 6

Écrire un algorithme qui demande le prénom d'un utilisateur puis qui lui dit combien il y a de lettres dans son prénom.

Code python :

```python
# Demander le prénom de l'utilisateur
prenom = input("Entrez votre prénom : ")

# Calculer le nombre de lettres dans le prénom
nombre_de_lettres = len(prenom)

# Afficher le nombre de lettres
print(f"Votre prénom contient {nombre_de_lettres} lettres.")
```

Question 7

Rédiger un programme Python sous le nom bissextile.py qui, pour une variable recevant une valeur entière, indique si l'année correspondante est bissextile en affichant : « Cette année est bissextile » ou « Cette année n'est pas bissextile » selon le cas.

Vérifier si l'année est bissextile en utilisant les règles données :

— Une année est bissextile si elle est divisible par 4.

— Cependant, si l'année est divisible par 100, elle n'est pas bissextile, sauf si elle est aussi divisible par 400.

Tester votre programme avec les années 2000, 2013, et 2100.

Écrire ensuite une seconde version de ce fichier qui teste ces 3 valeurs à partir d'une liste sans demander de valeur à l'utilisateur.

Code python :

Version 1 :

```python
annee = int(input("Entrez une année : "))

if (annee % 4 == 0 and annee % 100 != 0) or (annee % 400 == 0):
    print("Cette année est bissextile")
else:
    print("Cette année n'est pas bissextile")
```

Version 2 :

```python

annees_a_tester = [2000, 2013, 2100]
for annee in annees_a_tester:
    if (annee % 4 == 0 and annee % 100 != 0) or (annee % 400 == 0):
        print(f"L'année {annee} est bissextile")
    else:
        print(f"L'année {annee} n'est pas bissextile")
```

2 Boucles et fonctions

Question 1

Écrire un programme qui affiche tous les nombres impairs entre 0 et 15000, dans l'ordre croissant.

Code python :

```python
# Afficher tous les nombres impairs entre 0 et 15000
for nombre in range(0, 15001):
    if nombre % 2 != 0:
        print(nombre)
```

Question 2

Écrire un programme qui affiche tous les nombres pairs entre 0 et 15000, dans l'ordre décroissant.

Code python :

```python
# Afficher tous les nombres impairs entre 0 et 15000
for nombre in range(0, 15001):
    if nombre % 2 == 0:
        print(nombre)
```

Question 3

Écrire un programme qui calcule factorielle de n, un entier demandé à l'utilisateur au début du programme. il faudra nécessairement créer une fonction. (factorielle n est le produit de tous les entiers de 1 à n) :

Code python :

```python
# Demander à l'utilisateur de saisir un entier
n = int(input("Entrez un entier pour calculer sa factorielle : "))

# Calculer la factorielle de n
if n == 0:
    resultat = 1
else:
    resultat = 1
    for i in range(1, n + 1):
        resultat *= i

# Afficher la factorielle de n
print(f"La factorielle de {n} est {resultat}")
```

Question 4

Écrire un programme qui calcule puis affiche le produit des nombres pairs compris entre 1 et n, avec n étant une variable demandée à l'utilisateur :

Code python :

```python
# Demander à l'utilisateur de saisir un entier
n = int(input("Entrez un entier : "))

# Initialiser le produit des nombres pairs
produit = 1

# Calculer le produit des nombres pairs entre 1 et n
for i in range(1, n + 1):
    if i % 2 == 0:
        produit *= i

# Afficher le produit des nombres pairs
print(f"Le produit des nombres pairs entre 1 et {n} est {produit}")
```

Question 5

Écrire un programme qui demande les deux côtés adjacents à l'angle droit d'un triangle rectangle et qui donne en réponse la longueur de l'hypoténuse, ainsi que le périmètre de ce triangle. On utilisera une fonction par calcul et la fonction sqrt() du module math.

Code python :

```python
import math

def calculer_hypotenuse(a, b):
    """Calcule la longueur de l'hypoténuse d'un triangle rectangle."""
    return math.sqrt(a**2 + b**2)

def calculer_perimetre(a, b, hypotenuse):
    """Calcule le périmètre d'un triangle rectangle."""
    return a + b + hypotenuse

# Demander à l'utilisateur de saisir les longueurs des deux côtés adjacents
a = float(input("Entrez la longueur du premier côté adjacent à l'angle droit : "))
b = float(input("Entrez la longueur du deuxième côté adjacent à l'angle droit : "))

# Calculer l'hypoténuse et le périmètre
hypotenuse = calculer_hypotenuse(a, b)
perimetre = calculer_perimetre(a, b, hypotenuse)

# Afficher les résultats
print(f"La longueur de l'hypoténuse est {hypotenuse}")
print(f"Le périmètre du triangle est {perimetre}")
```

Question 6

Écrire un programme qui affiche un damier carré de taille n. Par exemple pour n = 5 :

X O X O X
O X O X O
X O X O X
O X O X O
X O X O X

Code python :

```python
# Demander à l'utilisateur de saisir un entier pour la taille du damier
n = int(input("Entrez la taille du damier : "))

# Générer et afficher le damier
for i in range(n):
    ligne = ""
    for j in range(n):
        if (i + j) % 2 == 0:
            ligne += "X"
        else:
            ligne += "O"
    print(ligne)
```

Question 7

Écrire un programme qui affiche un damier carré de taille n avec des cases de taille c. Par exemple n = 4 et c = 3 :

XXX OOO XXX OOO
XXX OOO XXX OOO
XXX OOO XXX OOO
OOO XXX OOO XXX
OOO XXX OOO XXX
OOO XXX OOO XXX
XXX OOO XXX OOO
XXX OOO XXX OOO
XXX OOO XXX OOO
OOO XXX OOO XXX
OOO XXX OOO XXX
OOO XXX OOO XXX
XXX OOO XXX OOO
XXX OOO XXX OOO
XXX OOO XXX OOO
OOO XXX OOO XXX
OOO XXX OOO XXX
OOO XXX OOO XXX

Code python :

```python
# Demander à l'utilisateur de saisir un entier pour la taille du damier
n = int(input("Entrez la taille du damier : "))

# Demander à l'utilisateur de saisir un entier pour la taille des cases
c = int(input("Entrez la taille des cases : "))

# Générer et afficher le damier
for i in range(n):
    for k in range(c):  # Répéter chaque ligne c fois pour la hauteur des cases
        ligne = ""
        for j in range(n):
            if (i + j) % 2 == 0:
                ligne += "X" * c  # Répéter le caractère c fois pour la largeur des cases
            else:
                ligne += "O" * c
        print(ligne)
```

3 Les tuples

Question 1

1. Créer une variable age contenant la valeur 81.
2. Créer un tuple nommé personne contenant les valeurs "Dupont", "Maurice", age, 25, 59000 rangées dans cet ordre.
3. Quelle instruction entre-t-on pour accéder à la première valeur du tuple personne ?
4. Effectuer l'instruction personne[2]. Qu'observez vous ?
5. Modifier la valeur de la variable age et effectuer à nouveau l'instruction personne[2]. Que se passe-t-il ?

Code python :

Après avoir modifié la valeur de la variable age, la valeur de personne[2] reste 81.

Cela montre que les tuples sont immuables et que la modification de la variable age n'affecte pas le contenu du tuple personne.

```python
# Créer une variable age contenant la valeur 81
age = 81
print(f'Adresse mémoire de age {id(age)}')

# Créer un tuple nommé personne
personne = ("Dupont", "Maurice", age, 25, 59000)

# Accéder à la première valeur du tuple personne
premiere_valeur = personne[0]
print(f"La première valeur du tuple personne est : {premiere_valeur}")

# Effectuer l'instruction personne[2]
valeur_age = personne[2]
print(f'Adresse mémoire de personne[2] {id(personne[2])}')

print(f"La valeur de personne[2] est : {valeur_age}")

# Modifier la valeur de la variable age
age = 82
print(f'Adresse mémoire de age {id(age)}')

# Effectuer à nouveau l'instruction personne[2]
valeur_age_modifiee = personne[2]
print(f'Adresse mémoire de personne[2] {id(personne[2])}')
print(f"Après modification de la variable age, la valeur de personne[2]
↪  est toujours : {valeur_age_modifiee}")
```

Question 2

Déterminez ce qu'il se passe lorsque l'on cherche à obtenir un fragment de tuple (slice) dans les cas suivants :

— le second indice est plus petit que le premier ;
— le second indice est plus grand que la taille du tuple.

Code python :

— Second indice plus petit que le premier : Lorsque le second indice est plus petit que le premier, le résultat est un tuple vide (). Cela s'explique par le fait que les slices en Python parcourent les éléments de gauche à droite, et un second indice plus petit que le premier ne correspond à aucun élément dans cet ordre.
— Second indice plus grand que la taille du tuple : Lorsque le second indice est plus grand que la taille du tuple, Python retourne tous les éléments du premier indice jusqu'à la fin du tuple. Python ne lève pas d'erreur et gère les indices hors limites en les ajustant à la taille du tuple.

```python
# Créer un tuple d'exemple
exemple_tuple = (1, 2, 3, 4, 5)
print(exemple_tuple)
# Cas où le second indice est plus petit que le premier
slice_inverse = exemple_tuple[3:1]
print(f"Slice avec second indice plus petit que le premier (3:1) : {slice_inverse}")

# Cas où le second indice est plus grand que la taille du tuple
slice_out_of_bounds = exemple_tuple[2:10]
print(f"Slice avec second indice plus grand que la taille du tuple (2:10) : {slice_out_of_bounds}")
```

Question 3

Écrire une fonction "trouve" qui prend en argument un tuple et un élément et qui renvoie la position de cet élément dans le tuple. Elle renverra -1 si l'élément n'est pas trouvé.

Code python :

```python
def trouve(tup, element):
    """Renvoie la position de l'élément dans le tuple, ou -1 si l'élément n'est pas trouvé."""
    for index, value in enumerate(tup):
        if value == element:
            return index
    return -1


# Exemple d'utilisation
exemple_tuple = (1, 2, 3, 4, 5)

ELEMENT_A_TROUVER = 3
position = trouve(exemple_tuple, ELEMENT_A_TROUVER)
print(f"La position de l'élément {ELEMENT_A_TROUVER} dans le tuple est : {position}")

ELEMENT_A_TROUVER = 6
position = trouve(exemple_tuple, ELEMENT_A_TROUVER)
print(f"La position de l'élément {ELEMENT_A_TROUVER} dans le tuple est : {position}")
print(exemple_tuple.index(3))
print(exemple_tuple.index(6))
```

Question 4

Écrire une fonction qui prend en argument un tuple composé de nombres entiers et

renvoie un tuple contenant le plus grand des entiers et le plus petit.

Code python :

```python
def trouve_min_max(tup):
    """Renvoie un tuple contenant le plus petit et le plus grand entier du tuple."""
    if not tup:  # Vérifier si le tuple est vide
        return None, None
    min_val = min(tup)
    max_val = max(tup)
    return min_val, max_val


# Exemple d'utilisation
exemple_tuple = (1, 2, 3, 4, 5)
min_max = trouve_min_max(exemple_tuple)
print(f"Le plus petit et le plus grand entier du tuple sont : {min_max}")

exemple_tuple_vide = ()
min_max_vide = trouve_min_max(exemple_tuple_vide)
print(f"Le plus petit et le plus grand entier du tuple vide sont : {min_max_vide}")
```

Question 5

Écrire une fonction multiplie qui prend en argument un tuple appelé nombres composé de nombres, et un entier naturel n non nul et renvoie un nouveau tuple obtenu en multipliant chaque élément du tuple par n.

Code python :

```python
def multiplie(nombres, n):
    """Renvoie un nouveau tuple obtenu en multipliant chaque élément du tuple par n."""
    retour = []
    for val in nombres:
        retour.append(val * n)
    return tuple(retour)
# Exemple d'utilisation
exemple_tuple = (1, 2, 3, 4, 5)
n = 3
nouveau_tuple = multiplie(exemple_tuple, n)
print(f"Tuple de départ : {exemple_tuple}")
print(f"Le nouveau tuple obtenu en multipliant chaque élément par {n} est : {nouveau_tuple}")
```

Question 6

Écrire une fonction "separe" qui prend en argument un tuple composé de nombres entiers et renvoie un tuple contenant deux tuples : le premier ne contenant que les entiers pairs du tuple de départ et le deuxième que les entiers impairs.

Code python :

```python
def separe(nombres):
    """Renvoie un tuple contenant deux tuples : les entiers pairs et
    ↪ les entiers impairs."""
    pairs = []
    impairs = []
    for val in nombres:
        if val % 2 == 0:
            pairs.append(val)
        else:
            impairs.append(val)
    return (tuple(pairs), tuple(impairs))

# Exemple d'utilisation
exemple_tuple = (1, 2, 3, 4, 5, 6, 7, 8, 9, 10)
resultat = separe(exemple_tuple)
print(f"Tuple de départ : {exemple_tuple}")
print(f"Tuple des entiers pairs : {resultat[0]}")
print(f"Tuple des entiers impairs : {resultat[1]}")
```

4 Les listes

Question 1

Écrivez un programme qui, à partir de deux tableaux à une dimension, créé un tableau à deux dimensions contenant les produits des termes de chaque tableau.

Par exemple, si on a deux tableaux [2,3] et [4,5] le tableau en sortie devra être [[8,10],[12,15]].

Code python :

```python
def produit_tableaux(tab1, tab2):
    resultat = []
    for i in range(len(tab1)):
        ligne = []
        for j in range(len(tab2)):
            ligne.append(tab1[i] * tab2[j])
        resultat.append(ligne)
    return resultat

# Exemple d'utilisation
tab1 = [2, 3]
tab2 = [4, 5]

tableau_2d = produit_tableaux(tab1, tab2)
print(tableau_2d)
```

Question 2

Écrivez un script affichant toutes les dates de l'année 2020 sachant qu'elle est bissextile (il y a un 29 février) et qu'elle commence un mercredi. On utilisera trois tableaux :

— un tableau des jours de la semaine ;
— un tableau des mois de l'année ;
— un tableau du nombre de jours dans chaque mois de l'année.

Par exemple le résultat devra commencer par les lignes suivantes :

mercredi 1 janvier
jeudi 2 janvier
vendredi 3 janvier

Code python :

```python
# Tableau des jours de la semaine
jours_semaine = [
    "Lundi",
    "Mardi",
    "Mercredi",
    "Jeudi",
    "Vendredi",
    "Samedi",
    "Dimanche",
]

# Tableau des mois de l'année
mois_annee = [
    "Janvier",
    "Février",
    "Mars",
    "Avril",
    "Mai",
    "Juin",
    "Juillet",
    "Août",
    "Septembre",
    "Octobre",
    "Novembre",
    "Décembre",
]

# Tableau du nombre de jours dans chaque mois (année bissextile)
jours_par_mois = [31, 29, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31]

# Le 1er janvier 2020 est un mercredi (indice 2 dans la liste des jours)
jour_index = 2

# Parcours de chaque mois
for i in range(12):
    mois = mois_annee[i]
    jours_dans_ce_mois = jours_par_mois[i]

    # Parcours des jours du mois
    for jour in range(1, jours_dans_ce_mois + 1):
        # Affichage de la date
        print(f"{jours_semaine[jour_index]} {jour} {mois} 2020")

        # Incrémentation du jour de la semaine
        jour_index = (jour_index + 1) % 7
```

1. jours_semaine contient les jours de la semaine.
2. mois_annee contient les noms des mois.

3. jours_par_mois contient le nombre de jours dans chaque mois en 2020 (29 jours pour février car 2020 est une année bissextile).
4. Le script commence par le 1er janvier 2020, qui est un mercredi, avec l'indice 2 pour les jours de la semaine.
5. Ensuite, il parcourt chaque jour de chaque mois en ajustant l'indice des jours de la semaine.

Ce script affichera toutes les dates de l'année 2020.